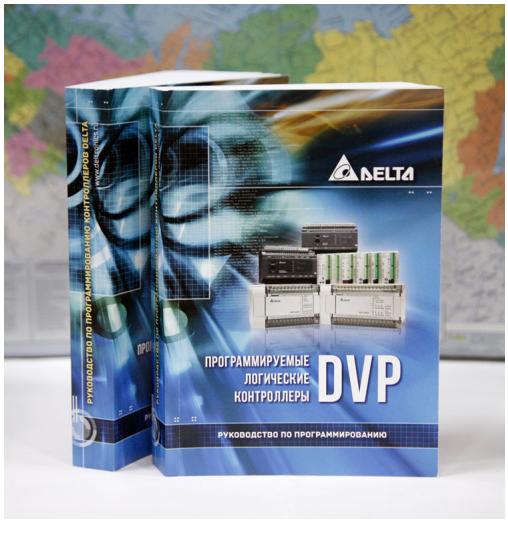
Программируемые логические контроллеры DVP

SS / SA / SX / SV / ES / EX / EH

Руководство по программированию



Вышло в свет 2-е печатное издание книги «Программируемые логические контроллеры DVP. Руководство по программированию».

В новом издании вы найдете инструкции по программированию контроллеров второго поколения: **SS2 / SA2 / SX2 / ES2 / EX2 / SE**

Инструкция продается только в печатном виде и не будет распространяться электронно.

По вопросам приобретения книги обращайтесь:

- по e-mail: sales@deltronics.ru
- по телефону (495) 661-24-61

ВВЕДЕНИЕ

Программируемые логические контроллеры семейства Delta DVP являются идеальным средством для построения высокоэффективных систем автоматического управления при минимальных затратах на приобретение оборудования и разработку системы.

Настоящее руководство по программированию описывает и поясняет все команды, инструкции, операнды и адресацию, которые нужны для написания программ контроллеров Delta DVP серий ES / EX / SS / SA / SX / SC / SV / EH /EH2.

Для отладки и написания программ предусмотрен пакет программирования WPLSoft, который не требует больших ресурсов компьютера и является простым инструментом для всех категорий специалистов. Используются три языка программирования: LD (ступенчатые диаграммы или релейно-контактная логика), IL (список инструкций), SFC (последовательные функциональные диаграммы). Описание программного продукта WPLSoft приведено в отдельном Руководстве пользователя.

Информация по аппаратной части, установке, монтажу, вводу в эксплуатацию, обслуживанию и устранению ошибок есть в соответствующих Руководствах по эксплуатации на каждую серию контроллеров.

ГЛАВА 1

Базовые понятия и принципы программирования промышленных контроллеров

Предисловие

Краткая история создания и функционирование промышленных контроллеров

Промышленный контроллер, именуемый также Программируемый Логический Контроллер или сокращенно ПЛК (англ. PLC – Programmable Logic Controller) относится к разряду электронных устройств. Ранее они назывались «Последовательные Контроллеры» (Sequence Controller) и были переименованы в ПЛК Национальной Ассоциацией Производителей Электрического Оборудования США (NEMA – National Electrical Manufacture Association) в 1978 году. Тогда же контроллеры были отнесены к классу электронного оборудования.

Любой программируемый логический контролер работает по следующей обобщенной схеме:
Шаг 1. – Чтение состояния входов, таких как датчики, клавиатуры, импульсы и кнопки.
Шаг 2. – Исполнение микропроцессором программы, заложенной в ПЛК пользователем и состоящей из логических связей, различных последовательностей, циклов, таймеров, счетчиков, формул и т.п.
Шаг 3. – Формирование выходных сигналов в соответствии с результатами, полученными в ходе обработки заложенной в ПЛК программы. В качестве выходных сигналов выступают открытие или закрытие реле (транзистора), запуск процедуры по управлению технологической установкой или процессом, формирование аналоговой величины или цифрового значения.

ПЛК могут использоваться совместно с различным периферийным оборудованием, например с панелями оператора, управляемым приводом, персональным компьютером и т.п. В связи с этим контроллеры отвечают строгим требованиям, предъявляемым к промышленным электронным устройствам и касающихся, например, электромагнитной совместимости, способности работать в коммуникационной сети, выполнять специфические задачи.

В системе автоматического управления ПЛК является важнейшим звеном.

Общепринятыми типами и обозначениями входных/выходных сигналов контроллера являются:

DI (Digital Input) – дискретный вход
AI (Analog Input) – аналоговый вход
PI (Pulse Input) – импульсный вход
DO (Digital Output) – дискретный выход
AO (Analog Output) – аналоговый выход
PO (Pulse Output) – импульсный выход

1.1 Принципы построения ступенчатых диаграмм

Большинство производителей промышленных контроллеров используют стандартный инструмент программирования – язык ступенчатых диаграмм (англ. LD – Ladder Diagram), который в отечественной практике автоматизации называется еще как релейно-контактные схемы. Данный язык является универсальным средством отображения схемы автоматического управления.

Ступенчатые диаграммы были изобретены во время Второй мировой войны и первоначально включали лишь базовые компоненты такие как: контакт А (нормально открытый), контакт В (нормально закрытый), выходная катушка, таймер, счетчик и ряд других элементов. Обычный пост управления состоит в основном из данных устройств.

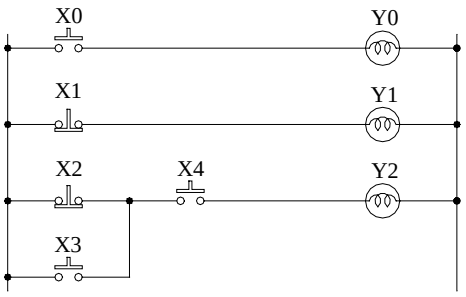
Однако, традиционный набор элементов не в состоянии выполнять математические и логические операции, сравнивать и преобразовывать величины, выполнять прикладные инструкции. Необходимость решения данных задач в ходе управления технологическими объектами и привела к созданию нового класса устройств – промышленные контроллеры.
В связи с тем, что контроллеры исторически создавались в развитие постов управления, к ним по наследству перешел и язык ступенчатых диаграмм. Данный термин теперь применяется именно по отношению к ПЛК.

Традиционные ступенчатые диаграммы и разработанные для контроллеров внешне очень похожи, а также построены по тем же принципам. Отличие заключается в том, что при создании традиционной диаграммы вид условных значков на схеме старались максимально приблизить к виду исходного объекта, а значки, используемые для контроллеров, адоптированы для исполнения диаграмм на компьютере или отображения в современном описании на изделие (datasheet).

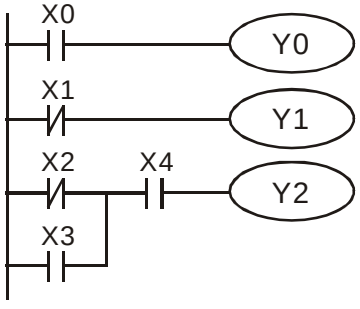
Логику формирования ступенчатой диаграммы можно подразделить на два подхода:
Комбинационная логика и Последовательная логика, отличие между которыми приведены ниже.

1. Комбинационная логика.
При данной логике схема состоит из независимых друг от друга фрагментов, положение которых в схеме роли не играет. Входы и выходы различных фрагментов не оказывают друг на друга никакого влияния.
На следующем примере изображена комбинационная ступенчатая диаграмма в традиционном виде и в форме для контроллеров:

Традиционная ступенчатая диаграмма (релейно-контактная электросхема)



Ступенчатая диаграмма в форме для ПЛК



Фрагмент 1.

Состоит из одного нормально открытого контакта X0, называемый еще как контакт или кнопка типа A, и выхода Y0. Исходным состоянием контакта X0, т.е. когда кнопка не нажата, является ВЫКЛ. Соответственно выход Y0 в исходном состоянии также будет ВЫКЛ. При нажатии кнопки контакт X0 замкнется, выходная катушка также перейдет в состояние ВКЛ.

Фрагмент 2.

Состоит из одного нормально закрытого контакта X1, называемый еще как контакт или кнопка типа B, и выхода Y1. В отличие от контакта X0 исходным состоянием контакта X1, т.е. когда кнопка не нажата, является ВКЛ. Соответственно выход Y1 в исходном состоянии также будет ВКЛ. Но при нажатии кнопки контакт X1 разомкнется и выходная катушка Y1 перейдет в состояние ВЫКЛ.

Фрагмент 3.

Представляет из себя комбинацию входных контактов X2 (нормально закрытый), X3 и X4 (нормально открытые). Выход Y2 перейдет в состояние ВКЛ. при условии, что будут замкнуты или X2 (кнопка не нажата) и X4 (кнопка нажата), или X3 и X4 (кнопки нажаты), или X2 (кнопка не нажата) и X4 (кнопка нажата).

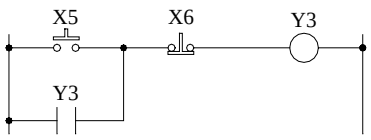
2. Последовательная логика

При данной логике выход одного шага схемы является входным условием для другого. Таким образом входы и выходы различных фрагментов схемы являются логически связанными и положение каждого фрагмента в схеме строго определено. Такие схемы нельзя «распараллелить».

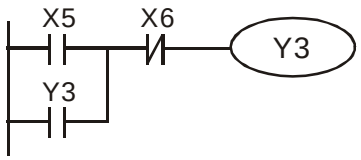
Для примера рассмотрим частный случай последовательной логики – схему с самоблокировкой выхода. В данной схеме условием включения выхода могут быть несколько параллельных входных контактов, один из которых логически связан с выходом. Когда выход замыкается, связанный с ним входной контакт также замыкается и выход самоблокируется, т.е. его состояние больше не зависит от состояния входных контактов. Таким образом, при одних и тех же входных условиях выход может принимать разные состояния.

Данная схема продемонстрирована ниже:

Традиционная ступенчатая диаграмма (релейно-контактная электросхема)



Ступенчатая диаграмма в форме для ПЛК



Когда на схему подается питание первый раз, выход Y3 будет в состоянии ВЫКЛ. Хотя контакт X6 и замкнут, но контакт X5 разомкнут и цепь соответственно тоже.

В данной схеме знаком Y3 обозначается как выход (катушка), так и входной контакт, состояние которого зависит от состояния выхода (катушки). Таким образом, в первоначальный момент связанный контакт Y3 будет также разомкнут, так как катушка Y3 не замкнута.

При замыкании контакта X5 сигнал пройдет через нормально замкнутый контакт X6 и переведет выход (катушку) Y3 в замкнутое состояние, что в свою очередь замкнет связанный входной контакт Y3. Схема перейдет в состояние самоблокировки. Теперь положение контакта X5 больше не оказывает влияния на состояние выхода (катушки) Y3.

Чтобы разомкнуть выход Y3, необходимо нажать на кнопку X6, т.е. разомкнуть цепь. Это приведет к размыкаю выхода (катушки) Y3 и взаимосвязанного с ней входного контакта Y3. Схема разблокируется. Логические связи и состояния по шагам в данной схеме приведены в таблице ниже:

Состояние / Шаг	X5	X6	Y3
1	неактивен	неактивен	ВЫКЛ
2	активен	неактивен	ВКЛ
3	неактивен	неактивен	ВКЛ
4	неактивен	активен	ВЫКЛ
5	неактивен	неактивен	ВЫКЛ

Примечание:
Важным моментом является то, что для нормально открытого и закрытого контактов состояние «активен» является диаметрально противоположным.
У нормально открытого контакта состояние «активен» соответствует замыканию цепи при нажатии кнопки (срабатывании датчика).
Для нормально закрытого контакта состояние «активен» соответствует разрыву цепи при нажатии кнопки (срабатывании датчика).

Другие примеры подобных схем с применением различных элементов приведены в Главе 3.

1.2 Различия в работе традиционных релейно-контактных схем и ступенчатых диаграмм ПЛК

Несмотря на общность подходов в логике работы релейно-контактных электросхем и ступенчатых диаграмм ПЛК, между ними существует два кардинальных отличия.
Первое и основное отличие заключается в том, что релейно-контактные электросхемы отражают работу реально существующих приборов (счетчиков, таймеров, контактов), а контроллер эмулирует их в своем процессоре.
Так как в ПЛК всего один процессор, то контроллер вынужден обрабатывать программу последовательно шаг за шагом, что порождает определенную временную задержку. Чем длиннее программа, тем больше это время задержки. Контроллер сначала опрашивает **все** входы, затем обрабатывает **всю** программу, и только после этого обновляет **все** свои физические выходы.
Таким образом, контроллеры работают циклично: опрос входов – отработка программы – установка выходов. Один полный цикл еще называется сканом, а сам процесс циклическим сканированием.

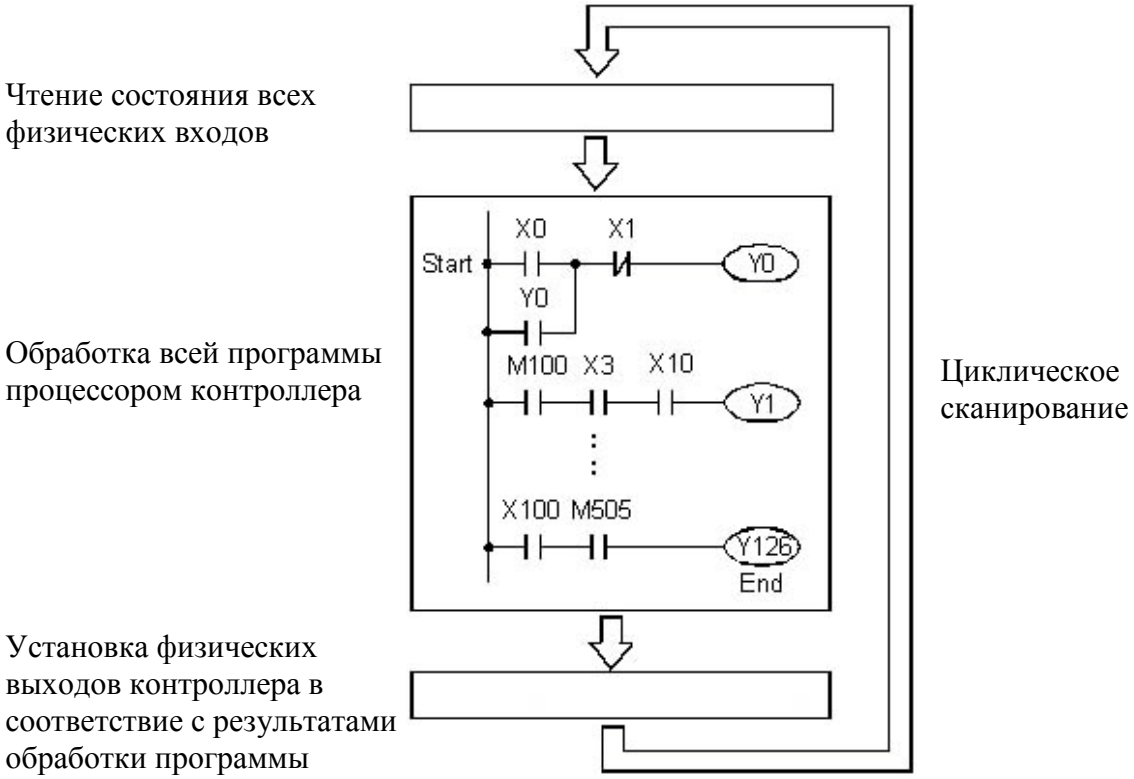
В традиционных схемах, где все элементы существуют физически, временная задержка не возникает, так как каждый прибор самостоятельно воспринимает свой входной сигнал, сам его отрабатывает, а затем обновляет свой выход. Таким образом, любые изменения на входах сразу же отрабатываются путем установки выходов в соответствии с логикой релейно-контактной схемы.

Помимо задержки на отработку программы, в контроллерах существует еще задержка на реакцию входов (цифровой фильтр), а также время на изменение физических выходов, что в совокупности дает еще большую задержку реакции на изменение во внешней среде. Единственным способом уменьшить время реакции является уменьшение времени одного скана, что достигается применением современных быстродействующих процессоров.

Вследствие наличия временной задержки в процессе работы ПЛК может возникнуть ситуация, когда какой-нибудь входной сигнал окажется слишком коротким и попадет в фазу цикла отработки программы, а в фазе опроса входов уже не будет активен. Следовательно ПЛК просто «не заметит» данный сигнал и это может привести к серьезной аварии. Поэтому

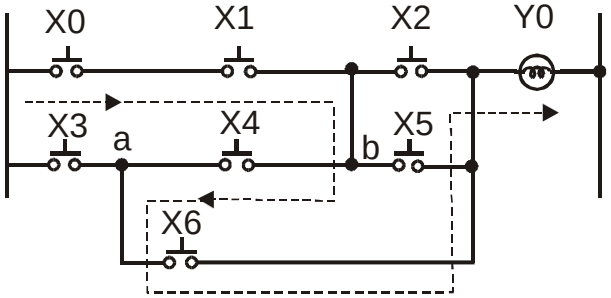
при проектировании систем автоматики необходимо выбирать модель контроллера с достаточным быстродействием.

Схематично работу контроллера можно представить следующим образом:



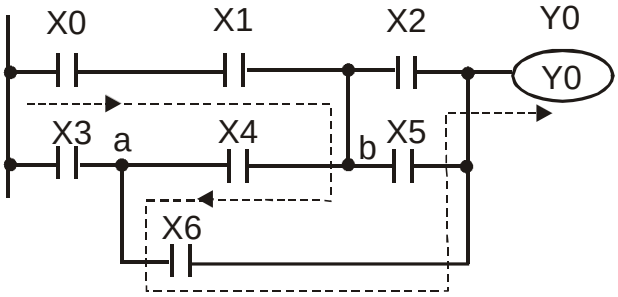
Второе важное отличие в работе релейно-контактных электросхем от ступенчатых диаграмм ПЛК заключается в том, что контроллер проходит программу исключительно слева на право и сверху вниз. Следовательно не допускаются никакие реверсивные направления при выполнение программы (обратное протекание тока по цепи). В обычных схемах допускаются любые направления исполнения схемы (протекание тока по элементам цепи). Данный момент иллюстрируются на рисунке ниже:

Обратное протекание тока в традиционной релейно-контактной схеме



Катушка Y0 замкнется

Обратное протекание тока в ступенчатой диаграмме ПЛК



Ошибка в 3 ряду. Катушка Y0 не замкнется
При компиляции будет выдана ошибка

Пояснения:

Если замкнуть контакты X0 , X1 , X4 и X6 , а остальные контакты будут разомкнуты, то возникнет «обратное» протекание тока по цепи: контакт X0 – контакт X1 – точка b – контакт X4 – контакт X6 и далее к катушке Y0.

Традиционная схема состоит из физически существующих приборов, поэтому протекание тока обуславливается только законами электротехники и возможно в любых направлениях.

В контроллерах все элементы ступенчатой диаграммы эмулируются процессором, который работает циклично только вперед и не может вернуться на шаг назад в рамках одного скана. Поэтому в случае возникновения ситуации, при которой процессору как бы придется «вернуться» назад в программе, в данном примере от точки b к контакту X6, приведет к сбою и будет выдана ошибка при компиляции.

1.3 Внутренние объекты контроллера (операнды)

Как было показано выше, процессор контроллера эмулирует функции различных приборов и эмитирует их работу. Все эмулируемые объекты, существующие только внутри процессора и памяти контроллера, называются операндами. Контроллер можно представить как определенный набор не существующих реально контактов, выходных катушек, реле, таймеров, счетчиков и т.д. Однако, при программировании их можно использовать как реальные приборы, или точнее сказать их функции. Каждая модель контроллера имеет свой ограниченный набор операндов, доступных для написания программы.

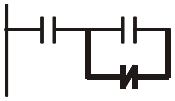
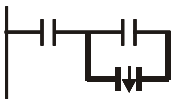
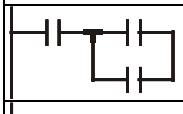
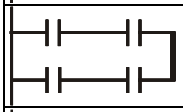
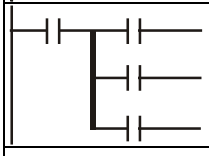
Любой операнд занимает определенный объем памяти контроллера – бит, байт (8 бит), слово (16 бит) или двойное слово (32 бит). Например для включения катушки необходимо записать «1» в соответствующую ячейку памяти, а чтобы выключить записать «0».

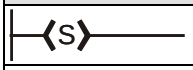
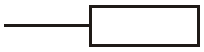
Для удобства создания и чтения ступенчатых диаграмм каждый операнд имеет свое стандартное название и обозначение. Данная мера позволяет использовать одну и туже схему разработчиком системы автоматизации, эксплуатационным инженером, электриком, монтажником, оператором технологической установки. Далее приводится описание всех типов операндов контроллера.

Тип и обозначение операнда		Описание
Вход	X	Входные реле. Определяют состояние внешних битовых устройств, подключенных к входным клеммам ПЛК. Могут принимать одно из двух состояний: 0 или 1. Адресация ведется в восьмеричной системе: X0, X1, … X7, X10, X11, …
Выход	Y	Выходные реле. Определяют состояние выходных клемм ПЛК, к которым подключается нагрузка. В программе могут быть как контактами, так и катушками, и принимать одно из двух состояний: 0 или 1. Адресация ведется в восьмеричной системе: Y0, Y1, … Y7, Y10, Y11, …
Маркер	M	Внутренние (вспомогательные) реле. Память для двоичных промежуточных результатов. В программе могут быть как контактами, так и катушками, и принимать одно из двух состояний: 0 или 1. Адресация ведется в десятичной системе: M0, M1, … M7, M8, M9, …

Состояние шага	S	Управляющие шаговые реле. Используются для программирования последовательного управляющего процесса. Могут принимать одно из двух состояний: 0 или 1. Адресация ведется в десятичной системе: S0, S1, … , S1023
Таймер	T	Реле времени. В программе могут использоваться для хранения текущего значения таймера и иметь 16-ти битный формат, а также могут быть контактами, и принимать одно из двух состояний: 0 или 1. Адресация ведется в десятичной системе: T0, T1, …, T255
Счетчик	C	Используются для реализации счета. В программе могут использоваться для хранения текущего значения счетчика и иметь 16-ти или 32-х битный формат, а также могут быть контактами, и принимать одно из двух состояний: 0 или 1. Адресация ведется в десятичной системе: C0, C1, …, C255
Десятичная константа	K	Определение числа в десятичной системе отсчета
Шестнадцатеричная константа	H	Определение числа в шестнадцатеричной системе отсчета
Регистр данных	D	Память данных. 16-ти или 32-х битный формат. Адресация ведется в десятичной системе: D0, D1, …, D9999. В 32-х битном формате один регистр занимает две ячейки, например при обращении к D10, данные будут прочитаны из ячеек D10 и D11.
Файловый регистр		Используются для хранения данных, когда не хватает регистров данных. Для чтения и записи необходимо использовать специальные инструкции MEMR и MEMW. Операнд не имеет своего символа, а адресация ведется с помощью десятичных констант: K0, K1, …, K9999.
Индексный регистр	E, F	Память данных для промежуточных результатов и индексной идентификации. 16-ти битный формат. Адресация: E0 – E7, F0 – F7
Указатель	P	Адрес для перехода к подпрограмме.
Указатель прерывания	I	Адрес обработки прерывания.
Номера вложенности	N	Используются для нумерации вложенных схем исключения. N0 – N7.

Условные обозначения элементов ступенчатой диаграммы ПЛК

Графический значок	Значение	Команда (инструкция)	Операнд
	Нормально-открытый контакт, тип «А»	LD	X, Y, M, S, T, C
	Нормально-закрытый контакт, тип «В»	LDI	X, Y, M, S, T, C
	Последовательный нормально-открытый контакт (логическое «И»)	AND	X, Y, M, S, T, C
	Параллельный нормально-открытый контакт (логическое «ИЛИ»)	OR	X, Y, M, S, T, C
	Параллельный нормально-закрытый контакт (логическое «ИЛИ-НЕ»)	ORI	X, Y, M, S, T, C
	Контакт, формирующий импульс по переднему фронту входного сигнала	LDP	X, Y, M, S, T, C
	Контакт, формирующий импульс по заднему фронту входного сигнала	LDF	X, Y, M, S, T, C
	Последовательный контакт, формирующий импульс по переднему фронту входного сигнала	ANDP	X, Y, M, S, T, C
	Последовательный контакт, формирующий импульс по заднему фронту входного сигнала	ANDF	X, Y, M, S, T, C
	Параллельный контакт, формирующий импульс по переднему фронту входного сигнала	ORP	X, Y, M, S, T, C
	Параллельный контакт, формирующий импульс по заднему фронту входного сигнала	ORF	X, Y, M, S, T, C
	Последовательный блок контактов	ANB	–
	Параллельный блок контактов	ORB	–
	Срабатывание нескольких выходов от одного входа	MPS MRD MPP	–
	Команда для активации выхода (катушки)	OUT	Y, M, S

Графический значок	Значение	Команда (инструкция)	Операнд
	Шаговое реле	STL	S
	Выбор команды или прикладной инструкции и установка ее параметров	Прикладная инструкция	См. описание команд и инструкций
	Логическая инверсия	INV	–

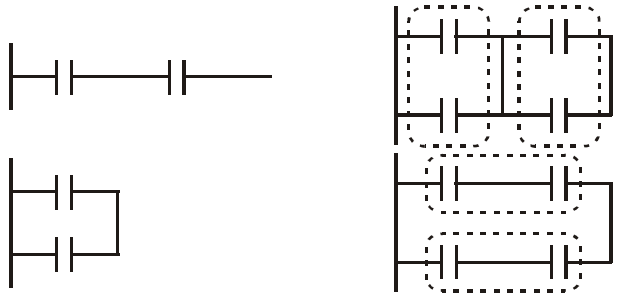
Примечание:
Графические значки элементов используются в языке ступенчатых диаграмм (англ. LD), а буквенное обозначение команд и инструкций (напр. LD, OR, STL и т.д.) используется в языке «список инструкций» (англ. IL).

Пояснения:
Блоком называется такая ступенчатая диаграмма, которая состоит из последовательно или параллельно объединенных групп взаимосвязанных контактов. Каждая группа является как бы единым объектом и внутри блока происходят логические операции именно с этими объектами, а не с отдельными контактами.
В целом блок обрабатывается в следующей последовательности: сначала обрабатывается состояние контактов внутри каждой группы, затем формируется выходной сигнал каждой группы, далее происходит логическая операция уже над данными группами, которые выступают в качестве объектов логической операции.

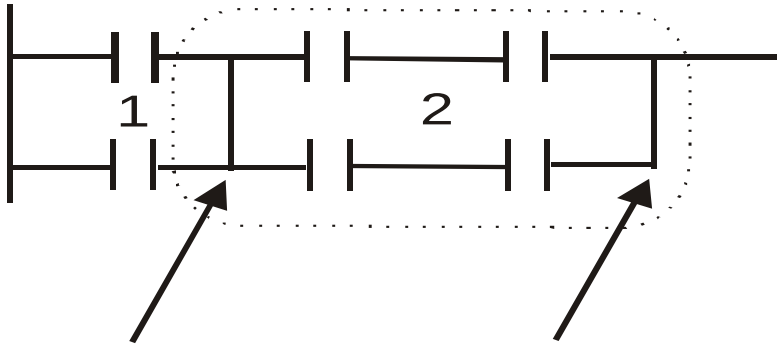
Наиболее распространенными операциями над блоками является логическое умножение и сложение.

Последовательный блок
Команда ANB
(логическое умножение)

Параллельный блок
Команда ORB
(логическое сложение)



Тип блока, параллельный или последовательный, определяется положением соединительной вертикальной линии. Слева от линии один блок, справа другой. В зависимости от схемы одна и та же линия может одну группу контактов объединять последовательно, а другую параллельно.

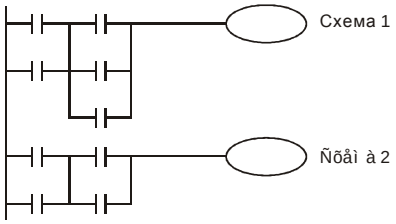


Линия 1 Линия 2

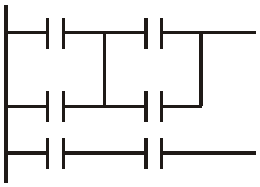
Линия 1. Для блока 1 выступает как параллельное объединение. Для блоков 1 и 2 выступает как последовательное объединение.
Линия 2. Для блока 2 выступает как параллельное объединение.

Контакты и блоки, которые объединены вертикальными и горизонтальными линиями в единую систему формируют одну независимую схему. Программа может содержать много независимых схем. Каждая схема должна заканчиваться каким-либо выходным операндом. В противном случае схема будет незаконченной и при компиляции будет выдана ошибка.

Две независимые схемы



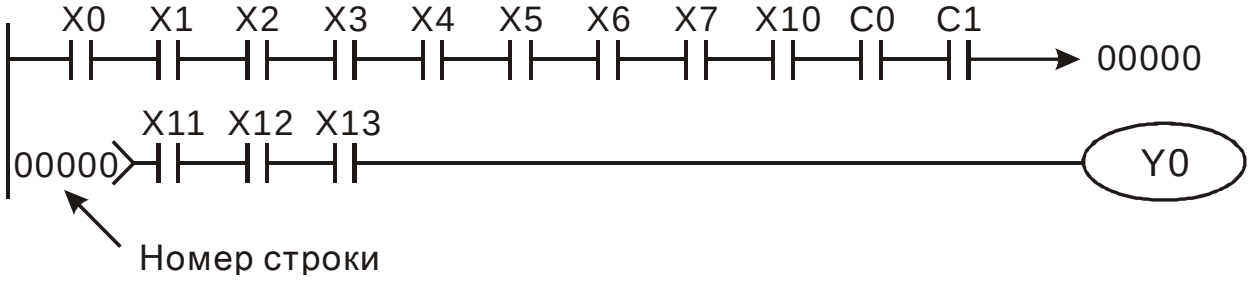
Незаконченная схема



1.4 Базовые правила написания ступенчатых диаграмм контроллеров

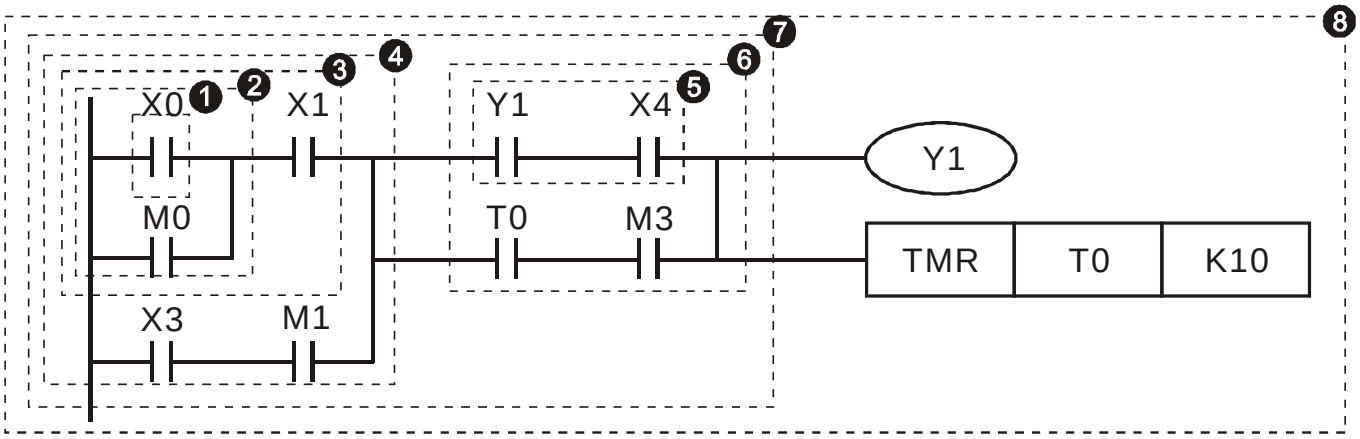
Написание программы необходимо осуществлять по строкам. Закончив одну строку переходите к следующей строке. Левая вертикальная линия олицетворяет собой общую силовую шину. В традиционных релейно-контактных схемах рисовалась и правая силовая шина, т.е. показывался путь тока через контакты и приборы. В ступенчатых диаграммах контроллеров правая шина не рисуется.

Максимальное количество контактов в одном ряду – 11. Если количество превышает 11, то контакты с порядковым номером больше 11 автоматически переносятся на новую строку, в начале которой будет стоять номер исходной строки, продолжением которой является новая строка. В конечном счете каждый ряд должен заканчиваться одним из выходных операндов (контакт, катушка, инструкция).



При написании программы необходимо учитывать, что контроллер сканирует программу слева на право сверху вниз и только вперед, т.е. из верхнего левого угла к правому нижнему. Программа обрабатывается по взаимосвязанным контактам одного фрагмента. Когда обработана одна связка контактов, контроллер делает шаг вперед и захватывает следующие контакты и т.д. пока не дойдет в данном фрагменте до выходного операнда, которым должен заканчиваться каждый фрагмент ступенчатой диаграммы. Потом контроллер переходит к следующему фрагменту.

Процесс сканирования программы проиллюстрирован ниже.



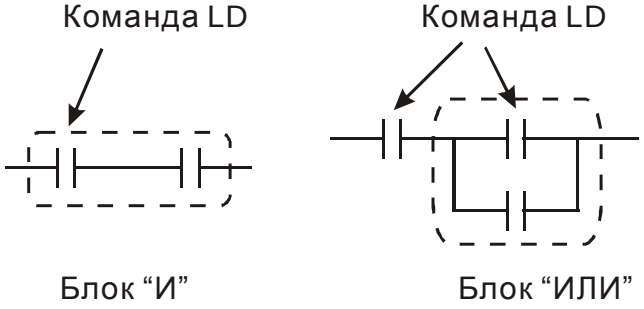
Последовательность отработки контроллером данного фрагмента программы следующая:

Шаг сканирования	Отрабатываемые команды	Контакты на ступенчатой диаграмме
1	LD	X0
2	OR	M0
3	AND	X1
4	LD	X3
	AND	M1
	ORB	
5	LD	Y1
	AND	X4
6	LD	T0
	AND	M3
	ORB	
7	ANB	
8	OUT	Y1
	TMR	T0 K10

Как видно из примера, один шаг при сканировании соответствует участку до ближайшего контакта или вертикальной линии. Обработка осуществляется по блокам, начиная с левого. Состояние выходных операндов изменяется в последнем шаге.

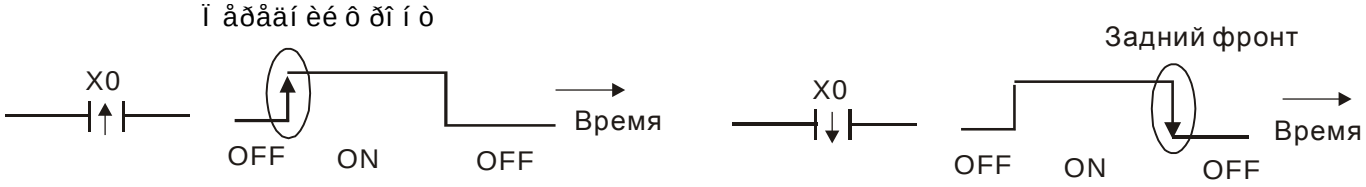
1.5 Разъяснение базовых элементов и структур ступенчатой диаграммы

1. Команды LD и LDI (нормально открытый и закрытый контакты).

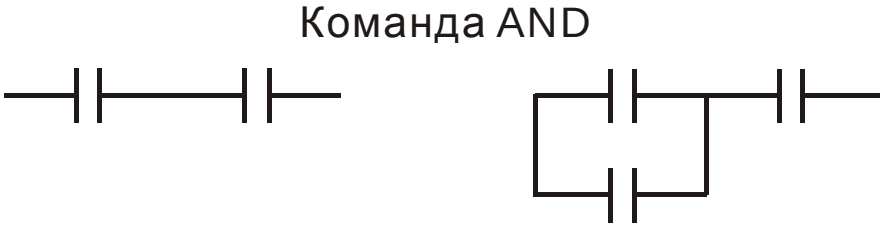


Вводят ряд или блок.
Активируются как от физических входов, так и от операндов контроллера.

Команды LDP и LDF работают аналогично команде LD. Отличие заключается в том, что при активном входном физическом контакте, команда LD формирует постоянный сигнал, а команда LDP формирует импульс по переднему фронту входного сигнала, а команда LDF по заднему фронту. Далее команды LDP и LDF до пропадания и следующего появления входного сигнала будут неактивны.

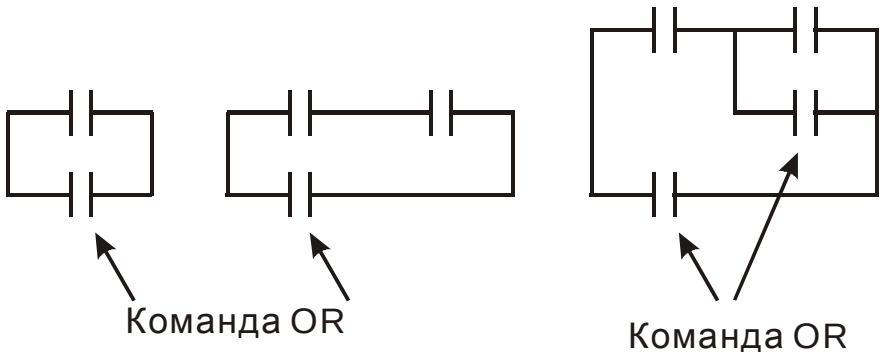


2. Команды AND («И») и ANI («И–НЕ»). Последовательно подсоединяют контакт к контакту или к блоку.



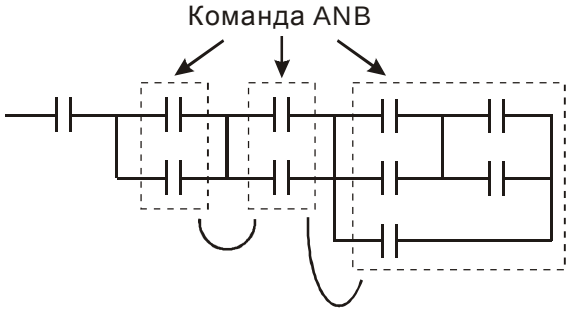
Команды ANDP и ANDF работают аналогично, но по переднему или заднему фронту соответственно.

3. Команды OR («ИЛИ») и ORI («ИЛИ–НЕ»). Параллельно подсоединяют контакт к контакту или к блоку.

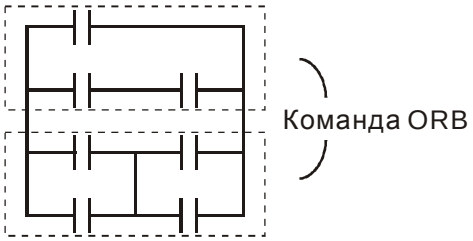


Команды ORP и ORF работают аналогично, но по переднему или заднему фронту соответственно.

4. Команда ANB. Последовательно объединяет блок с контактом или другим блоком.



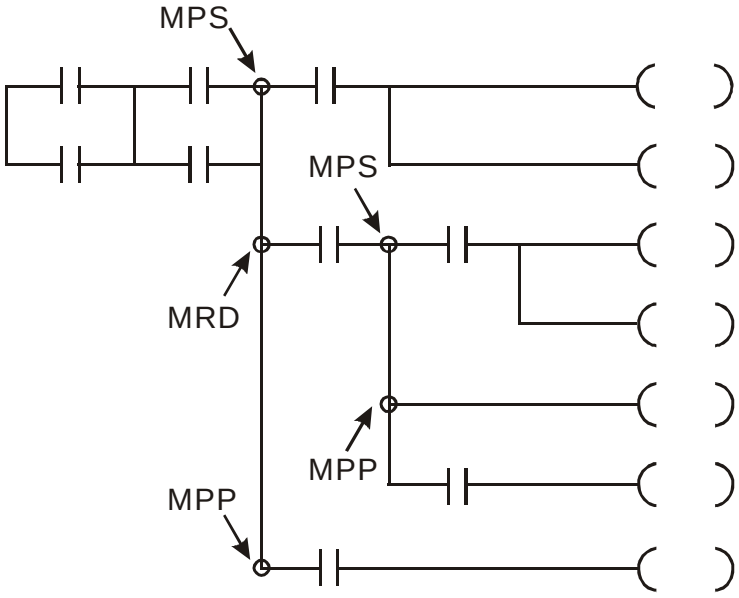
5. Команда ORB. Параллельно объединяет блок с контактом или другим блоком.



Примечание:
Для правильной работы команд ANB и ORB с несколькими блоками, они должны быть сгруппированы сверху вниз или слева на право.

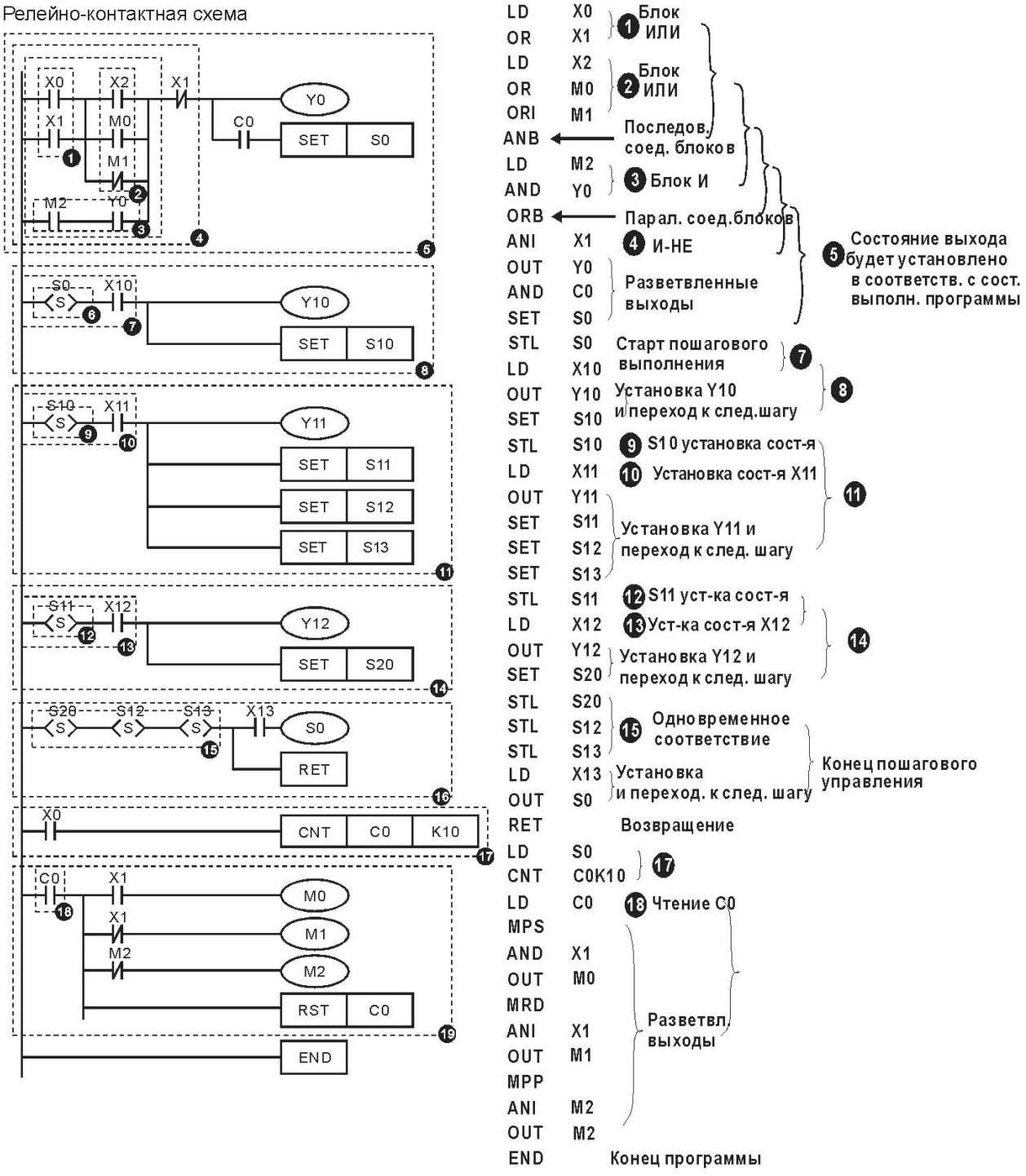
6. Команды MPS, MRD, MPP. Организуют срабатывание нескольких выходов от одного входа.
Команда MPS обозначает начало разветвления, т.е. точки соединения горизонтальной и вертикальной линии. С программной точки зрения это представляет из себя точку на схеме, обладающую памятью о состоянии контакта, находящегося перед ней. Благодаря этому, состояние одного входа можно передать сразу нескольким выходам. Может использоваться последовательно до 8 раз. В ступенчатой диаграмме можно узнать по символу «┬».
Команда MRD считывает память точки разветвления MPS и передает следующему за собой по горизонтальной прямой контакту. В ступенчатой диаграмме можно узнать по символу «├». Т.е. это точка соединения вертикальной линии, идущей от точки MPS, с горизонтальной линией, ведущей к выходному (промежуточному) контакту (команде или инструкции).
Команда MPP заканчивает ответвления от вертикальной линии, идущей от точки MPS, заканчивая тем самым блок разветвления. В ступенчатой диаграмме можно узнать по символу «└».

Важное замечание:
Как правило входные контакты все позволяют делать после себя разветвления. Однако, изредка компилятор может не присоединить какой-нибудь выход. Данная ситуация продемонстрирована на рисунке ниже. Это можно выявить при отладке программы.



1.6 Преобразование ступенчатых диаграмм в мнемокод

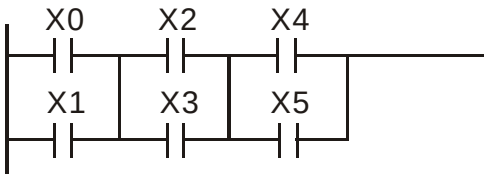
При компиляции ступенчатая диаграмма преобразуется в программу, представляющую из себя столбец команд и инструкций, а затем в мнемокод, который и передается в ПЛК.
Ниже показана последовательность преобразования ступенчатой диаграммы в последовательность команд и инструкций.



Процесс обработки релейно-контактной схемы идет с верхнего левого угла и заканчивается в правом нижнем, однако могут быть различные варианты преобразования в мнемокод, как показано в следующих примерах:

Пример 1. Ниже приведенную схему можно кодировать двумя различными методами, однако результат будет одинаковым.

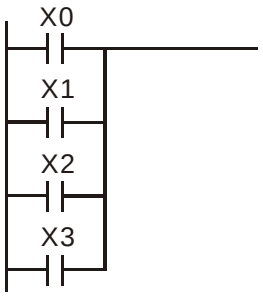
Первый метод кодирования является наиболее предпочтительным.



Хороший метод (1)		Плохой метод (2)	
LD	X0	LD	X0
OR	X1	OR	X1
LD	X2	LD	X2
OR	X3	OR	X3
ANB		LD	X4
LD	X4	OR	X5
OR	X5	ANB	
ANB		ANB	

Первый метод лучше второго ввиду особенностей работы процессора, для которого нежелательно использовать подряд такие команды как ANB. Первым методом можно объединять неограниченное количество блоков, а вторым максимум 8.

Пример 2. Различное кодирование параллельно соединенных контактов.

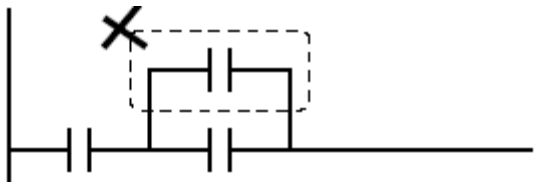


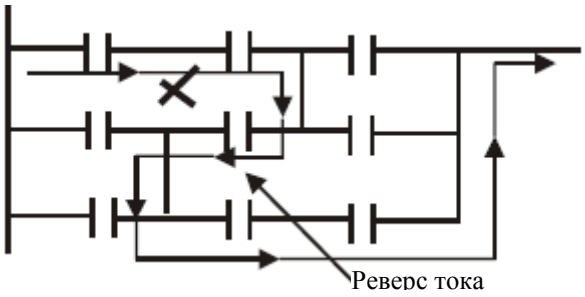
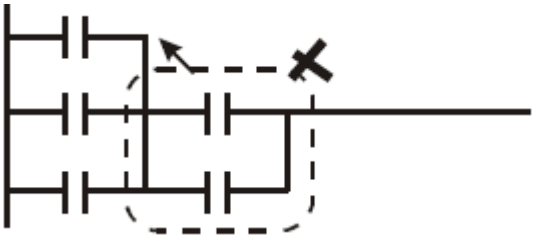
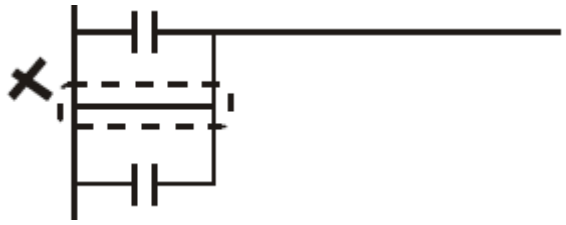
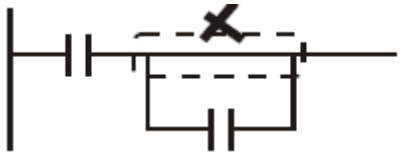
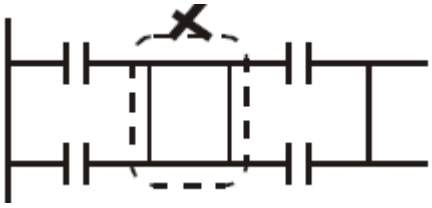
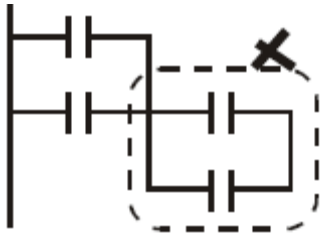
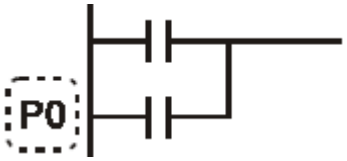
Хороший метод (1)		Плохой метод (2)	
LD	X0	LD	X0
OR	X1	LD	X1
OR	X2	LD	X2
OR	X3	LD	X3
		ORB	
		ORB	
		ORB	

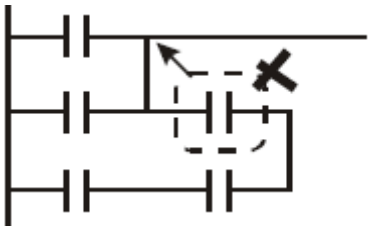
Преимущество первого метода очевидно, так как меньше задействуется памяти. Во втором методе программа существенно длиннее.

Типичные ошибки при написании ступенчатых диаграмм

Ввиду особенностей работы процессора при написании ступенчатых диаграмм необходимо придерживаться определенных правил. Ниже приведены типичные ошибки, которые вызывают сбои в работе контроллера.

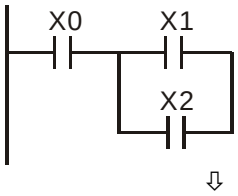
	Нельзя подключать параллельный контакт сверху.

	"Ток" через контакты должен протекать только слева на право. Реверсивное направление не допускается.
	Командная линия должна располагаться выше.
	Блок ИЛИ должен быть расположен выше
	Нельзя выполнить операцию параллельно пустой линии.
	Нельзя выполнить операцию параллельно пустой линии.
	В среднем блоке отсутствуют объекты.
	Блоки должны располагаться на одном уровне.
	Указатель должен быть расположен напротив верхнего устройства командной строки.

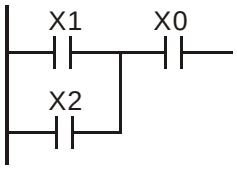
	Блок, который соединен последовательно, должен располагаться параллельно на верхней командной линии.

1.7 Оптимизация ступенчатых диаграмм

- Если поставить блок вначале командной строки, то можно избежать применения команды ANB для присоединения контакта или блока.

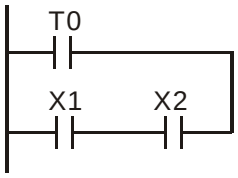


	Команды
LD	X0
LD	X1
OR	X2
ANB	

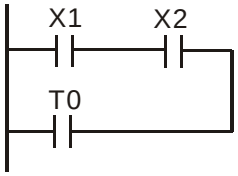


	Команды
LD	X1
OR	X2
AND	X0

- Если поставить последовательные контакты выше параллельного контакта, то можно избежать применения команды ORB.

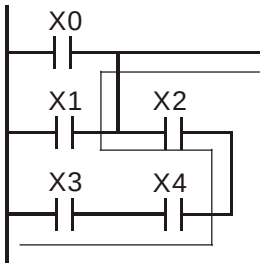


	Команды
LD	T0
LD	X1
AND	X2
ORB	



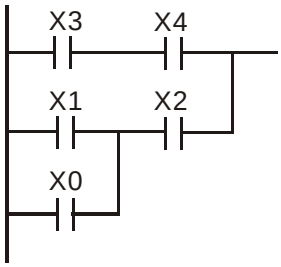
	Команды
LD	X1
AND	X2
OR	T0

- На следующих рисунках иллюстрируется пример, где показан метод, позволяющий избежать реверсивное течение тока. На верхнем рисунке показана неправильная компоновка ступенчатой диаграммы, которая приводит к «обратному шагу» при прохождении программы. В данном примере верхняя часть короче схемы, поэтому ее можно перенести вниз и тем самым избежать реверсивного направления тока.



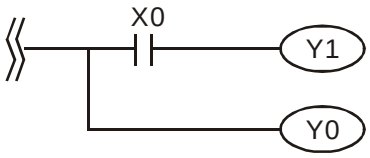
Команды	
LD	X0
OR	X1
AND	X2
LD	X3
AND	X4
ORB	

⇩



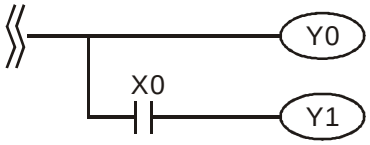
Команды	
LD	X3
AND	X4
LD	X1
OR	X0
AND	X2
ORB	

- Можно избежать использования команд MPS и MPP, если выходной контакт на той же горизонтальной линии, что и общий входной контакт, не имеет больше условий срабатывания.



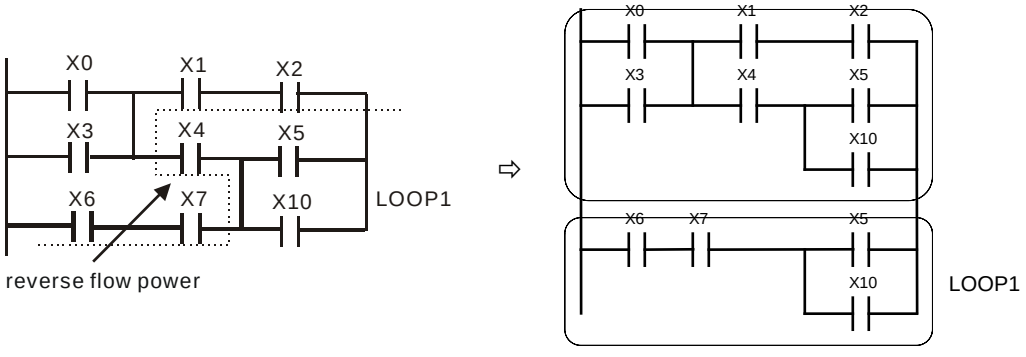
Команды	
MPS	
AND	X0
OUT	Y1
MPP	
OUT	Y0

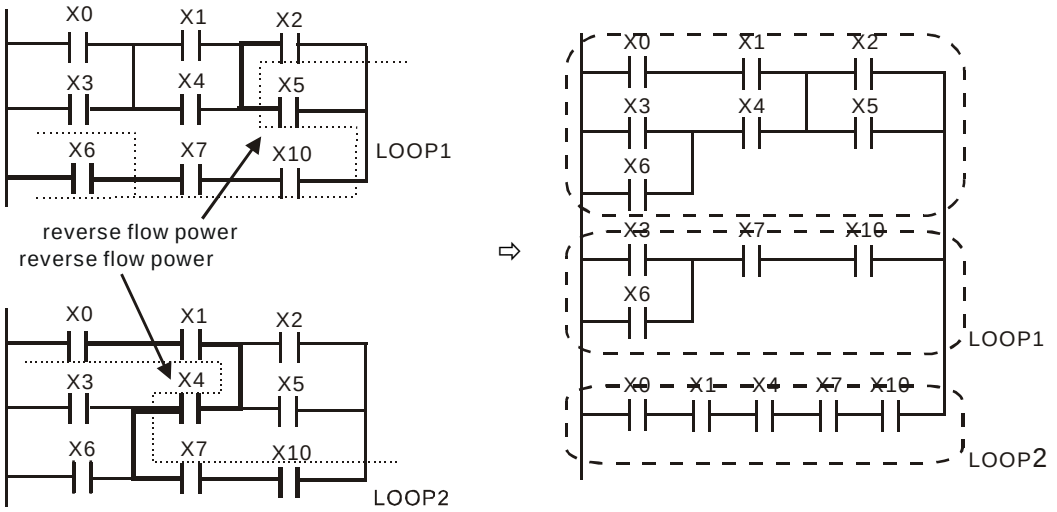
⇩



Команды	
OUT	Y0
AND	X0
OUT	Y1

- Примеры исправления ступенчатых диаграмм с «обратным течением тока». На рисунках слева приведены ошибочные варианты, а справа исправленные аналоги, сохраняющие логику исходной диаграммы.



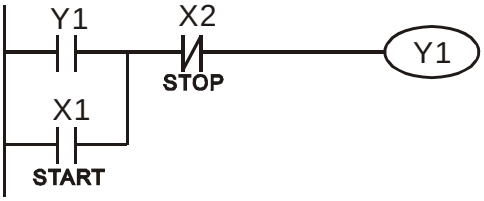


1.8 Примеры реализации в ступенчатых диаграммах часто встречающихся задач

Очень часто в системах автоматики необходимо организовать пуск и стоп с помощью кнопок без фиксации, т.е. организовать самоблокировку выхода. Примеры реализации подобной задачи приведены ниже.

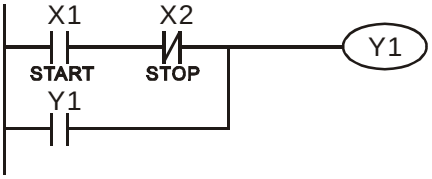
Пример 1. Самоблокировка выхода с приоритетом Стопа

При нажатии кнопки X1=On, сигнал проходит через нормально замкнутую кнопку X2 и вызывает замыкание катушки Y1. При этом замыкается связанный входной контакт Y1. При нажатии кнопки X2 (Стоп) цепь разомкнется и катушка (выход) Y1 отключится. Поэтому данную схему называют приоритетом Стопа.



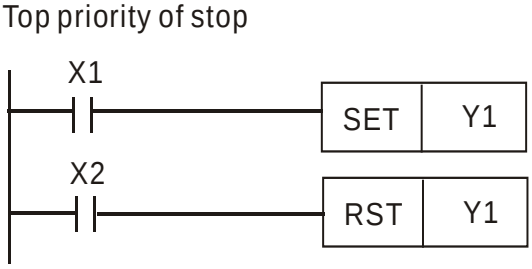
Пример 2. Самоблокировка выхода с приоритетом Старта

При нажатии кнопки X1=On, сигнал проходит через нормально замкнутую кнопку X2 и вызывает замыкание катушки Y1. При этом замыкается связанный входной контакт Y1. При нажатии кнопки X2 (Стоп) цепь не разомкнется и катушка (выход) Y1 останется включенной. Поэтому данную схему называют приоритетом Старта.

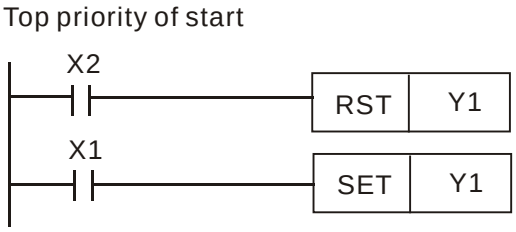


Пример 3. Самоблокировка выхода с использованием команд SET и RST

Если команда RST следует за командой SET, то данная цепь называется с приоритетом Стопа. Программа выполняется сверху вниз, поэтому при одновременном замыкании контактов X1 и X2 катушка (выход) Y1 сначала замкнется, а затем разомкнется, т.е. состояние выхода в конечном итоге определяется кнопкой Стоп.

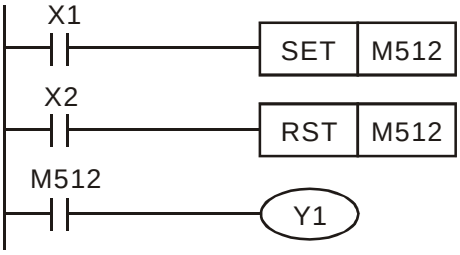


Если команда RST стоит перед командой SET, то данная цепь называется с приоритетом Старта. Программа выполняется сверху вниз, поэтому при одновременном замыкании контактов X1 и X2 катушка (выход) Y1 сначала разомкнется, а затем снова замкнется, т.е. состояние выхода в конечном итоге определяется кнопкой Старт.



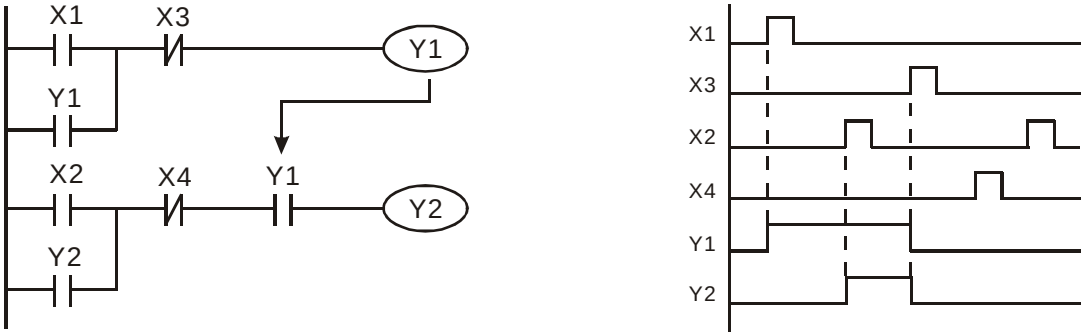
Пример 4. Самоблокировка выхода с использованием команд SET и RST и энергонезависимого промежуточного реле M512

Команды SET и RST замыкают и размыкают реле M512. Так как оно энергонезависимое, то при пропадании и последующем восстановлении напряжения питания катушка (выход) Y1 останется замкнутой, т.е. самоблокировка выхода сохранится.



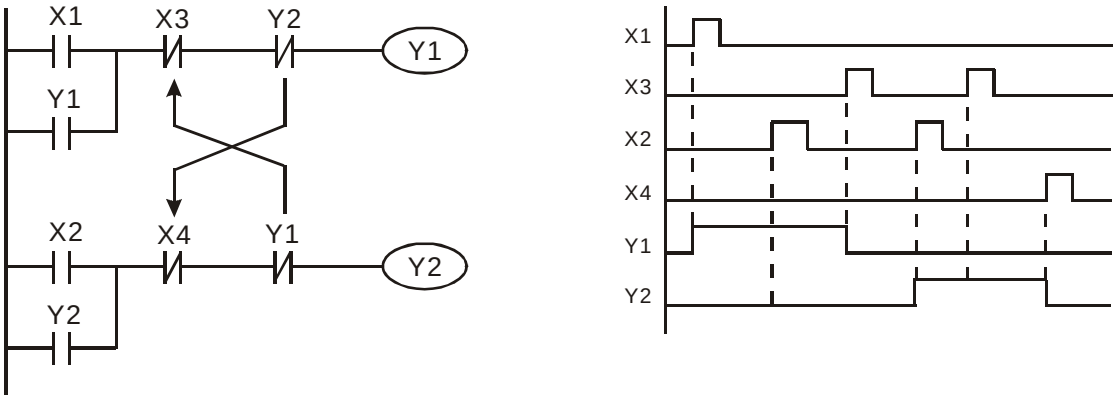
Далее приводятся типовые схемы, реализующие различные варианты управления выходами.

Пример 5. Схема с включением выхода при условии включения другого выхода



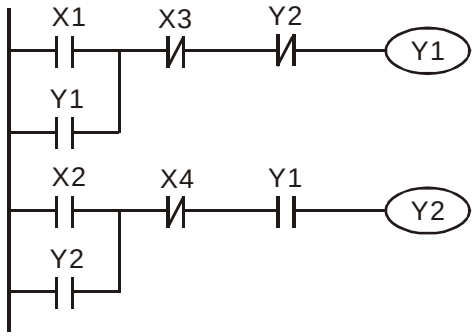
Контакты X1 и X3 самостоятельно включают и выключают выход Y1. Контакты X2 и X4 могут включать и выключать выход Y2 только при условии, что выход Y1 замкнут, так как он включен последовательно (логическое «И») в цепь выхода Y2.

Пример 6. Схема со взаимоблокировкой выходов



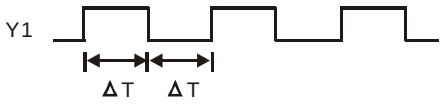
Контакты X1 и X2 включают выходы Y1 и Y2 соответственно. Однако, выходы Y1 и Y2 не могут быть активны одновременно, так как включены последовательно в цепь друг друга. Когда Y1 включен, Y2 не может быть включен, так как взаимосвязанный с выходом Y1 контакт будет разомкнут и наоборот.
Даже если X1 и X2 сработают одновременно, выходы Y1 и Y2 не включатся сразу оба ввиду прохождения программы сверху вниз. В данной схеме выход Y1 будет иметь приоритет над выходом Y2 (т.е. Y1 включится, а Y2 нет).

Пример 7. Последовательное включение выходов



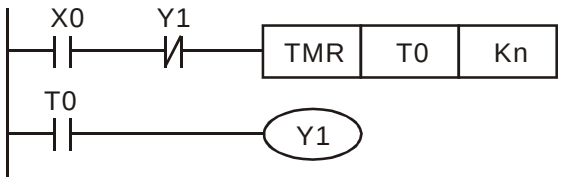
Данная схема реализует последовательное включение выходов.
Сначала должен включится Y1, и только после этого сможет включится выход Y2, но при этом разомкнется выход Y1.

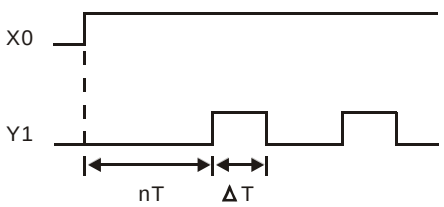
Пример 8. Колебательные схемы



Колебательные схемы используют одну из особенностей работы контроллеров, а именно циклическое сканирование программы с единственным чтением входов и обновлением выходов в рамках одного скана (цикла).
На приведенном выше примере при первом скане выход неактивен, следовательно нормально закрытый контакт Y1 также неактивен и пропустит через себя сигнал и катушка (выход) Y1 замкнется и на физическом выходе контроллера появится сигнал. Данное состояние останется до следующего скана.
При следующем скане контакт Y1 уже разомкнут, следовательно процессор переведет выход Y1 в выключенное состояние. На физическом выходе контроллера сигнал пропадет, а взаимосвязанный с выходом Y1 контакт Y1 снова будет неактивен, т.е. замкнется.
При следующем скане выход Y1 замыкается, на физическом выходе контроллера появляется сигнал, а при следующем скане выход снова размыкается и сигнал пропадает.
Таким образом, на выходе контроллера формируется последовательность прямоугольных импульсов, частота следования которых определяется длительностью одного скана (цикла) прохождения программы. Один цикл (ΔT) сигнал на выходе есть и один цикл нет (см. диаграмму выше ΔT(On)+ΔT(Off)).

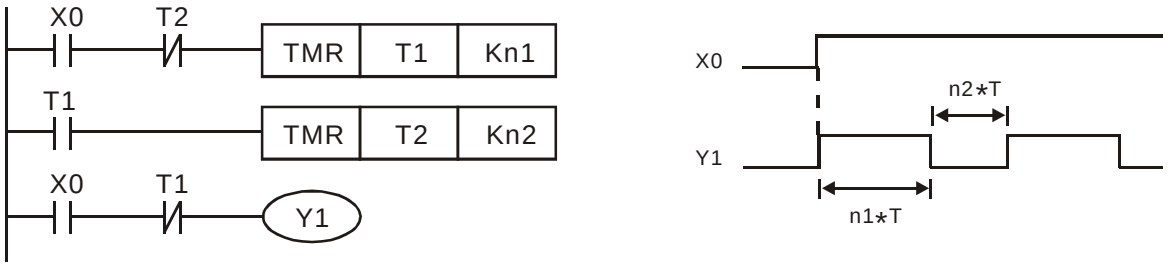
На следующем примере приведена схема с использованием таймера для изменения промежутка между импульсами.





В начальный момент времени контакт T0 и соответственно выход Y1 разомкнуты, а взаимосвязанный нормально закрытый контакт Y1 замкнут. При замыкании входного контакта X0 начнется отсчет уставки времени таймера T0, которая задается в десятичном формате (Kn). После отсчета заданного времени контакт T0 и соответственно выход Y1 замкнуться, а на физическом выходе контроллера появится сигнал. Данное состояние останется до конца текущего скана.
При следующем скане (контакт X0 замкнут) контакт Y1 разомкнется, так как выход Y1 замкнут. Контакт T0 разомкнется сам и разомкнет катушку (выход) Y1.
На следующем скане (контакт X0 замкнут) снова начнется отсчет уставки таймера и процедура повторится. Таким образом, в данной схеме длинна импульса будет равна времени скана, а промежуток между импульсами будет равен времени уставки таймера.
На временной диаграмме сверху «T» – это шаг таймера, «n» - количество шагов, отсчитываемых таймером (вместе это уставка), а ΔT – время одного скана.

Пример 9. Схема для организации повторно-прерывистого свечения лампочек или звучания зуммера

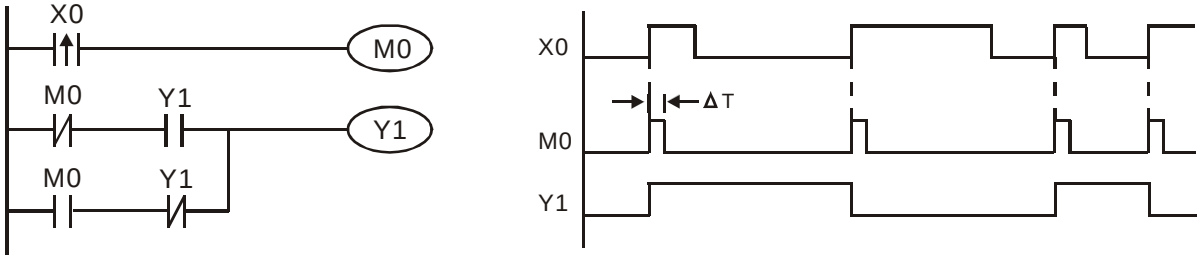


При помощи вышеприведенной схемы можно организовать управление длительностью периодов свечения лампочек или звучания зуммера. С этой целью используется два таймера, которые управляют одним выходом Y1.
При замыкании контакта X0 сигнал проходит через нормально закрытые контакты T2 и T1, замыкая выход Y1 (лампочка зажглась или зуммер начал звучать), а также начинается отсчет уставки времени таймера T1.
По достижении уставки Kn1 замыкается нормально открытый контакт T1, начинается отсчет уставки времени таймера T2, размыкается нормально закрытый контакт T1, который размыкает выход Y1 (лампочка перестает гореть или зуммер звучать).
По достижении уставки таймера T2 размыкается нормально закрытый контакт T2 и соответственно сбрасывается таймер T1, замыкается нормально закрытый контакт T1, который замыкает выход Y1 (лампочка снова начинает гореть или зуммер звучать).
Данный цикл будет продолжаться до тех пор пока замкнут контакт X0. Уставка таймера T1 определяет продолжительность горения лампочки (звучания зуммера), а уставка таймера T2 определяет время отсутствия свечения лампочки (звучания зуммера).
На временной диаграмме сверху «T» – это шаг таймера, «n1» - количество шагов, отсчитываемых таймером T1 (вместе это уставка), «n2» - количество шагов, отсчитываемых таймером T2.

Пример 10. Триггерная схема

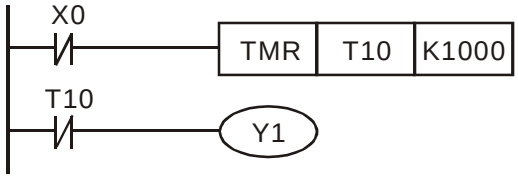
В ряде случаев бывает необходимо сигналом на входе, который может быть различной длительности, перебрасывать выход из состояния «выключен» в состояние «включен» и наоборот. Данные схемы моделируют работу триггера, который имеет два устойчивых состояния.

Важной особенностью данной схемы является то, что выход будет находится в заданном устойчивом состоянии независимо от длительности одного непрерывного входного сигнала, и изменит свое состояние только при пропадании предыдущего и появлении нового входного сигнала любой длительности.

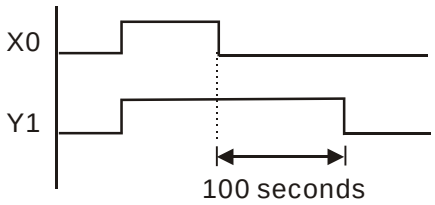


Передний фронт сигнала с X0 замкнет на время одного скана вспомогательное реле M0. При этом разомкнется нормально закрытый контакт M0, а нормально открытый контакт M0 наоборот замкнется и активирует выход Y1.
При следующем скане реле M0 будет уже разомкнуто, следовательно нормально открытый контакт M0 разомкнется, а нормально закрытый контакт M0 снова замкнется. Так как связанный контакт Y1 будет в данный момент замкнут, то выход Y1 останется активным.
При появлении следующего сигнала на входе X0 промежуточное реле M0 снова замкнется на время одного скана. В этот раз сигнал не пройдет на катушку Y1, так как в одной линии контакт M0 будет замкнут, а контакт Y1 разомкнут. В другой линии наоборот. Следовательно по результату скана выход Y1 сбросится (разомкнется).
При появлении нового сигнала на входе X0 процерура повторится, выход Y1 замкнется и т.д.
Данная схема еще называется импульсным реле.
Графически работа данной схемы поясняется на временной диаграмме сверху.

Пример 11. Схема задержки отключения выхода

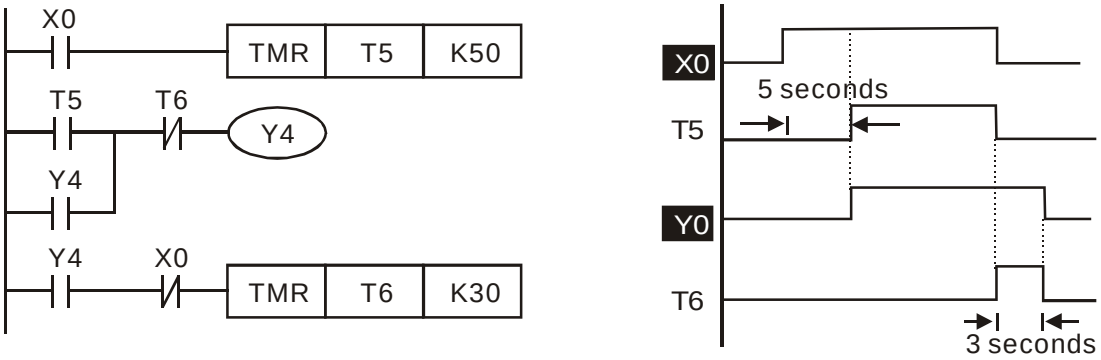


шаг таймера = 0,1 сек



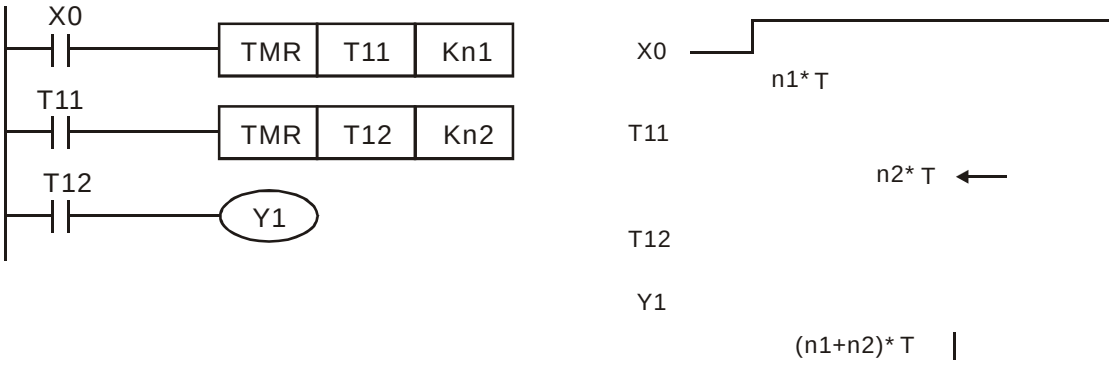
Исходным состоянием будем считать, когда нормально закрытый контакт X0 неактивен, т.е. замкнут, а таймер T10 отсчитал свою уставку и разомкнул свой контакт T10, отключив тем самым катушку (выход) Y1.
Когда на X0 подается сигнал, контакт размыкается, сбрасывая тем самым таймер T10, который сбрасывает свой контакт T10, включая тем самым выход Y1. После снятия сигнала с контакта X0, он обратно замыкается и таймер T10 начинает отсчет своей уставки. По достижению установленного времени таймер активирует свой контакт T10 (размыкает его) и выход Y1 отключается.
В данном примере шаг таймера 0,1 сек., значение шагов 1000, следовательно уставка получается K1000*0.1 сек = 100 сек
Графически данный пример иллюстрируется на временной диаграмме сверху.

Пример 12. Схема задержки включения и отключения выхода



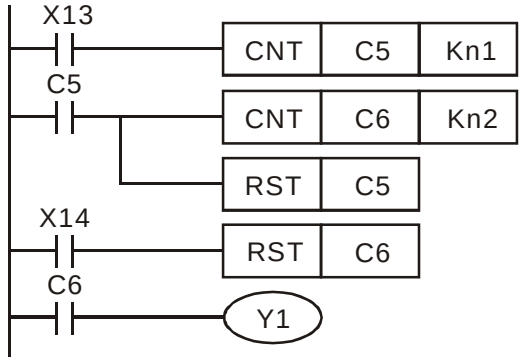
Будем считать исходным состоянием, когда X0 разомкнут, все таймеры отсчитали свои уставки и выход Y4 соответственно отключен.
При подаче сигнала замыкается контакт X0, таймер T5 начинает отсчет своей уставки и по ее достижении замыкает свой контакт T5, который включает самоблокирующийся выход Y4. Входной сигнал на X20 размыкает также нормально закрытый контакт X20 в линии таймера T6, блокируя тем самым возможность одновременной работы таймеров T5 и T6.
При активации катушки Y4 замыкается контакт Y4 на линии таймера T6, который начинает отсчет своей уставки при пропадании входного сигнала на нормально закрытом контакте X0, т.е. его замыкании.
При достижении значения уставки таймер T6 активирует свой контакт, что приведет к размыканию нормально закрытого контакта T6 на линии выхода Y4 и его сброса. Соответственно размыкаются все контакты Y4 и таймер T6 сбрасывается. Схема переходит в исходное состояние.
Таким образом, таймер T5 организует задержку включения выхода Y4 (уставка 5 сек = шаг 0,1 сек * 50 шагов), а таймер T6 определяет время задержки отключения выхода Y4 (уставка 3 сек = шаг 0,1 сек * 30 шагов).
Графически данный пример иллюстрируется на временной диаграмме сверху.

Пример 13. Схема расширения диапазона таймера



Уставка таймера ограничена определенным значением. Если необходима более длительная задержка включения выхода, то можно соединить последовательно два и более таймеров. На указанном примере условием включения таймера T12 является замыкание контакта таймера T11. Таким образом, выход Y1 замкнется после отсчета уставок двух последовательно объединенных таймеров T11 и T12.
Графически данный пример иллюстрируется на временной диаграмме сверху.

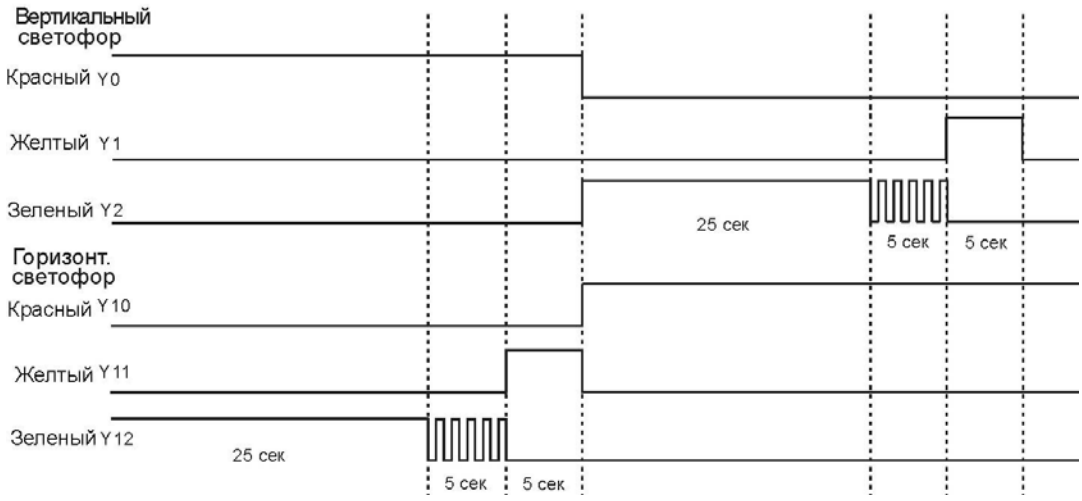
Пример 14. Схема расширения диапазона счетчика



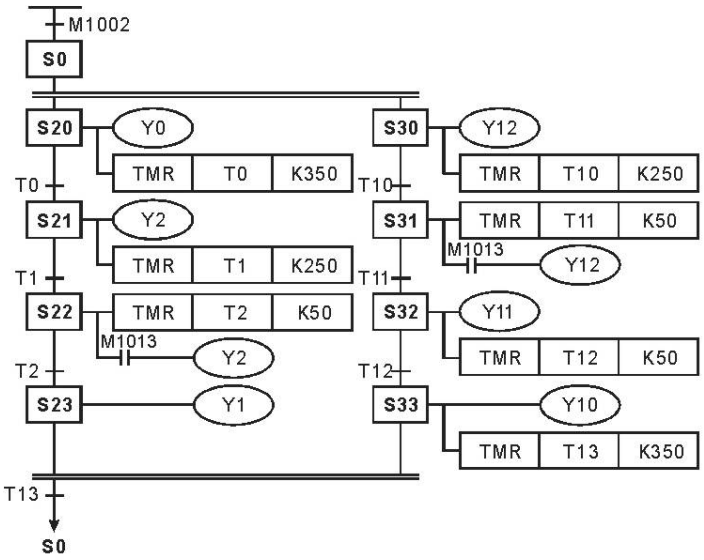
Диапазон 16-ти разрядного счетчика составляет 0~32767. Если необходимо считать большее количество входных импульсов, то можно объединить два счетчика как показано на схеме.
Входные импульсы на контакте X13 подсчитываются счетчиком C5. При достижении своей уставки счетчик замыкает свой контакт C5, который успевает выдать импульс на счетчик C6 и сбрасывает сам себя командой RST.
Далее счетчик C5 начинает снова считать импульсы от X13 и цикл повторяется до тех пор, пока счетчик C6 не достигнет своей уставки и замкнет выход Y1. Сброс счетчика C6 осуществляется контактом X14.
Таким образом, суммарное количество импульсов для срабатывания выхода Y1 будет произведением уставок двух счетчиков C5 и C6 (Kn1*Kn2). Максимальное количество импульсов при объединении двух 16-ти битных счетчиков становится:
32767*32767 = 1 073 676 289

Пример 15. Использование шаговых реле для организации циклограммы работы светофора

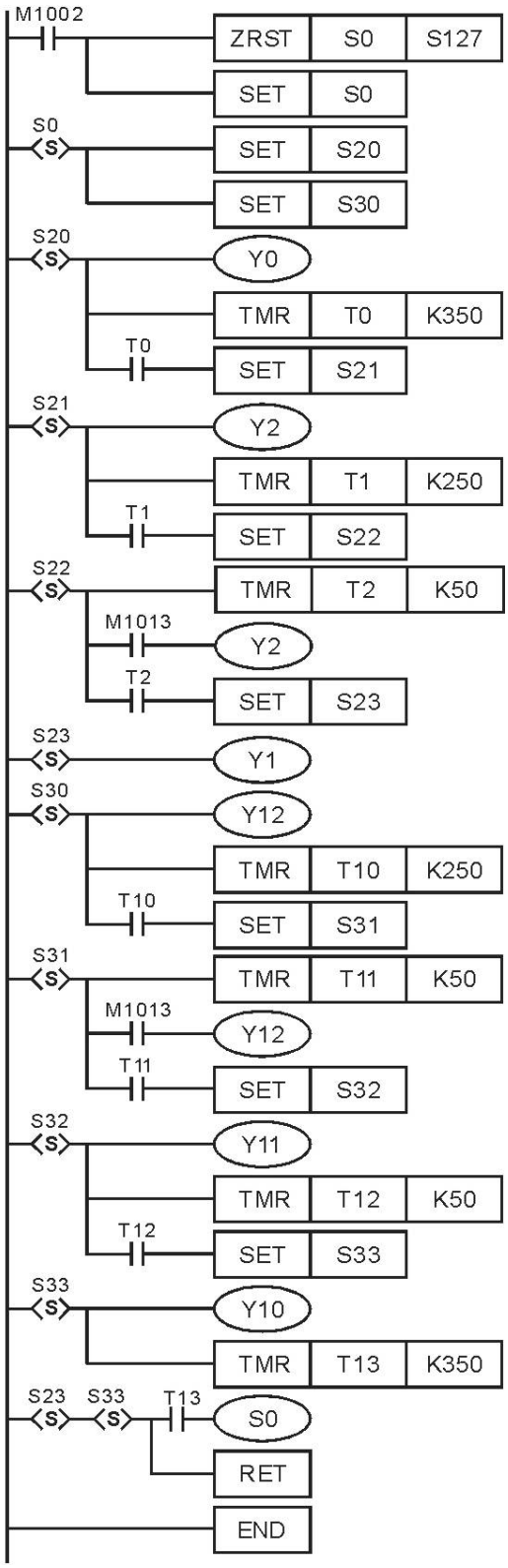
	Красный	Желтый	Зеленый	Зеленый мигает
Вертикальн. светофор	Y0	Y1	Y2	Y2
Горизонт. светофор	Y10	Y11	Y12	Y12
Время	35 сек	5 сек	25 сек	5 сек



SFC диаграмма:

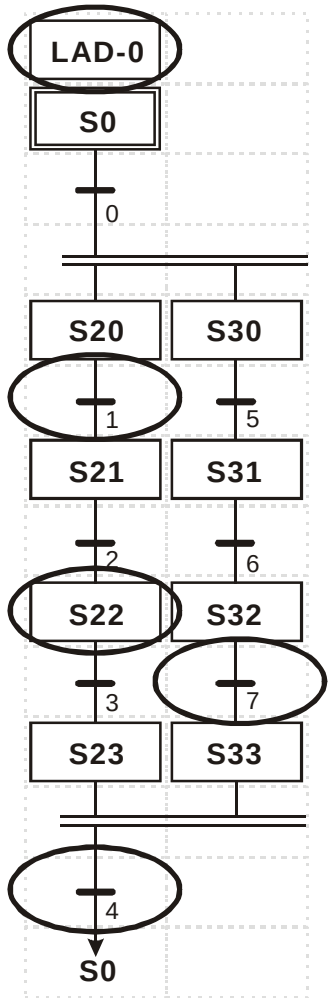
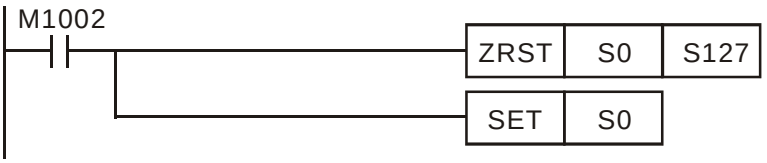
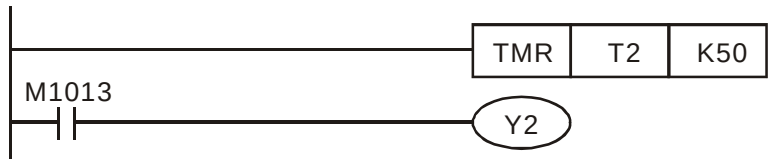


Релейно-контактная схема:



1.9 Пример написания программы на языке SFC

Программный продукт WPLSoft поддерживает язык SFC, который можно перевести как Последовательные функциональные блоки. Нижеприведенный пример показывает соответствие функциональных блоков фрагментам ступенчатых диаграмм.

SFC	Вид блоков на языке ступенчатых диаграмм
	■LAD-0  ■Условный переход 1  ■S22  ■Условный переход 4  ■Условный переход 7 

ГЛАВА 2

Назначение и описание операндов контроллеров Delta DVP

2.1 Общий обзор операндов контроллеров Delta DVP

Перечень доступных операндов для контроллеров типов ES/EX/SS

Элемент				Спецификация		Примечание
Битовые операнды (реле)	X	Физические входы		X0 ~ X177, 128 точек, восьмеричная система	Всего 256 точек	Соответствуют внешним точкам ввода/вывода
	Y	Физические выходы		Y0 ~ Y177, 128 точек, восьмеричная система		
	M	Вспомогательные реле (маркеры)	Общие Энергонезав. Специальные	M0 ~ M511, M768 ~ M999, 744 т. M512 ~ M767, 256 точек M1000 ~ M1279, 280 точек	Всего 1280 точек	Используются в программе как промежуточные реле
	T	Таймер	100 мс 10 мс (M1028=ON) 1 мс	T0 ~ T63, 64 точки T64 ~ T126, 63 точки (если M1028=OFF, то 100 мс) T127, 1 точка	Всего 128 точек	Инициируется инструкцией TMR. Когда отсчет времени достигнет уставки, то замкнется контакт «T» с соответствующим номером
	C	Счетчик	16-бит счет вверх общие 16-бит счет вверх энергонезав. 32-бит высокоскор. счет вверх/вниз энергонезав.	C0 ~ C111, 112 точек C112 ~ C127, 16 точек C235 ~ C238, C241, C242, C244 , 1 фаза 1 вход, 7 точек C246, C247, C249, 1 фаза 2 входа, 3 точки C251, C252, C254, 2 фазы 2 входа, 3 точки	128 точек 13 точек	Инициируется инструкцией CNT (DCNT). Когда счет достигнет уставки, то замкнется контакт «C» с соответствующим номером
	S	Шаговое реле	Инициализир. Возвращение в нулев. точку Энергонезав.	S0 ~ S9, 10 точек S10 ~ S19, 10 точек (*2), исп. с инструкцией IST S20 ~ S127, 108 точек	Всего 128 точек	

Словные операнды	**T**	Текущее значение таймера		T0 ~ T127, 128 точек		
	C	Текущее значение счетчика		C0 ~ C127, 16 бит, 128 точек C235 ~ C254, 32 бит, 13 точек		
	D	Регистры данных	Общие Энергонезав. Специальные Индексные	D0 ~ D407, 408 точек D408 ~ D599, 192 точки D1000 ~ D1311, 312 точек E, F – 2 точки	Всего 912 точек	Область для хранения данных. E и F для косвенной индексации
Индексы	**N**	Для мастер-контроля		N0 ~ N7, 8 точек		
	P	Для инструкций CJ, CALL		P0 ~ P63, 64 точки		
	I	Для прерываний	Внешние Временные Коммуникац.	I001, I101, I201, I301, 4 точки I6xx, (xx = 1~99), шаг 1 мс I150, 1 точка		Позиционный указатель для подпрограммы прерывания
Константы	**K**	Десятичные		K32768 ~ K32767 (16 бит) K2147483648 ~ K2147483647 (32 бит)		
	H	Шестнадцатеричные		H0000 ~ HFFFF (16 бит) H0000 0000 ~ HFFFF FFFF (32 бит)		

Примечание

Энергонезависимые ячейки являются фиксированными и не могут быть переопределены в энергозависимую область

Перечень доступных операндов для контроллеров типов SA/SX/SC

Элемент				Спецификация		Примечание
Битовые операнды (реле)	**X**	Физические входы		X0 ~ X177, 128 точек, восьмеричная система	Всего 256 точек	Соответствуют внешним точкам ввода/вывода
	Y	Физические выходы		Y0 ~ Y177, 128 точек, восьмеричная система		
	M	Вспомогательные реле (маркеры)	Общие Энергонезав. Специальные	M0 ~ M511, 512 точек (*1) M512 ~ M999, 488 точек (*3) M2000 ~ M4095, 2096 точек (*3) M1000 ~ M1999, 1000 точек	Всего 4096 точек	Используются в программе как промежуточные реле
	T	Таймер	100 мс 10 мс 1 мс	T0 ~ T199, 200 точек (*1) T192 ~ T199 для подпрограмм T250 ~ T255, 6 точек аккумулятивного типа (*4) T200 ~ T239, 40 точек (*1) T240 ~ T245, 6 точек аккумулятивного типа (*4) T246 ~ T249, 4 точки аккумулятивного типа (*4)	Всего 256 точек	Инициируется инструкцией TMR. Когда отсчет времени достигнет уставки, то замкнется контакт «T» с соответствующим номером

Битовые операнды (реле)	**C**	Счетчик	16-бит счет вверх 32-бит счет вверх/вниз SA/SX 32-бит высокоскор. счет вверх/вниз	C0 ~ C95, 96 точек (*1) C96 ~ C199, 104 точек (*3) C200 ~ C215, 16 точек (*1) C216 ~ C234, 19 точек (*3) C235 ~ C242, C244, 1 фаза 1 вход, 9 точек (*3) C246, C247, C249, 1 фаза 2 входа, 3 точки (*3) C251 ~ C254, 2 фазы 2 входа, 4 точки (*3)	Всего 235 точек Всего 16 точек	Инициируется инструкцией CNT (DCNT). Когда счет достигнет уставки, то замкнется контакт «C» с соответствующим номером
			SC 32-бит высокоскор. счет вверх/вниз	C235 ~ C245, 1 фаза 1 вход, 11 точек (*3) C246, C247, C249, C250, 1 фаза 2 входа, 4 точки (*3) C251, C252, C254, C255, 2 фазы 2 входа, 4 точки (*3)	Всего 19 точек	
	S	Шаговое реле	Инициализир. Возвращение в нулев. точку Общие Энергонезав. Аварийные	S0 ~ S9, 10 точек (*1) S10 ~ S19, 10 точек (*1), исп. с инструкцией IST S20 ~ S511, 492 точки (*1) S512 ~ S895, 384 точки (*3) S896 ~ S1023, 128 точки (*3)	Всего 1024 точки	Установка энергонезависим. области: Начало: D1214 (K512) Конец: D1215 (K895)
Словные операнды (регистры)	**T**	Текущее значение таймера		T0 ~ T255, 256 точек		
	C	Текущее значение счетчика		C0 ~ C199, 16 бит, 200 точек C200 ~ C254, 32 бит, 50 точек		
	D	Регистры данных	Общие Энергонезав. Специальные Индексные	D0 ~ D199, 200 точек (*1) D200 ~ D999, 800 точек (*3) D2000 ~ D4999, 3000 точек (*3) D1000 ~ D1999, 1000 точек E0 ~ E3, F0 ~ F3, 8 точек (*1)	Всего 5000 точек	Область для хранения данных. Может использоваться для косвенной индексации
	–	Файловые регистры		0 ~ 1599, 1600 точек (*4)	Дополнительные регистры для хранения данных	
Индексы	**N**	Для мастер-контроля		N0 ~ N7, 8 точек		
	P	Для инструкций CJ, CALL		P0 ~ P255, 256 точек		
	I	Для прерываний	Внешние Временные Высокоскор. счетчика Коммуникац.	I001, I101, I201, I301, I401, I501 6 точек I6xx, I7xx (xx = 1~99), шаг 1 мс 2 точки I010, I020, I030, I040, I050, I060, 6 точек I150, 1 точка		Позиционный указатель для подпрограммы прерывания
Константы	**K**	Десятичные		K32768 ~ K32767 (16 бит) K2147483648 ~ K2147483647 (32 бит)		
	H	Шестнадцатеричные		H0000 ~ HFFFF (16 бит) H0000 0000 ~ HFFFF FFFF (32 бит)		

Примечание

*1 – энергозависимая область, не может быть переопределена

*2 – по умолчанию энергозависимая область, может быть переопределена в энергонезависимую путем выставления соответствующих параметров

*3 - по умолчанию энергонезависимая область, может быть переопределена в энергозависимую путем выставления соответствующих параметров

*4 – энергонезависимая область, не может быть переопределена

Адресация для определения областей энергозависимых и энергонезависимых регистров

Вспомогательные реле (M)

Общие	**Энергонезависимые**	**Специальные**	**Энергонезависимые**
M0 ~ M511	M512 ~ M999	M1000 ~ M1999	M2000 ~ M4095
не могут быть переопределены	по умолч. энергонез. могут быть переопред. Начало: D1200 (K512) Конец: D1201 (K999)	частично энергонезависимая область, не подлежит переопределению	по умолчанию энергонезависимые, могут быть переопределены Начало: D1202 (K2000) Конец: D1203 (K4095)

Таймеры (T)

100 мс	**10 мс**	**10 мс**	**1 мс**	**100 мс**
T0 ~ T199	T200 ~ T239	T240 ~ T245	T246 ~ T249	T250 ~ T255
Энергозависимые, не могут быть переопределены		Аккумулятивного типа, энергонезависимые, переопределению не подлежат		

Счетчики (C)

16 бит, счет вверх		**32 бит, счет вверх/вниз**		**32 бит, высокоскоростной счет вверх/вниз**	
C0 ~ C95	**C96 ~ C199**	**C200 ~ C215**	**C216 ~ C234**	**C235 ~ C245**	**C246 ~ C255**
фиксировано энергозависимые	по умолчанию энергонезавис.	фиксировано энергозависимые	по умолчанию энергонезавис.	по умолчанию энергозависимые	
Начало: D1208 (K96) Конец: D1209 (K199)		Начало: D1210 (K216) Конец: D1211 (K234)		Начало: D1212 (K235) Конец: D1213 (K255)	

Шаговые реле (S)

Инициализирующие	**Выход в нулевую точку**	**Общие (энергозависимые)**	**Энергонезависимые**	**Аварийные**
S0 ~ S9	**S10 ~ S19**	**S20 ~ S511**	**S512 ~ S895**	**S896 ~ S1023**
Фиксировано энергозависимые, переопределению не подлежат			по умолчанию энергонезависимые, могут быть переопределены Начало: D1214 (K512) Конец: D1215 (K895)	Фиксировано энергонезависимые, переопределению не подлежат

Регистры данных (D)

Общие	Энергонезависимые	Специальные	Энергонезависимые
D0 ~ D199	D200 ~ D999	D1000 ~ D1999	D2000 ~ D4999
Энергозависимые, не могут быть переопределены	по умолчанию энергонезависимые, могут быть переопределены Начало: D1216 (K200) Конец: D1217 (K999)	используются системой	по умолчанию энергонезависимые, могут быть переопределены Начало: D1218 (K2000) Конец: D1219 (K9999)

Файловые регистры

K0 ~ K1599
По умолчанию энергонезависимые, переопределению не подлежат

Перечень доступных операндов для контроллеров типов SV/EH/EH2

Элемент				Спецификация		Примечание
Битовые операнды (реле)	X	Физические входы		X0 ~ X377, 256 точек, восьмеричная система	Всего 512 точек	Соответствуют внешним точкам ввода/вывода
	Y	Физические выходы		Y0 ~ Y377, 256 точек, восьмеричная система		
	M	Вспомогательные реле (маркеры)	Общие Энергонезав. Специальные	M0 ~ M499, 500 точек (*2) M500 ~ M999, 500 точек (*3) M2000 ~ M4095, 2096 точек (*3) M1000 ~ M1999, 1000 точек	Всего 4096 точек	Используются в программе как промежуточные реле
	T	Таймер	100 мс 10 мс 1 мс	T0 ~ T199, 200 точек (*2) T192 ~ T199 для подпрограмм T250 ~ T255, 6 точек аккумулятивного типа (*4) T200 ~ T239, 40 точек (*2) T240 ~ T245, 6 точек аккумулятивного типа (*4) T246 ~ T249, 4 точки аккумулятивного типа (*4)	Всего 256 точек	Инициируется инструкцией TMR. Когда отсчет времени достигнет уставки, то замкнется контакт «T» с соответствующим номером
	C	Счетчик	16-бит счет вверх 32-бит счет вверх/вниз 32-бит высокоскор. счет вверх/вниз	C0 ~ C99, 100 точек (*2) C100 ~ C199, 100 точек (*3) C200 ~ C219, 20 точек (*2) C220 ~ C234, 15 точек (*3) C235 ~ C244, 1 фаза 1 вход, 10 точек (*3) C246 ~ C249, 1 фаза 2 входа, 4 точки (*3) C251 ~ C254, 2 фазы 2 входа, 4 точки (*3)	Всего 253 точки	Инициируется инструкцией CNT (DCNT). Когда счет достигнет уставки, то замкнется контакт «C» с соответствующим номером
	S	Шаговое реле	Инициализир. Возвращение в нулев. точку Общие Энергонезав. Аварийные	S0 ~ S9, 10 точек (*2) S10 ~ S19, 10 точек (*2), исп. с инструкцией IST S20 ~ S499, 480 точек (*2) S500 ~ S899, 400 точек (*3) S900 ~ S1023, 124 точки (*3)	Всего 1024 точки	Установка энергонезависим. области: Начало: D1214 (K500) Конец: D1215 (K899)
Словные операнды (регистры)	T	Текущее значение таймера		T0 ~ T255, 256 точек		
	C	Текущее значение счетчика		C0 ~ C199, 16 бит, 200 точек C200 ~ C254, 32 бит, 53 точки		
	D	Регистры данных	Общие Энергонезав. Специальные Индексные	D0 ~ D199, 200 точек (*2) D200 ~ D999, 800 точек (*3) D2000 ~ D9999, 8000 точек (*3) D1000 ~ D1999, 1000 точек E0 ~ E7, F0 ~ F7, 16 точек (*1)	Всего 10000 точек	Область для хранения данных. Может использоваться для косвенной индексации
	–	Файловые регистры		0 ~ 9999, 10000 точек (*4)	Регистры расширения для хранения данных	

Индексы	**N**	Для мастер-контроля		N0 ~ N7, 8 точек		
	P	Для инструкций CJ, CALL		P0 ~ P255, 256 точек		
	I	Для прерываний	Внешние	I00x (X0), I10x (X1), I20x (X2), I30x (X3), I40x (X4), I50x (X5); 6 точек (x=1 – передний фронт, x=0 – задний фронт)		Позиционный указатель для подпрограммы прерывания
			Временные	I6xx (1 мс), I7xx (1 мс), I8xx (0,1 мс); xx = 1~99		
			Высокоскор. счетчика	I010, I020, I030, I040, I050, I060, 6 точек		
			Импульсные	I110, I120, I130, I140, 4 точки		
			Коммуникац.	I150, I160, I170, 3 точки		
Константы	**K**	Десятичные		K32768 ~ K32767 (16 бит) K2147483648 ~ K2147483647 (32 бит)		
	H	Шестнадцатеричные		H0000 ~ HFFFF (16 бит) H0000 0000 ~ HFFFF FFFF (32 бит)		
	F	С плавающей точкой		Отображение плавающей точки длинной 32 бит в соотв. со стандартом IEEE754 +/- 1,1755x10^{-38} ~ +/- 3,4028x10^{+38}		

Примечание

*1 – энергозависимая область, не может быть переопределена

*2 – по умолчанию энергозависимая область, может быть переопределена в энергонезависимую путем выставления соответствующих параметров

*3 - по умолчанию энергонезависимая область, может быть переопределена в энергозависимую путем выставления соответствующих параметров

*4 – энергонезависимая область, не может быть переопределена

Адресация для определения областей энергозависимых и энергонезависимых регистров

Вспомогательные реле (M)

Общие	**Энергонезависимые**	**Специальные**	**Энергонезависимые**
M0 ~ M499	M500 ~ M999	M1000 ~ M1999	M2000 ~ M4095
Начало: D1200 (K512) Конец: D1201 (K999)		частично энергонезависимая область, не подлежит переопределению	по умолчанию энергонезависимая, может быть переопределена
			Начало: D1202 (K2000) Конец: D1203 (K4095)

Таймеры (T)

100 мс	**10 мс**	**10 мс**	**1 мс**	**100 мс**
T0 ~ T199	T200 ~ T239	T240 ~ T245	T246 ~ T249	T250 ~ T255
по умолчанию энергозависимые, могут быть переопределены		Аккумулятивного типа, энергонезависимые, переопределению не подлежат		
Начало: D1204 Конец: D1205	Начало: D1206 Конец: D1207			

Счетчики (C)

16 бит, счет вверх		**32 бит, счет вверх/вниз**		**32 бит, высокоскоростной счет вверх/вниз**	
C0 ~ C99	**C100 ~ C199**	**C200 ~ C219**	**C220 ~ C234**	**C235 ~ C245**	**C246 ~ C255**
по умолчанию энергозависимые	по умолчанию энергонезавис.	по умолчанию энергозависимые	по умолчанию энергонезавис.	по умолчанию энергозависимые	
Начало: D1208 (K96) Конец: D1209 (K199)		Начало: D1210 (K216) Конец: D1211 (K234)		Начало: D1212 (K235) Конец: D1213 (K255)	

Шаговые реле (S)

Инициализирующие	Возврат в нулевую точку	Общие	Энергонезависимые	Аварийные
S0 ~ S9	S10 ~ S19	S20 ~ S499	S500 ~ S899	S900 ~ S1023
	по умолчанию энергозависимые		по умолчанию энергонезависимые	энергонезависимые, переопределению не подлежат
	Начало: D1214 (K500), Конец: D1215 (K899)			

Регистры данных (D)

Общие	Энергонезависимые	Специальные	Энергонезависимые
D0 ~ D199	D200 ~ D999	D1000 ~ D1999	D2000 ~ D9999
по умолчанию энергозависимые	по умолчанию энергонезависимые, могут быть переопределены	используются системой	по умолчанию энергонезависимые, могут быть переопределены
Начало: D1216 (K200) Конец: D1217 (K999)			Начало: D1218 (K2000) Конец: D1219 (K9999)

Файловые регистры

K0 ~ K9999
По умолчанию энергонезависимые, переопределению не подлежат

Реакция системы на изменение режимов Вкл./Выкл. (ON/OFF), Работа/Стоп (Run/Stop)

У контроллеров типов ES/EX/SS

Тип регистров	Питание Off -> On	Stop => Run	Run => Stop	очистить общую память (M1031=ON)	очистить энергонезавис. память (M1032=ON)	Заводская установка
Общие	очистка	очистка при M1033=Off без изменений при M1033=On		очистка	без изменений	0
Энергонезав.	без изменений			без изменений	очистка	0
Специальные M и D	исходная уставка	без изменений		без изменений		исходная уставка
Файловые	без изменений					0

У контроллеров типов SA/SX/SC/SV/EH/EH2

Тип регистров	Питание Off -> On	Stop => Run	Run => Stop	очистить общую память (M1031=ON)	очистить энергонезавис. память (M1032=ON)	Заводская установка
Общие	очистка	без изменений	очистка при M1033=Off без изменений при M1033=On	очистка	без изменений	0
Энергонезав.	без изменений			без изменений	очистка	0
Специальные M и D	исходная уставка	без изменений		без изменений		исходная уставка
Файловые	без изменений					0

2.2 Описание форматов числовых значений и констант [K] и [H]

Для осуществления вычислений, операций с данными, присвоения адресов, определения уставок и подобных операций в программе ПЛК всегда присутствуют числовые значения и константы.
С помощью констант задаются какие-либо фиксированные параметры, например уставки счетчиков и таймеров, а числовые значения получаются в процессе вычислений, обработки данных и т.п.
Пользователь оперирует с константами и числовыми значениями в основном в десятичном формате. Контроллер для своих внутренних вычислений использует двоичную систему.
Десятичные константы задаются операндом «K», а шестнадцатеричные операндом «H», которые могут принимать следующий диапазон значений:

Константы	K	Десятичные	K-32,768 ~ K32,767 (16 бит) K-2,147,483,648 ~ K2,147,483,647 (32 бит)
	H	Шестнадцатеричные	H0 ~ HFFFF (16 бит) H0 ~ HFFFFFFFF (32 бит)

Например, K100 означает 100 в десятичном формате, а H100 означает 100 в шестнадцатеричном формате (в десятичном это 256).

Исключение составляет использование символа «K» для для представления однобитных операндов X, Y, M, S в виде байтов, слов и двойных слов. Например, K2Y10 или K4M100.
В данной инструкции K1 означает не число «1», а 4-х битный формат, K2 – 8-ми битный и т.д.
Подробнее см. Главу 5.

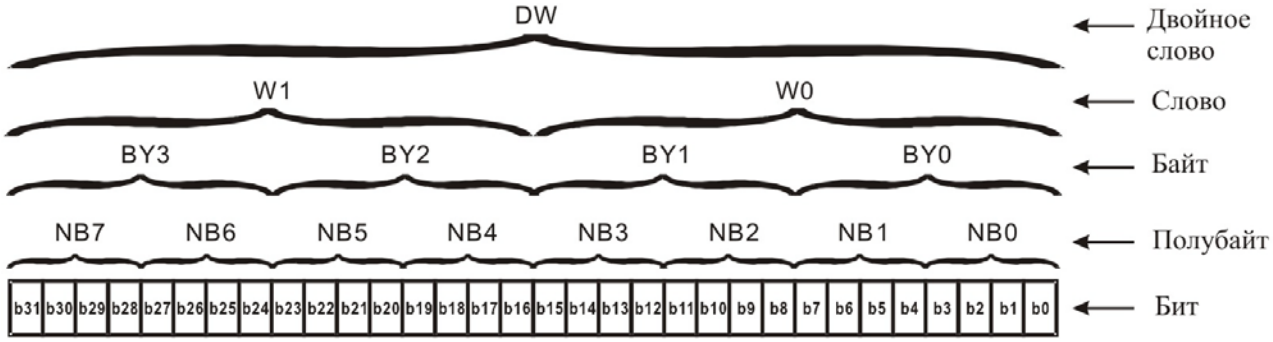
Числовые значения, используемые в программе ПЛК для различных целей, бывают 5-ти видов:

1. Двоичный формат (BIN)

Используется для контроллером для внутренних вычислений и хранения данных.
Данные в двоичном формате представляются следующими стандартными единицами:

Бит (Bit), b	-	Бит – базовая единица двоичной системы. Может принимать значения 1 или 0
Полубайт (Nibble), NB	-	Состоит из 4-х последовательных битов, например b3~b0. Используется для представления десятичных 0~9 или шестнадцатеричных 0~F символов
Байт (Byte), BY	-	Состоит из 8 последовательных битов b7~b0 или двух последовательных полубайтов. Используется для представления двухразрядных шестнадцатеричных символов 00~FF
Слово (Word), W	-	Состоит из 16 последовательных битов b15~b0 или двух последовательных байтов. Используется для представления четырехразрядных шестнадцатеричных символов 0000~FFFF
Двойное слово (Double Word), DW	-	Состоит из 32 последовательных битов b31~b0 или двух последовательных слов. Используется для представления восьмиразрядных шестнадцатеричных символов 00000000~FFFFFFFF

Взаимосвязь между различными единицами двоичной системы представлена на рисунке ниже:



2. Восьмеричный формат (OCT)

Данный формат используется для нумерации (адресации) внешних входов (X) и выходов (Y)

Внешние физические входы: X0~X7, X10~X17…(номер устройства)
Внешние физические выходы: Y0~Y7, Y10~Y17…(номер устройства)

3. Десятичный формат (DEC)

Десятичный формат является наиболее часто употребительным и используется для следующих задач:

- Задание уставок таймеров и счетчиков, например TMR C0 K50 (K - константа)
- Присвоение номеров операндам S, M, T, C, D, E, F, P, I, например M10, T30
- Для MOV K123 D0. (K - константа)

4. Двоично-десятичный формат (BCD)

Данный формат используется для чтения входных значений от DIP-переключателей или для отображения выходных значений на 7-ми сегментном индикаторе.
В данном формате десятичный символ представляется четырехразрядным двоичным числом. Для этого каждое десятичное число последовательно записывается соответствующим двоичным числом (не путать с переводом десятичного числа в двоичное!, см. таблицу ниже).

5. Шестнадцатеричный формат (HEX)

Используется для задания и отображения значений в прикладных инструкциях, когда это удобно, например для записи адреса регистра памяти MOV H1A2B D0 (H - константа)

Ниже приводится сводная таблица для сравнения различных форматов представления чисел.

Таблица соотношений числовых форматов в DVP-PLC

BIN		**OCT**	**DEC**	**BCD**		**HEX**
Для внутренних вычислений		Адресация входов/ выходов X/Y	Константы K, адресация S, M, T, C, D, E, F, P, I	Для DIP-переключателей и 7-ми сегментных индикаторов		Константы H
0000	0000	0	0	0000	0000	0
0000	0001	1	1	0000	0001	1
0000	0010	2	2	0000	0010	2
0000	0011	3	3	0000	0011	3
0000	0100	4	4	0000	0100	4
0000	0101	5	5	0000	0101	5
0000	0110	6	6	0000	0110	6
0000	0111	7	7	0000	0111	7
0000	1000	10	8	0000	1000	8
0000	1001	11	9	0000	1001	9
0000	1010	12	10	0001	0000	A
0000	1011	13	11	0001	0001	B
0000	1100	14	12	0001	0010	C
0000	1101	15	13	0001	0011	D
0000	1110	16	14	0001	0100	E
0000	1111	17	15	0001	0101	F
0001	0000	20	16	0001	0110	10
0001	0001	21	17	0001	0111	11
…	…	…	…	…	…	…
0110	0011	143	99	1001	1001	63

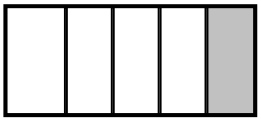
2.3 Адресация и назначение внешних контактов входов [X] и выходов [Y]

Так как контроллеры осуществляют управления различными технологическими установками и объектами, они оснащены физическими контактами для подключения внешних источников сигналов (кнопки, датчики), которые обозначаются Xn, где n – порядковый номер контакта, а также для подключения внешних приемников сигналов (катушки реле, входные каскады электронных устройств и т.п.), которые обозначаются Yn, где n – порядковый номер контакта.

Для обращения к внешним физическим контактам (внешним по отношению к программе ПЛК, которая эмулирует свои внутренние объекты), существуют специальные операнды, которые для удобства обозначаются как и физические контакты Xn и Yn.
Операнды Xn по номерам соответствуют физическим входам. Например, при обращении в программе к операнду X2 происходит считывание состояния физического входа X2 (есть внешний сигнал на нем или нет).
Операнды Yn по номерам соответствуют физическим выходам. Например, при обращении в программе к операнду Y2 происходит включение или выключение физического выхода Y2, который коммутирует сигнал на внешнюю нагрузку. Физические выходы работают в режиме ключа, т.е. пропускают или не пропускают сигнал в нагрузку.
Нумерация входов и выходов осуществляется в восьмеричном формате, т.е. от 0 до 7, а

цифры 8 и 9 не используются. Первый контакт на центральном процессорном модуле всегда X0/Y0. Общее количество точек входов/выходов на центральном модуле зависит от модели ПЛК (см. ниже).
Для увеличения точек ввода/вывода к контроллеру могут присоединяться модули расширения (EXT). Нумерация входов и выходов зависит от порядкового номера модуля по отношению к центральному процессорному модулю (MPU).
У модуля расширения, ближайшему к центральному модулю входы будут начинаться с X20, а выходы Y20 независимо от того, сколько входов/выходов было на центральном модуле. У следующего модуля расширения с X30 или Y30. Если у модуля только входы, то нумерация выходов пропускается и переходит на следующий модуль и наоборот, как показано на рисунке ниже.

Пример конфигурации:



ПЛК	Модели	Число входов	Число выходов	Нумерация входов	Нумерация выходов
MPU	SS/SA/SX/SC	8	4/6	X0~X7	Y0~Y5
EXT1	DVP16SP11T	8	8	X20~X27	Y20~Y27
EXT2	DVP08SM11N	8	0	X30~X37	-
EXT3	DVP06SM11R	0	6	-	Y30~Y35
EXT4	DVP08SP11R	4	4	X40~X43	Y40~Y43

Примечание.
На корпусе модуля расширения входы/выходы нумеруются как X0-X7 и Y0-Y7, а в программе их нумерация будет зависеть от положения модуля расширения относительно центрального процессорного модуля.

В ниже приведенных таблицах отображается количество дискретных точек ввода/вывода в зависимости от типа ПЛК.

Тип ES/EX/SS

Модель	DVP-14ES	DVP-14SS	DVP-20EX	DVP-24ES	DVP-32ES	DVP-60ES	Расширение входов/выходов
Входы X	X0~X7 (8 точек)	X0~X7 (8 точек)	X0~X7 (8 точек)	X0~X17 (16 точек)	X0~X17 (16 точек)	X0~X43 (36 точек)	X20(X50)~X177
Выходы Y	Y0~Y5 (6 точек)	Y0~Y5 (6 точек)	Y0~Y5 (6 точек)	Y0~Y7 (8 точек)	Y0~Y17 (16 точек)	Y0~Y27 (24 точки)	Y20(Y30)~Y177

Примечание.
Во всех моделях кроме DVP60ES нумерация входов модулей расширения начинается с X20, а выходов с Y20. В модели DVP60ES нумерация входов модулей расширения начинается с X50, а выходов с Y30. Адресация входов/выходов в модулях расширения увеличивается на 8, даже если в модуле входов/выходов меньше восьми.

Тип SA/SX/SC

Модель	DVP-10SX	DVP-12SA	DVP-12SC	Расширение входов/выходов
Входы X	X0~X3 (4 точки)	X0~X7 (8 точек)	X0~X5, X10~X11 (8 точек)	X20~X177
Выходы Y	Y0~Y1 (2 точки)	Y0~Y3 (4 точки)	Y0~Y1, Y10~Y11 (4 точки)	Y20~Y177

Тип SV

Модель	DVP-28SV	Расширение входов/выходов
Входы X	X0~X7, X10~X17 (16 точек)	X20~X377
Выходы Y	Y0~Y4, Y5~Y7, Y10~Y13 (12 точек)	Y20~Y377

Примечание.

- Для контроллеров типов SA/SX/SC/SV используются те же модули расширения дискретных входов/выходов, что и для типа SS. У модели DVP-10SX число дискретных входов/выходов уменьшено вследствие наличия 2-х аналоговых входов и 2-х выходов.
- Нумерация входов модулей расширения начинается с с X20, а выходов с Y20.
- У модели DVP-28SV выходы Y0 ~ Y7, могут работать в высокоскоростном режиме до 200 кГц каждый.
- У модели DVP-12SC выходы Y10 и Y11 могут работать в высокоскоростном режиме до 100 кГц (один из выходов) с общим диапазоном 130 кГц.
- У моделей DVP-28SV и DVP-12SC часть входов X может работать с высокоскоростными счетчиками (см. раздел описания счетчиков).

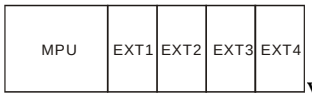
Тип EH/EH2

Модель	DVP-16EH	DVP-20EH	DVP-32EH	DVP-48EH	DVP-64EH	DVP-80EH	Расширение входов/выходов
Входы X	X0~X7 (8 точек)	X0~X13 (12 точек)	X0~X17 (16 точек)	X0~X27 (24 точки)	X0~X37 (32 точки)	X0~X47 (40 точек)	X~X377
Выходы Y	Y0~Y7 (8 точек)	Y0~Y7 (8 точек)	Y0~Y17 (16 точек)	Y0~Y27 (24 точки)	Y0~Y37 (32 точки)	Y0~Y47 (40 точек)	Y~Y377

Примечание.

- У моделей DVP-20EH и DVP-32EH выходы Y0 и Y2 являются высокоскоростными до 200 кГц каждый.
- У модели DVP-40EH выходы Y0 ~ Y3, Y4 и Y6 являются высокоскоростными до 200 кГц каждый.
- У модели DVP-32EH компоновка выходов отличается от других моделей, см. описание аппаратной части контроллера.
- У моделей DVP-16EH, DVP-20EH и DVP-32EH нумерация входов/выходов первого модуля расширения будет начинаться с X20/Y20. Для остальных моделей нумерация

начинается с номера, следующего за последним номером входа/выхода на центральном модуле, см. пример ниже.



MPU – центральный процессорный модуль
EXT – модуль расширения с номером, соответствующим степени удаления от центрального модуля

	Модуль	Входы	Выходы	Число входов	Число выходов
MPU	16EH/32EH/64EH	8/16/32	8/16/32	X0~X7/X0~X17/X0~X37	Y0~Y7/Y0~Y17/Y0~Y37
EXT1	32HP	16	16	X20~X37 / X20~X37/X40~X57	Y20~Y37/Y20~Y37/Y40~Y57
EXT2	48HP	24	24	X40~X67/X40~X67/X60~X107	Y40~Y67/Y40~Y67/Y60~Y107
EXT3	08HP	4	4	X70~X73/X70~X73/X110~X113	Y70~Y73/Y70~Y73/Y110~Y113
EXT4	08HN	0	8	-	Y74~Y103/Y74~Y103/Y114~Y123

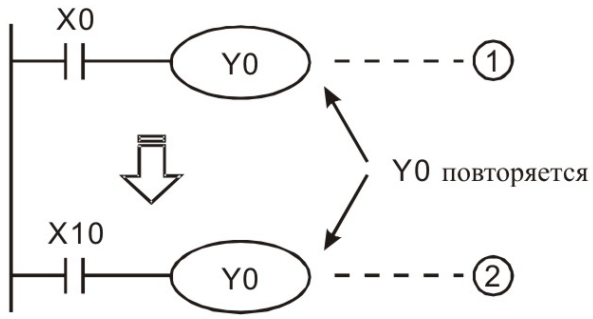
Назначение входов X

Данный операнд воспринимает сигналы от внешних источников (датчики, кнопки), подключенных непосредственно к клеммам контроллера (модулей расширения) и передает в процессор ПЛК. Каждый вход X может использоваться в программе неограниченное количество раз. В ступенчатой диаграмме обозначается как нормально открытый или закрытый контакт.

Включать и выключать вход можно как внешним сигналом, так и при помощи программатора HPP или программного пакета WPLSoft. Для этого необходимо включить в программе специальное реле M1304=ON (данная функция недоступна в контроллерах типов ES/EX/SS).

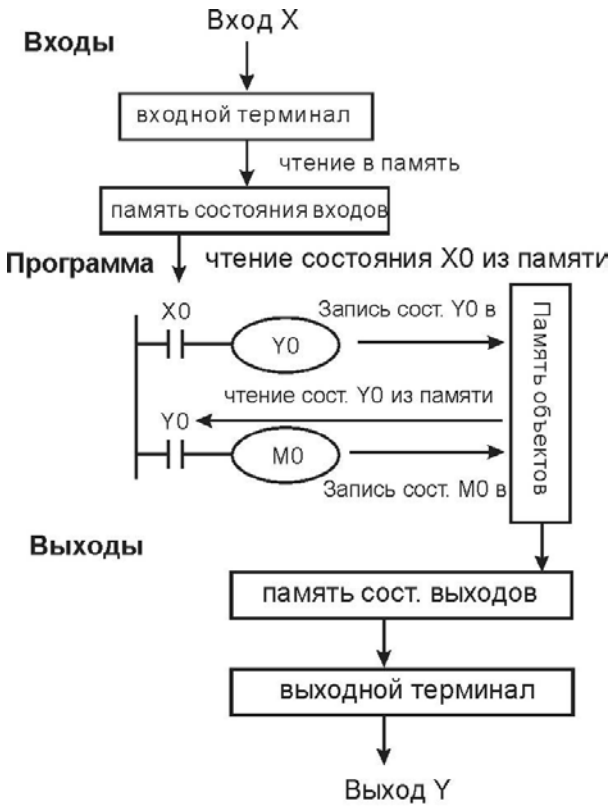
Назначение выходов Y

Данный операнд предназначен для подключения или отключения нагрузки, подключенной непосредственно к клеммам контроллера (модуля расширения). В качестве нормально открытого или закрытого контакта выход Y может использоваться неограниченное количество раз. Однако, в качестве выходной катушки может использоваться 1 раз, так как в противном случае возникнет ошибка как показано на примере ниже.



Ввиду того, что сканирование идет сверху вниз, состояние выходной катушки Y0 будет определяться только входным контактом X10.

Порядок обработки входов и выходов контроллером



- Входы:
  1. Перед каждым сканом программы ПЛК осуществляет групповое чтение всех входов (наличие или отсутствие внешних сигналов на своих клеммах) и записывает в память состояния входов.
  2. В ходе исполнения скана программы появление новых сигналов не изменит состояние в памяти входов, которое туда было записано при чтении перед исполнением текущего скана. Новые сигналы будут выявлены и записаны в память входов только на следующем скане.
  3. Время задержки с момента появления сигнала на входе до изменения состояния контакта Вкл→Выкл или Выкл→Вкл составляет 10 мс (фильтр помех, уставку можно изменить).
- Программа:
  ПЛК полностью выполняет программу от начала до конца. Состояние входов считывается из памяти входов. В соответствии с заложенной программой контроллер изменяет состояние выходов, которые записываются в память выходов (физические выходы при этом состояние не меняют).
- Выходы:
  1. После выполнения инструкции END в программе, состояние выходов из памяти выходов посылается на физические выходы, т.е. на катушки выходных реле, которые фиксируют свое состояние до появление новой команды.
  2. Время задержки с момента подачи сигнала на катушку до срабатывания контакта реле с Выкл на Вкл или с Вкл на Выкл составляет 10 мс.
  3. Время задержки с момента подачи сигнала на транзистор до его открытия составляет 10-20 мкс.

2.4 Адресация и назначение внутренних реле [M]

Внутренние реле применяются для запоминания двоичных результатов логических связей (состояний сигналов "0" или "1") внутри программы. Они соответствуют промежуточным реле в системах управления на релейно-контактной логике.

В контроллерах DVP используется три типа внутренних реле:

1. Общие
Не сохраняют свое состояние при отключении питания, т.е. при повторной подаче питания промежуточные реле данного типа будут в состоянии "Выкл".

2. Энергонезависимые
Сохраняют свое состояние при отключении питания.

3. Специальные
Предоставляют в распоряжение пользователя различные полезные функции (см. Главы 2.10 и 2.11). Специальные реле нельзя использовать как обычные вспомогательные реле, они могут использоваться только в соответствии со своим фукциональным предназначением.

В программе внутренние реле могут использоваться как контакты и как выходы. Однако они не могут воспринимать сигналы от внешних устройств, для этого нужно использовать операнды X, и не могут воздействовать на внешние выходы, для этого нужно использовать операнды Y. В программе каждое внутренее реле может использоваться неограниченное количество раз.

Адресация внутренних реле выполняется в десятичном формате.

Тип ES/EX/SS

Вспомогательные реле M	Общие	M0~M511, M768~M999, 744 точки. Фиксировано энергозависимые. Переопределению не подлежат.	Всего 1280 точек
	Энергонезависимые	M512~M767, 256 точек. Фиксировано энергонезависимые. Переопределению не подлежат.	
	Специальные	M1000~M1279, 280 точек.	

Тип SA/SX/SC

Вспомогательные реле M	Общие	M0~M511, 512 точки. Фиксировано энергозависимые. Переопределению не подлежат.	Всего 4096 точек
	Энергонезависимые	M512~M999, M2000~M4095, 2584 точек. По умолчанию энергонезависимые, могут быть переопределены.	
	Специальные	M1000~M1999, 1000 точек.	

Тип EH/EH2/SV

Вспомогательные реле M	Общие	M0~M499, 500 точек. По умолчанию энергозависимые, могут быть переопределены.	Всего 4096 точек
	Энергонезависимые	M500~M999, M2000~M4095, 2596 точек. По умолчанию энергонезависимые, могут быть переопределены.	
	Специальные	M1000~M1999, 1000 точек.	

2.5 Адресация и назначение шаговых реле [S]

Шаговое реле является базовым элементом пошагового управления в шаговой ступенчатой диаграмме или в языке последовательных функциональных блоков (SFC), в котором они должны использоваться с командами STL/RET.

В программе шаговые реле могут использоваться как контакты и как выходы. Однако они не могут воспринимать сигналы от внешних устройств, для этого нужно использовать операнды X, и не могут воздействовать на внешние выходы, для этого нужно использовать операнды Y. В программе шаговые реле могут использоваться неограниченное количество раз (за исключением инициализирующих). Если шаговые реле не используются в пошаговой инструкции, то они могут использоваться как внутренние реле.

В контроллерах DVP используются пять типов шаговых реле:

1. Инициализирующие шаговое реле - S0~S9, 10 точек.
   В SFC используется как точка начала процесса.
2. Шаговое реле возврата в нулевую точку - S10~S19, 10 точек.
   S10 – S19 используются совместно с инструкцией API 60 IST для возврата в исходную точку. Если данная инструкция не задействована, то могут использоваться в программе как обычные внутренние реле.
3. Шаговые реле общего назначения - Используются в программе по усмотрению пользователя. При пропадании питания не сохраняют свое текущее состояние.
   SA, SX, SC: S20~S511, 492 точки.
   EH/EH2/SV: S20~S499, 480 точек.
4. Энергонезависимые шаговые реле - Используются в программе по усмотрению пользователя. При пропадании питания сохраняют свое текущее состояние.
   ES, EX, SS: S20~S127, 108 точек.
   SA, SX, SC: S512~S895, 384 точки.
   EH/EH2/SV: S500~S899, 400 точек.
5. Аварийные шаговые реле - Используются совместно с прикладной инструкцией API 46 ANS в качестве контакта для аварийного сигнала. Также, используются для записи аварийных событий и для устранения последствий нарушения функционирования внешнего оборудования.
   SA, SX, SC: S896~S1023, 128 точек.
   EH/EH2/SV: S900~S1023, 124 точки.

Адресация шаговых реле выполняется в десятичном формате.

Тип ES/EX/SS

Шаговые реле S	Инициализирующие	S0~S9, 10точек. Фиксировано энергонезависимые.	Всего 128 точек
	Возвращение в нулевую точку	S10~S19, 10 точек. Фиксировано энергонезависимые.	
	Энергонезависимые	S20~S127, 108 точек. Фиксировано энергонезависимые.	

Тип SA/SX/SC

Шаговые реле S	Инициализирующие	S0~S9, 10 точек. Фиксировано энергонезависимые, переопределению не подлежат.	Всего 1024 точки
	Возвращение в нулевую точку	S10~S19, 10 точек. Фиксировано энергонезависимые, переопределению не подлежат.	
	Общие	S20~S511, 492 точки. Фиксировано энергозависимые, переопределению не подлежат.	
	Энергонезависимые	S512~S895, 384 точки. По умолчанию энергонезависимые, могут быть переопределены.	
	Аварийные	S896~S1023, 128 точек. Фиксировано энергонезависимые, переопределению не подлежат.	

Тип EH/EH2/SV

Шаговые реле S	Инициализирующие	S0~S9, 10 точек. По умолчанию энергозависимые. Могут быть переопределены.	Всего 1024 точки
	Возвращение в нулевую точку	S10~S19, 10 точек. По умолчанию энергозависимые. Могут быть переопределены.	
	Общие	S20~S499, 480 точек. По умолчанию энергозависимые. Могут быть переопределены.	
	Энергонезависимые	S500~S899, 400 точек. По умолчанию энергонезависимые. Могут быть переопределены.	
	Аварийные	S900~S1023, 124 точки. По умолчанию энергонезависимые. Могут быть переопределены.	

2.6 Адресация и назначение таймеров [T]

Таймер предназначен для отсчета заданной уставки времени при выполнении входного условия. По достижении установленного значения замыкается контакт таймера с соответствующим номером. В программе таймеры могут использоваться как контакты и выходы. Однако воздействовать на внешние выходы не могут.
Таймеры бывают 3-х разновидностей: общие, аккумулятивные и для подпрограмм. По шагу уставки таймеры подразделяются также на 3 вида: с шагом 1 мс, 10 мс и 100 мс. Счет всегда идет вверх (в сторону увеличения). Количество шагов, которые необходимо отсчитать, задается десятичной константой "K". Также можно использовать регистр "D".

Значение уставки вычисляется следующим образом:

Уставка = Количество шагов (K или D) * на значение шага (1, 10 или 100 мс).

Например.

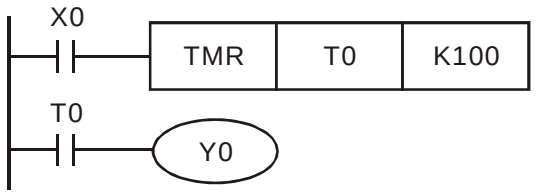
K=30
Шаг=100 мс
Уставка=3 сек.

Описание таймеров

1. Общие таймеры

Общие таймеры отсчитывают заданную уставку при непрерывном выполнении входного условия до достижения заданного значения. Если выполнение условия прерывается, то таймер сбрасывается в ноль и при возобновлении входного условия начинает отчитывать уставку заново.

Например.

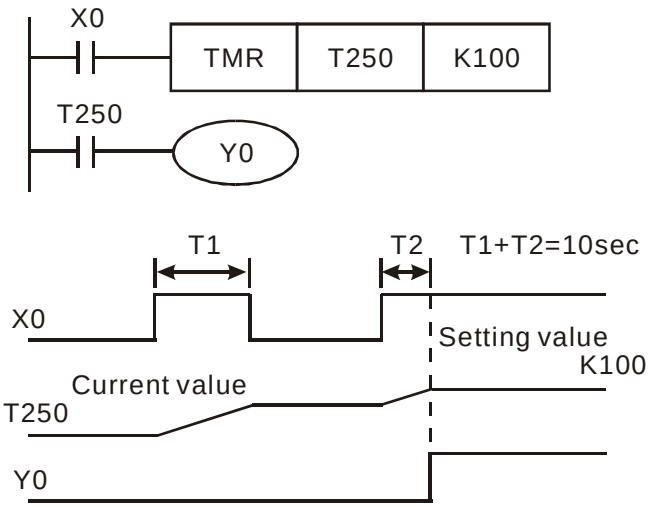


При замыкании контакта X0 начнется отсчет уставки таймера T0 (10 сек). При достижении уставки замкнется контакт таймера T0, который замкнет катушку Y0. Как только перестанет выполняться входное условие, контакт T0 разомкнется и разомкнет выходной контакт Y0.

Если при отсчете уставки таймера контакт X0 хотя бы на мгновение разомкнется, таймер T0 сбросится в ноль и при восстановлении входного условия отсчет начнется заново с нуля.

2. Аккумулятивные таймеры

Аккумулятивные таймеры сохраняют текущее отсчитанное значение уставки при прекращении выполнения входного условия. При восстановлении входного условия отсчет уставки таймера продолжается с последнего места и до достижения заданного значения. Сброс аккумулятивного таймера осуществляется командой RST.



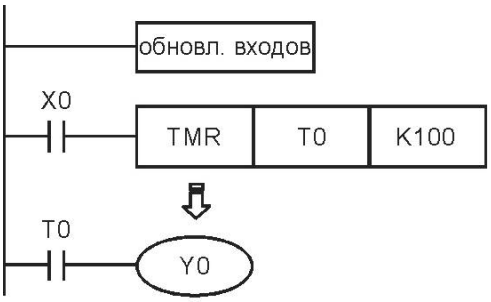
Допустим уставка таймера T250 10 сек. (K=100). При замыкании контакта X0 первый раз отсчет уставки не достигает заданного значения, так как контакт X0 размыкается раньше. Но таймер T250 "запомнит" накопленное значение и при повторном замыкании контакта X0 продолжит отсчет уставки дальше до достижения заданного значения, при достижении которого замкнется контакт T250, который замкнет катушку Y0. После этого состояние контакта X0 уже не будет влиять на состояние контакта T250. Таймер можно будет сбросить только командой RST.

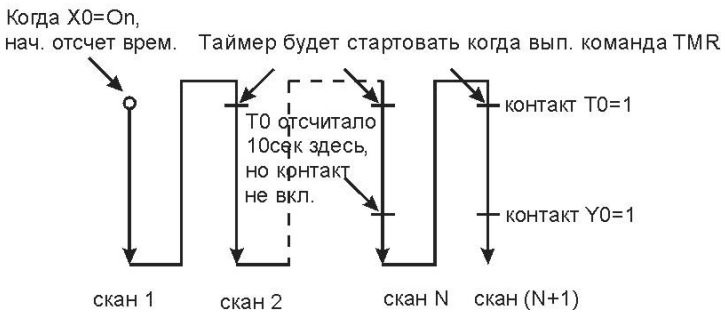
3. Таймеры для подпрограмм

Если таймер используется в подпрограмме или имеет прерывание в подпрограмме, применяйте таймеры с адресами T192-T199, т.к. обычные таймеры в этих случаях не будут работать корректно.

Точность таймера

В контроллерах ES/EX/SS/SA/SX/SC таймер начинает отчет времени после команды END в начале следующего скана. В контроллерах EH/EH2/SV таймер начинает отчет времени сразу с выполнением команды TMR.





Точность таймера составляет: $(T-\alpha) \leq T \leq (T+T0)$, где

T – заданное значение времени,

T0 – время цикла программы,

α – дискретность таймера (100 мс, 10 мс, 1 мс)

Если исполняемая инструкция рабочего контакта таймера находится в программе перед записью инструкции TMR, то ошибочная задержка будет составлять (+2T0), т.к. T+T0+T0 = T+2T0.

Если уставка времени T = 0, то рабочий контакт таймера сработает, как только в программе начнет отрабатываться инструкция, содержащая этот контакт.

Адресация таймеров выполняется в десятичном формате.

Тип ES/EX/SS

Таймер T	100 мс общий	T0~T63, 64 точки	Всего 128 точек
	10 мс общий	T64~T126, 63 точки (если M1028=On, шаг 10 мс. Если M1028=Off, шаг 100 мс)	
	1 мс общий	T127, 1 точка	

Тип SA/SX/SC

Таймер T	100 мс общий	T0~T199, 200 точек (T192~T199 таймеры для подпрограмм)	Всего 256 точек
	100 мс аккумулятивный	T250~T255, 6 точек. Фиксировано энергонезависимые, переопределению не подлежат.	
	10 общий	T200~T239, 40 точек	
	10 мс аккумулятивный	T240~T245, 6 точек. Фиксировано энергонезависимые, переопределению не подлежат.	
	1 мс аккумулятивный	T246~T249, 4 точки. Фиксировано энергонезависимые, переопределению не подлежат.	

Тип EH/EH2/SV

Таймер T	100 мс общие	T0~T199, 200 точек. Могут быть переопределены в аккумулятивные. (T192~T199 таймеры для подпрограмм)	Всего 256 точек
	100 мс аккумулятивные	T250~T255, 6 точек. Фиксировано энергонезависимые, переопределению не подлежат.	
	10 мс общие	T200~T239, 40 точек. Могут быть переопределены в аккумулятивные.	
	10 мс аккумулятивные	T240~T245, 6 точек. Фиксировано энергонезависимые, переопределению не подлежат.	
	1 мс аккумулятивные	T246~T249, 4 точки. Фиксировано энергонезависимые, переопределению не подлежат.	

2.7 Адресация и назначение счетчиков [C]

Счетчики используются для организации подсчета входных импульсов с дальнейшим их суммированием (счет вверх) или вычитанием (счет вниз). По достижении установленного значения замыкается контакт счетчика с соответствующим номером. В программе счетчики могут использоваться как контакты и выходы. Однако воздействовать на внешние (физические) выходы не могут.
Счетчики бывают 16 бит, 32 бит, обычные, скоростные, высокоскоростные, однофазные, двухфазные, со счетом вверх или вниз.

Адресация счетчиков осуществляется в десятичном формате.

Тип ES/EX/SS

16 бит, счет вверх	Общие	C0 ~ C111, 112 точек, фиксировано энергозависимые	Всего 141 точка
	Энергонезависимые	C112 ~ C127, 16 точек, фиксировано энергонезависимые	
32 бит, счет вверх/вниз, скоростные	однофазные 1 вход	C235 ~ C238, C241, C242, C244, 7 точек, фиксировано энергонезависимые	
	однофазные 2 входа	C246, C247, C249, 3 точки, фиксировано энергонезависимые	
	двухфазные 2 входа	C251, C252, C254, 3 точки, фиксировано энергонезависимые	

Тип SA/SX/SC

Счетчики простые	Общие 16 бит, счет вверх	C0 ~ C95, 96 точек, фиксировано энергозависимые	Всего 250 точек
	Энергонезависимые, 16 бит счет вверх	C96 ~ C199, 104 точки, по умолчанию энергонезависимые, могут быть переопределены	
	Общие 32 бит, счет вверх/вниз	C200 ~ C215, 15 точек, фиксировано энергозависимые	
	Энергонезависимые, 32 бит счет вверх/вниз	C216 ~ C234, 19 точек, по умолчанию энергонезависимые, могут быть переопределены	
Счетчики 32 бит, счет вверх/вниз, скоростные	однофазные 1 вход	C235 ~ C242, C244, 9 точек, по умолчанию энергонезависимые, могут быть переопределены	
	однофазные 2 входа	C246, C247, C249, 3 точки, по умолчанию энергонезависимые, могут быть переопределены	
	двухфазные 2 входа	C251, C252, C254, 3 точки, по умолчанию энергонезависимые, могут быть переопределены	
Счетчики 32 бит, счет вверх/вниз, высокоскоростные (только для типа SC)	однофазные 1 вход	C243, C245, 2 точки, по умолчанию энергонезависимые, могут быть переопределены	Всего 3 точки
	однофазный 2 входа	C250, 1 точка, по умолчанию энергонезависимая, может быть переопределена	

Тип EH/EH2/SV

Счетчики простые	Общие 16 бит, счет вверх	C0 ~ C99, 100 точек, по умолчанию энергонезависимые, могут быть переопределены	Всего 253 точки
	Энергонезависимые, 16 бит счет вверх	C100 ~ C199, 100 точек, по умолчанию энергонезависимые, могут быть переопределены	
	Общие 32 бит, счет вверх/вниз	C200 ~ C219, 20 точек, по умолчанию энергонезависимые, могут быть переопределены	
	Энергонезависимые, 32 бит счет вверх/вниз	C220~C234, 15 точек, по умолчанию энергонезависимые, могут быть переопределены	
Счетчики 32 бит, счет вверх/вниз, скоростные и высокоскоростные	Программные, однофазные 1 вход	C235~C240, 6 точек, по умолчанию энергонезависимые, могут быть переопределены	
	Аппаратные, однофазные 1 вход	C241~C244, 4 точки, по умолчанию энергонезависимые, могут быть переопределены	
	Аппаратные, однофазные 2 входа	C246~C249, 4 точки, по умолчанию энергонезависимые, могут быть переопределены	
	Аппаратные, двухфазные 2 входа	C251~C254, 4 точки, по умолчанию энергонезависимые, могут быть переопределены	

Общие характеристики счетчиков

Параметр	Счетчик 16 бит	Счетчик 32 бит	
Тип	Простой	Простой	Высокоскоростной
Направление счета	Счет вверх	Счет вверх/вниз	
Диапазон	0 ~ 32 767	-2 147 483 648 ~ +2 147 483 647	
Вариант записи уставки	Константа K или регистр D	Константа K или регистр D (2 последовательных)	
Реакция на достижение уставки	Счетчик остановится по достижении уставки	Счетчик продолжит работу по достижении уставки по кругу	
Реакция контакта	Когда счет достигнет уставки, контакт замкнется и зафиксируется	Когда счет вверх достигнет уставки, контакт замкнется и зафиксируется Когда счет вниз достигнет уставки, сбросится	
Сброс	Текущее значение счетчика и контакт сбрасываются командой RST		
Регистр хранения текущего значения	16 бит	32 бит	
Обновление состояния контактов	Групповое обновление в конце скана	Групповое обновление в конце скана	Включается сразу по достижении уставки. Никак не связан со временем скана

Счетчики 16 бит, C0 ~ C199

1. Диапазон уставки: K0 – K 32767. K0 и K1 по смыслу идентичны – контакт счетчика замкнется сразу при появлении первого импульса.
2. В счетчиках общего назначения (энергозависимых) текущее значение и контакт сбрасываются при отключении питания от ПЛК. В энергонезависимых счетчиках сохраняется текущее значение счета и состояние контакта. При возобновлении питания ПЛК счет продолжиться с того же места.
3. Если во время работы счетчика, который еще не достиг уставки, записать в регистр текущего значения счетчика величину большую, чем заданная уставка, то при

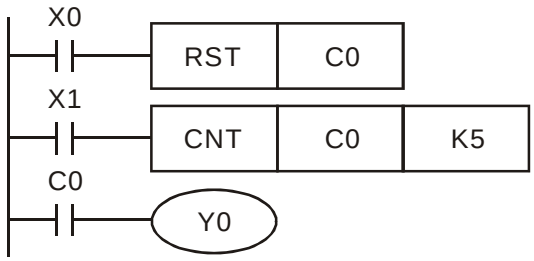
появлении ближайшего входного импульса счетчик включит свой контакт и значение уставки автоматически станет равно текущему значению, которое Вы записали в регистр. Запись в регистр можно осуществить путем команды MOV, программного пакета WPLSoft или программатора HPP.

4. Уставку счетчика можно записать прямым путем константой К, или косвенно, используя регистр D.
5. Если для задания уставки используется константа, то она должна быть только положительное значение. Если используется регистр, то он может иметь как положительное, так и отрицательное значение.
6. Когда текущее значение счетчика достигает 32767, то следующим будет: - 32768.

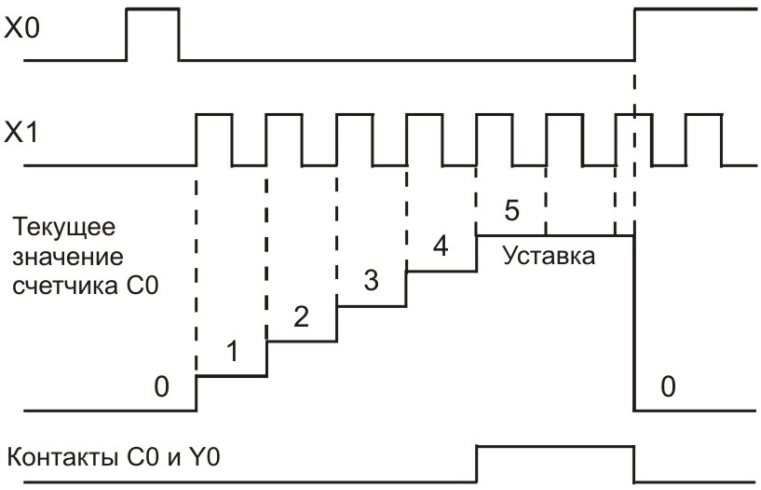
Пример использования счетчика

```
LD      X0
RST     C0
LD      X1
CNT     C0  K5
LD      C0

OUT     Y0
```



1. Когда X0=1, выполнится команда RST и текущее значение счетчика C0 сбросится на ноль, а контакт C0 перейдет в состояние ВЫКЛ
2. Когда X1 переходит с ВЫКЛ на ВКЛ счетчик отсчитывает 1 раз вверх
3. Когда текущее значение счетчика C0 достигнет уставки K5, замкнется контакт C0, а значение в регистре C0 останется K5 и импульсы от X1 перестанут восприниматься. Сбросить счетчик можно будет только командой RST. Контакт C0 включает выходную катушку Y0



Счетчики 32 бит общего назначения, счет вверх/вниз, C200 ~ C234

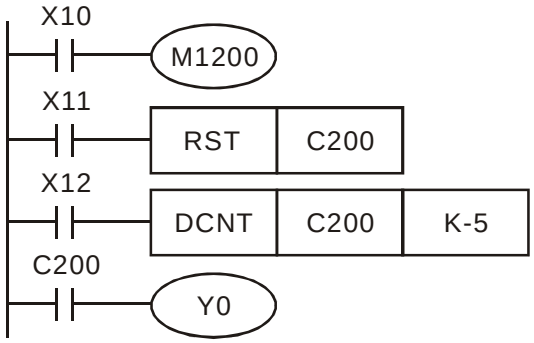
1. Диапазон 32-х разрядных счетчиков: К-2 147 483 648 ~ К2 147 483 647 (недоступны в контроллерах типов ES/EX/SS).
2. Режим работы счетчиков – сложение или вычитание – определяется состоянием специальных реле M1200 ~ M1234. Например, если M1200=0, то C200 будет складывать (счет вверх), если M1200=1, то C200 будет вычитать (счет вниз).
3. Уставку можно задавать константой К или регистром D (кроме специальных D1000 ~ D1999). Задавать можно задавать как положительные, так и отрицательные значения.

Так как счетчик 32-х разрядный, то значение уставки будет занимать 2 последовательных регистра.

4. В обычных счетчиках текущее значение будет обнулено при пропадании напряжения питания. В энергонезависимых счетчиках текущее значение и состояние контакта при пропадании питания будут сохранятся и при возобновлении питания счет продолжиться с текущего значения.
5. Когда текущее значение счетчика достигнет К2 147 483 647, следующим значением будет К-2 147 483 648. И наоборот, при достижении значения К-2 147 483 648, следующим значением будет К2 147 483 647.

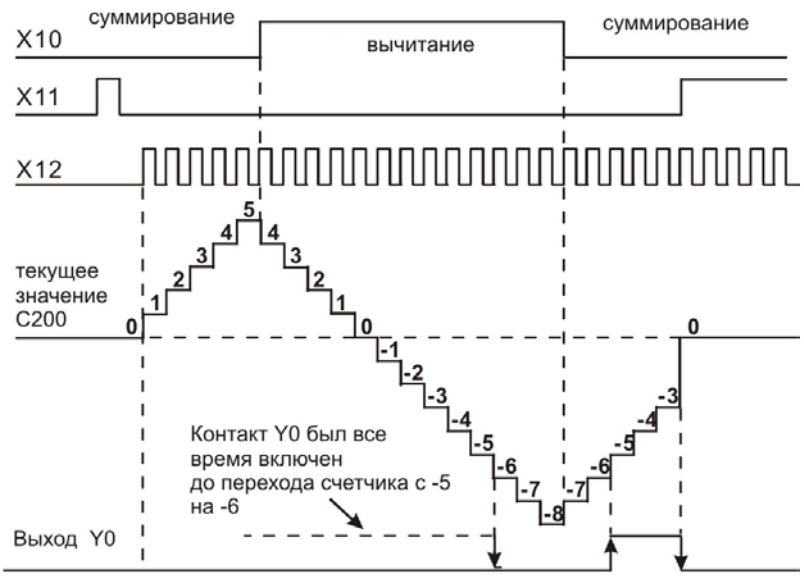
Пример работы 32-х разрядного счетчика

```
LD      X10
OUT     M1200
LD      X11
RST     C200
LD      X12
CNT     C200  K-5
LD      C200
OUT     Y0
```



1. X10 включает M1200, что определяет будет ли C200 суммировать или вычитать.

2. При включении X11 активируется команда RST и текущее значение C200 будет сброшено на 0, а контакт C200 выключен.
3. Уставка счетчика задана константой К-5.
(отрицательное значение).
4. При замыкании контакта X12 счетчик начнет подсчет входных импульсов: 1 импульс равен одному счету вверх или вниз в зависимости от состояние реле M1200.
5. Когда текущее значение счетчика перейдет с К-5 на К-6 контакт C200 и соответственно выход Y0 выключатся.
6. Когда текущее значение счетчика перейдет с К-6 на К-5 контакт снова включится. Т.е. счетчик будет включен при любом текущем значении не меньше К-5.



7. Если во время работы счетчика, который еще не достиг уставки, записать в регистр текущего значения счетчика величину большую, чем заданная уставка, то при появлении ближайшего входного импульса счетчик включит свой контакт и значение уставки автоматически станет равно текущему значению, которое Вы записали в регистр. Запись в регистр можно осуществить путем команды DMOV, программного пакета WPLSoft или программатора HPP.

Скоростные и высокоскоростные счетчики 32 бит, счет вверх/вниз, C235 ~ C255

1. Диапазон 32-х разрядных счетчиков: К-2 147 483 648 ~ К2 147 483 647.
2. Режим работы счетчиков C235 ~ C244 – сложение или вычитание – определяется состоянием специальных реле M1235 ~ M1244, а счетчиков C246 ~ C255 состоянием реле M1246 ~ M1255. Например, если M1235=0, то C235 будет складывать (счет вверх), если M1235=1, то C235 будет вычитать (счет вниз).
3. Уставку можно задавать константой К или регистром D (кроме специальных D1000 ~ D1999). Задавать можно задавать как положительные, так и отрицательные значения. Так как счетчик 32-х разрядный, то значение уставки будет занимать 2 последовательных регистра.
4. В режиме обычного счетчика текущее значение будет обнулено при пропадании напряжения питания. В режиме энергонезависимого счетчиках текущее значение и состояние контакта при пропадании питания будут сохранятся и при возобновлении питания счет продолжиться с текущего значения.
5. Когда текущее значение счетчика достигнет К2 147 483 647, следующим значением будет К-2 147 483 648. И наоборот, при достижении значения К-2 147 483 648, следующим значением будет К2 147 483 647.
6. Если во время работы счетчика, который еще не достиг уставки, записать в регистр текущего значения счетчика величину большую, чем заданная уставка, то при появлении ближайшего входного импульса счетчик не изменит свой контакт и продолжит счет с текущего значения.

Скоростные счетчики контроллеров типов ES/EX/SS

Общий диапазон скоростных счетчиков (если сложить частоту входных импульсов по всем входам) составляет максимум 20 кГц.

Тип	1 фаза 1 вход							1 фаза 2 входа			2 фазы 2 входа		
Вход	C235	C236	C237	C238	C241	C242	C244	C246	C247	C249	C251	C252	C254
X0	U/D				U/D		U/D	U	U	U	A	A	A
X1		U/D			R		R	D	D	D	B	B	B
X2			U/D			U/D			R	R		R	R
X3				U/D		R	S			S			S

U: Суммирование (счет вверх) A: Фаза A входа S: Разрешение счета
D: Вычитание (счет вниз) B: Фаза B входа R: Сброс на ноль

1. Физические входы X0 и X1 могут работать до частоты 20 кГц в режиме 1 фаза 1 вход. Однако, необходимо учитывать, что сумма входных частот данных 2-х входов не должна превышать 20 кГц (другие входы при этом не используются).
2. Физические входы X2 и X3 могут работать до частоты 10 кГц в режиме 1 фаза 1 вход.
3. В режиме двухфазного счетчика входная частота не должна превышать 4 кГц по любому из входов.
4. Использование в программе инструкции DHSCR не должно превышать 4-х раз.

Скоростные и высокоскоростные счетчики контроллеров типов SA/SX/SC

Общий диапазон скоростных счетчиков по входам X0 ~ X5 составляет максимум 40 кГц. Высокоскоростные входы X10 и X11 доступны только в контроллерах типа SC.

Тип	1 фаза 1 вход											1 фаза 2 входа				2 фазы 2 входа		
Вход	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245	C246	C247	C249	C250	C251	C252	C254
X0	U/D						U/D			U/D		U	U	U		A	A	A
X1		U/D					R			R		D	D	D		B	B	B
X2			U/D					U/D					R	R			R	R
X3				U/D				R		S				S				S
X4					U/D													
X5						U/D												
X10									U/D						U			
X11											U/D				D			

Примечание:
счетчики C253 и C255 в таблице не показаны

U:	Суммирование (счет вверх)	A:	Фаза A входа	S:	Разрешение счета
D:	Вычитание (счет вниз)	B:	Фаза B входа	R:	Сброс на ноль

1. Физические входы X0 и X1 могут работать до частоты 20 кГц в режиме 1 фаза 1 вход. Однако, необходимо учитывать, что сумма входных частот данных 2-х входов не должна превышать 40 кГц (другие входы при этом не используются).
2. Физические входы X2 ~ X5 могут работать до частоты 10 кГц в режиме 1 фаза 1 вход.
3. Максимальная частота двухфазных счетчиков C251, C252 и C254 составляет 4 кГц, а счетчика C253 до 25 кГц в режиме 4-х кратной частоты (в таблице не показан).
4. Вход X5 может работать в 2-х режимах:
   - когда M1260=0, C240 будет работать как обычный счетчик (U/D)
   - когда M1260=1 и инструкция DCNT активирует C240, вход X5 будет общей точкой сброса на ноль для счетчиков C235 ~ C239 (счетчик C240 будет продолжать получать сигналы с X5).
5. Входы X10 и X11 являются высокоскоростными и доступны только в контроллерах типа SC. Совокупный диапазон до 130 кГц. В однофазном режиме счетчики C243 (X10), C245 (X11) и C250 (X10, X11) могут работать до 100 кГц каждый по отдельности (но совокупно до 130 кГц).
6. Максимальная частота двухфазного счетчика C255 составляет 50 кГц (в таблице не показан).
7. Совокупное использование в программе инструкций DHSCS и DHSCR не должно превышать 6 раз. Использование инструкции DHSZ не должно превышать также 6 раз. Если при выполнении инструкции DHSCS в отношении нее сработала команда прерывания "I", то соответствующий счетчик не сможет далее исполнять инструкции DHSCS, DHSCR и DHSZ.
8. Входы X10 и X11 можно настроить на работу по восходящему и падающему фронту входного импульса. Для X10 это устанавливается в D1166, а для X11 в D1167: K0 – счет по восходящим фронтам импульсов, K1 – счет по нисходящим фронтам, K2 – счет по восходящим и нисходящим фронтам (доступно только для X10).
9. Режим работы счетчиков C243 и C245 (вверх или вниз) определяется специальными реле M1243 и M145 соответственно.
10. В C250 счет по восходящим или нисходящим фронтам определяется в D1166 (K0 или

К1).

11. Счетчик С255 может работать только в режиме 4-х кратной частоты и для него недоступен выбор восходящего или нисходящего фронта.
12. При использовании С243 и С245 нельзя использовать С250 и С255, и наоборот.

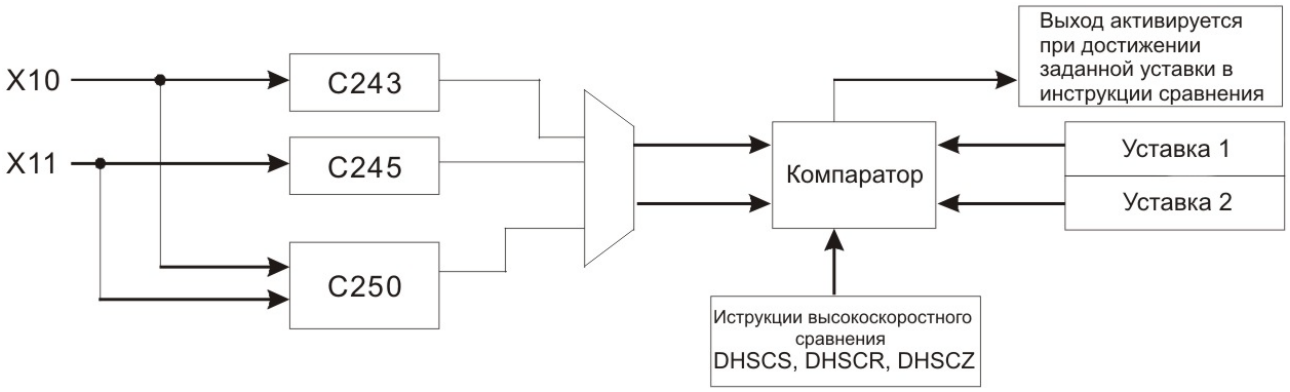
Использование высокоскоростных счетчиков с инструкциями высокоскоростного сравнения DHSCS, DHSCR и DHSCZ в контроллерах типа SC

Высокоскоростные счетчики С243/С245/С250/С255 контроллеров типа SC могут применяться с инструкциями высокоскоростного сравнения DHSCS и DHSCR только два раза в программе каждый, а с инструкцией DHSCZ только один раз. Т.е. каждому счетчику можно присваивать только две уставки, по достижении которых инструкции DHSCS и DHSCR будут активировать свой выход.

Также, один и тот же выход инструкций DHSCS и DHSCR может использоваться в программе только два раза.

Например, если в программе уже используется высокоскоростной счетчик для включения катушки Y10: DHSCS D0 C243 Y10, то для данной катушки можно использовать высокоскоростной счетчик еще только один раз DHSCR D2 C243 Y10 или DHSCS D4 C245 Y10.

Работа высокоскоростных счетчиков и инструкций сравнения схематично показана на рисунке ниже. Под блоком "Компаратор" понимается функция сравнения количества принятых импульсов от высокоскоростных счетчиков с заданной уставкой в инструкциях DHSCS и DHSCR с последующей активаций выхода (выходов), заданного в параметрах DHSCS и DHSCR.



Добавление высокоскоростных счетчиков (входы X10 и X11) не влияет на работу скоростных счетчиков (входы X0 ~ X5) с инструкциями высокоскоростного сравнения.

Инструкций DHSCZ может применяться с каждым счетчиком только один раз, а также один раз для одного и того же выхода.

Если для инструкции DHSCS требуется высокоскоростной выход, то необходимо использовать выходы Y10 и Y11, так как они обновляются не дожидаясь окончания скана программы, а обычные выходы будут иметь задержку на период одного полного скана программы.

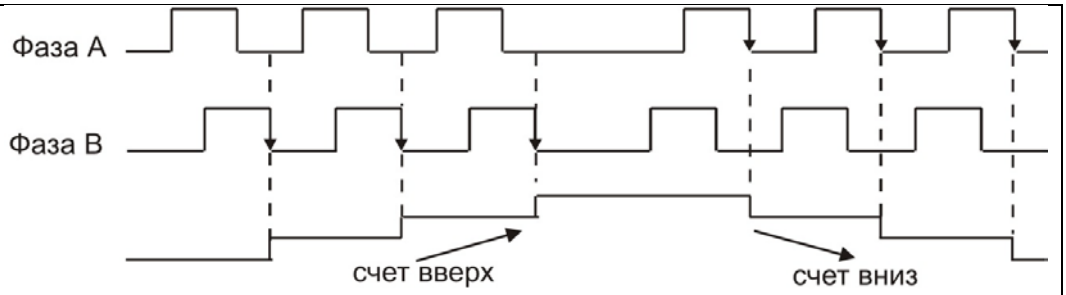
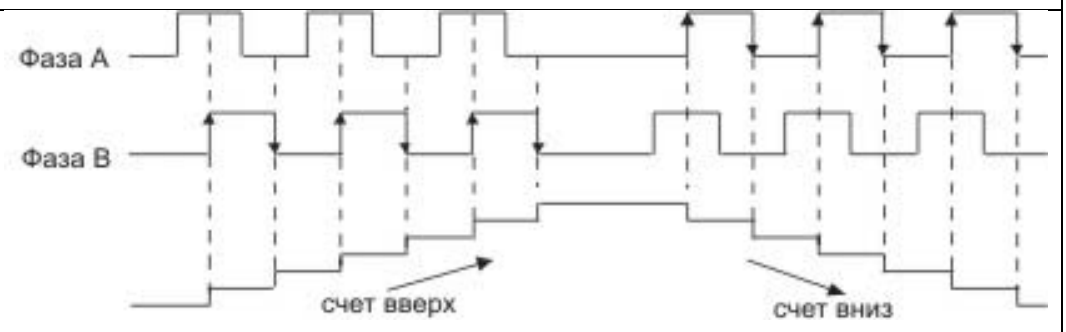
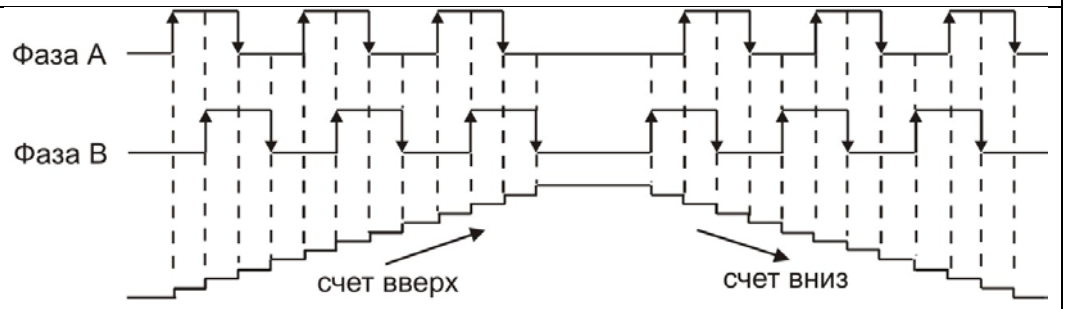
При использовании команды прерывания С243 будет соответствовать I020, С245 – I040, а С250 и С255 будет соответствовать I060.

Инструкция DHSCR может обнулять высокоскоростные счетчики, но только те, которые

используются непосредственно в самой инструкции. Например: DHSCR K10 C243 C243.

Режимы счета двухфазных счетчиков контроллеров ES/EX/SS/SA/SX/SC

Специальный регистр D1022 определяет режим работы скоростных двухфазных счетчиков – единичная частота (K1), двойная (K2 или K0, стоит по умолчанию) или четырехкратная (K4). Содержимое регистра загружается при переводе контроллера из режима СТОП в режим РАБОТА.
Ниже объясняется работа двухфазных счетчиков в каждом из режимов. Если опережает фаза "A", то счет идет вверх, если фаза "B", то счет идет вниз.

D1022 = 1	
D1022 = 2 или 0	
D1022 = 4	

1. При одинарной частоте счет идет только по заднему фронту импульса той фазы, которая в настоящий момент "опаздывает". Таким образом, значение в регистре счетчика увеличивается (уменьшается) на единицу при прохождении по каждой фазе одного входного импульса.
2. При двойной частоте счет идет по переднему и заднему фронту импульса той фазы, которая в настоящий момент "опаздывает". Таким образом, значение в регистре счетчика увеличивается (уменьшается) на два при прохождении по каждой фазе одного входного импульса.
3. При четырехкратной частоте счет идет по переднему и заднему фронту импульсов обоих фаз. Таким образом, значение в регистре счетчика увеличивается (уменьшается)

на четыре при прохождении по каждой фазе одного входного импульса.

Скоростные и высокоскоростные счетчики контроллеров типов EH/EH2/SV

Контроллеры типов EH/EH2/SV оснащены программными скоростными счетчиками C235 ~ C240 и работают в режиме 1 фаза – 1 вход с общим диапазоном 20 кГц. Максимально допустимая входная частота для отдельно взятого счетчика 10 кГц, но в совокупности суммарная частота всех задействованных счетчиков не должна превышать 20 кГц.
Также, контроллеры данных типов оснащены четырьмя аппаратными счетчиками, которые обозначаются как HHSC0, HHSC1, HHSC2 и HHSC3. В зависимости от режима работы они соответствуют следующим операндам:

- HHSC0 – C241, C246 и C251
- HHSC1 – C242, C247 и C252
- HHSC2 – C243, C248 и C253
- HHSC3 – C244, C249 и C254

Максимальная входная частота для HHSC0 и HHSC1 составляет 200 кГц для каждого независимо друг от друга. У аппаратных счетчиков HHSC2 и HHSC3 максимальная частота может быть 20 кГц (в однофазном и двухфазном режимах). У модели контроллера 40EH2 все четыре аппаратных счетчика могут работать с частотой до 200 кГц независимо друг от друга.
В рамках одной программы командой DCNT аппаратному счетчику может быть присвоен только один операнд (см. Таблицу ниже).

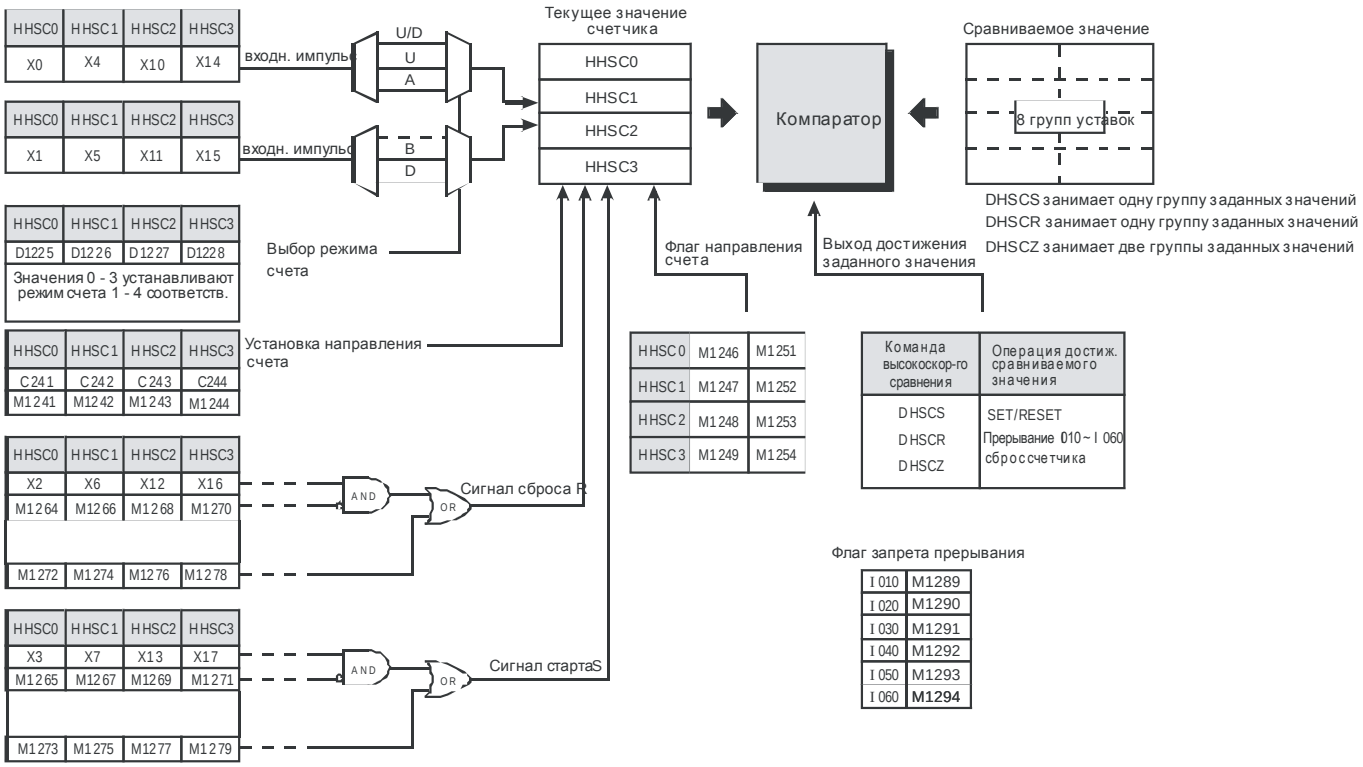
Тип	Программные высокоскоростные счетчики						Аппаратные высокоскоростные счетчики											
	1 фаза 1 вход						1 фаза 1 вход				1 фаза 2 входа				2 фазы 2 входа			
Вход	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C246	C247	C248	C249	C251	C252	C253	C254
X0	U/D						U/D				U				A			
X1		U/D									D				B			
X2			U/D				R				R				R			
X3				U/D			S				S				S			
X4					U/D			U/D				U				A		
X5						U/D						D				B		
X6								R				R				R		
X7								S				S				S		
X10									U/D				U				A	
X11													D				B	
X12									R				R				R	
X13									S				S				S	
X14										U/D				U				A
X15														D				B
X16										R				R				R
X17										S				S				S

U: Суммирование (счет вверх)
D: Вычитание (счет вниз)
A: Фаза A входа
B: Фаза B входа
S: Разрешение счета
R: Сброс на ноль

Описание аппаратных высокоскоростных счетчиков контроллеров типов EH/EH2/SV

- Аппаратные счетчики HHSC0 ~ HHSC3 включаются и сбрасываются на ноль внешними сигналами от физических входов (аппаратный старт и сброс).
- Также, существует возможность программного сброса счетчиков HHSC0 ~ HHSC3. Для этого необходимо отключить возможность аппаратного старта/сброса (см. пункт ниже), а затем путем включения специальных реле M1272, M1274, M1276 и M1278 можно сбросить соответствующий аппаратный счетчик HHSC0 ~ HHSC3 на ноль (нумерация последовательная). Включение специальных реле M1273, M1275, M1277 и M1279 запускает аппаратные счетчики HHSC0 ~ HHSC3 (нумерация последовательная). При использовании специальных реле для старта и сброса необходимо учитывать время скана программы, так как в данном случае внешние прерывания не действуют.
- Если для управления аппаратными счетчиками HHSC0 ~ HHSC3 не требуются внешние сигналы запуска и сброса на ноль, то включением специальных реле M1264, M1266, M1268 и M1270 можно дезактивировать функцию сброса от внешних входов, а включением реле M1265, M1267, M1269 и M1271 можно дезактивировать функцию запуска от внешних входов. Соответствующие входы в данном случае можно использовать как обычные физические входы X.

Ниже приводится общая блок-схема по настройке и работе аппаратных счетчиков, где указана взаимосвязь всех управляющий параметров, физических входов, специальных реле (флагов), операндов-счетчиков, а также функция "Компаратора" с использованием инструкций высокоскоростного сравнения DHSCS, DHSCR и DHSCZ.

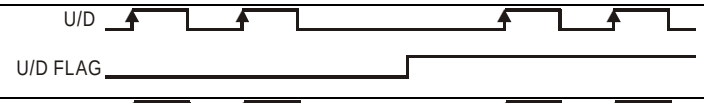
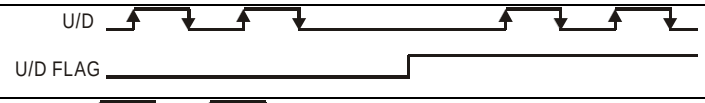
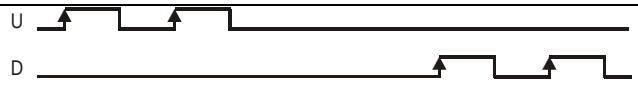
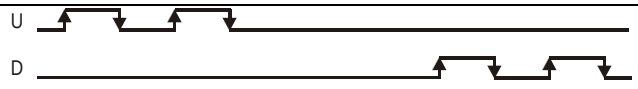
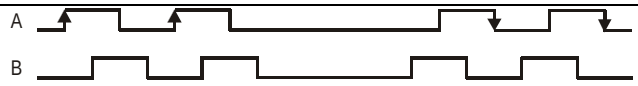
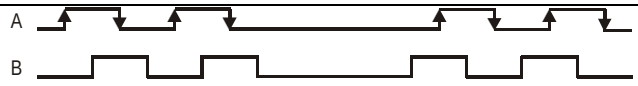
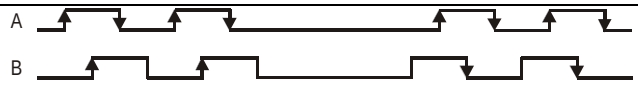
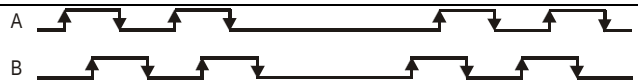


Аппаратные счетчики могут применяться с инструкциями высокоскоростного сравнения

DHSCS и DHSCR только четыре раза в программе каждый, а с инструкцией DHSCZ только два раза. Т.е. каждому счетчику можно присваивать только четыре уставки, по достижении которых инструкции DHSCS и DHSCR будут активировать свой выход.
Также, один и тот же выход инструкций DHSCS и DHSCR может использоваться в программе только четыре раза.

Режимы счета аппаратных счетчиков контроллеров EH/EH2/SV

Режим счета аппаратных счетчиков устанавливается в специальных регистрах D1225 ~ D1228 (см. сводную таблицу ниже).

Тип счетчика	Режим	Счет вверх (+1) Счет вниз (-1)
1 фаза 1 вход	однократный	
	двукратный	
1 фаза 2 входа	однократный	
	двукратный	
2 фазы 2 входа	однократный	
	двукратный	
	трехкратный	
	четырехкратный	

Комментарии.

1. 1 фаза 1 вход. В однократном режиме счет увеличивается (уменьшается) на единицу при появлении переднего фронта входного сигнала. Направление счета регулируется соответствующим флагом. В двукратном режиме счет увеличивается (уменьшается) на единицу по переднему и заднему фронту входного импульса, т.е. на один входной сигнал счет увеличивается на 2. Направление счета регулируется соответствующим флагом.
2. 1 фаза 2 входа. Режимы аналогичны предыдущему пункту, но сторона счета регулируется подачей сигнала на соответствующий физический вход.
3. 2 фазы 2 входа.
   - В однократном режиме счет вверх идет по переднему фронту опережающей фазы A, а счет вниз идет по заднему фронту фазы A, когда она "отстает" от фазы B.
   - В двукратном режиме добавляется при счете вверх добавляется задний фронт фазы A, а при счет вниз передний фронт. Таким образом, на 1 сигнал по каждой

фазе происходит два счета вверх или вниз (сигнал фазы B не учитывается).

- В трехкратном режиме добавляется еще 1 фронт фазы B, что дает 3 счета на 1 сигнал по каждой фазе.
- В четырехкратном режиме используются оба фронта обоих фаз. Таким образом, на 1 входной сигнал по каждой фазе осуществляется 4 счета вверх или вниз.

Сводная таблица специальных регистров и реле аппаратных счетчиков контроллеров EH/EH2/SV

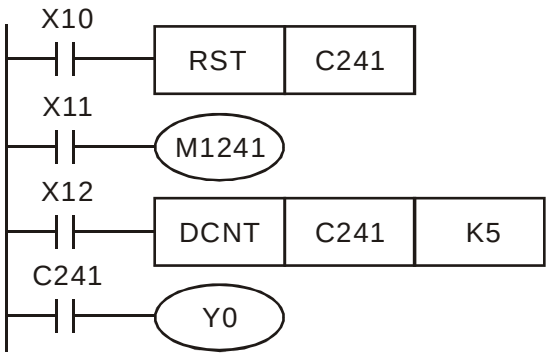
Номер	Функция
M1235 ~ M1244	Выбор направления счета для счетчиков C235 - C244 (0: суммирование; 1: вычитание)
M1246 ~ M1249 M1251 ~ M1254	Индикация направления счета счетчиков C246 – C249 и C251 – C254. (0: суммирование; 1: вычитание)
M1260	Определение входа X5 в качестве общего для сброса всех высокоскоростных счетчиков
M1264	Отключение функции сброса (R) счетчика HHSC0 от внешнего входа X2
M1265	Отключение функции запуска (S) счетчика HHSC0 от внешнего входа X3
M1266	Отключение функции сброса (R) счетчика HHSC1 от внешнего входа X6
M1267	Отключение функции запуска (S) счетчика HHSC1 от внешнего входа X7
M1268	Отключение функции сброса (R) счетчика HHSC2 от внешнего входа X12
M1269	Отключение функции запуска (S) счетчика HHSC2 от внешнего входа X13
M1270	Отключение функции сброса (R) счетчика HHSC3 от внешнего входа X16
M1271	Отключение функции запуска (S) счетчика HHSC3 от внешнего входа X17
M1272	Программный сброс (R) счетчика HHSC0 (M1272=1)
M1273	Программный запуск (S) счетчика HHSC0 (M1273=1)
M1274	Программный сброс (R) счетчика HHSC1 (M1274=1)
M1275	Программный запуск (S) счетчика HHSC1 (M1275=1)
M1276	Программный сброс (R) счетчика HHSC2 (M1276=1)
M1277	Программный запуск (S) счетчика HHSC2 (M1277=1)
M1278	Программный сброс (R) счетчика HHSC3 (M1278=1)
M1279	Программный запуск (S) счетчика HHSC3 (M1279=1)
M1289	Запрет прерывания высокоскоростного счетчика I010
M1290	Запрет прерывания высокоскоростного счетчика I020
M1291	Запрет прерывания высокоскоростного счетчика I030
M1292	Запрет прерывания высокоскоростного счетчика I040
M1293	Запрет прерывания высокоскоростного счетчика I050
M1294	Запрет прерывания высокоскоростного счетчика I060
M1312	Запуск счетчика C235
M1313	Запуск счетчика C236
M1314	Запуск счетчика C237
M1315	Запуск счетчика C238
M1316	Запуск счетчика C239
M1317	Запуск счетчика C240
M1320	Сброс счетчика C235
M1321	Сброс счетчика C236
M1322	Сброс счетчика C237
M1323	Сброс счетчика C238
M1324	Сброс счетчика C239
M1325	Сброс счетчика C240
M1328	Разрешение функции запуска/сброса счетчика C235
M1329	Разрешение функции запуска/сброса счетчика C236

M1330	Разрешение функции запуска/сброса счетчика C237
M1331	Разрешение функции запуска/сброса счетчика C238
M1332	Разрешение функции запуска/сброса счетчика C239
M1333	Разрешение функции запуска/сброса счетчика C240
D1225	Первая группа счетчиков (HHSC0). Счетные регистры: C241, C246, C251 Выбор режима
D1226	Вторая группа счетчиков (HHSC1). Счетные регистры: C242, C247, C252 Выбор режима
D1227	Третья группа счетчиков (HHSC2). Счетные регистры: C243, C248, C253 Выбор режима
D1228	Четвертая группа счетчиков (HHSC3). Счетные регистры: C244, C249, C254 Выбор режима
D1225 - D1228	Выбор режима счета для двухфазных высокоскоростных аппаратных счетчиков HHSC0 – HHSC3 контроллеров DVP-EH. 1: одинарная частота счета; 2: двойная частота (заводская уставка); 3: тройная частота; 4: четырехкратная частота счета.

Общие примеры применения высокоскоростных счетчиков

Пример 1.
Однофазный высокоскоростной счетчик с одним входом

```
LD      X10
RST     C241
LD      X11
OUT     M1241
LD      X12
DCNT    C241  K5
LD      C241
OUT     Y0
```



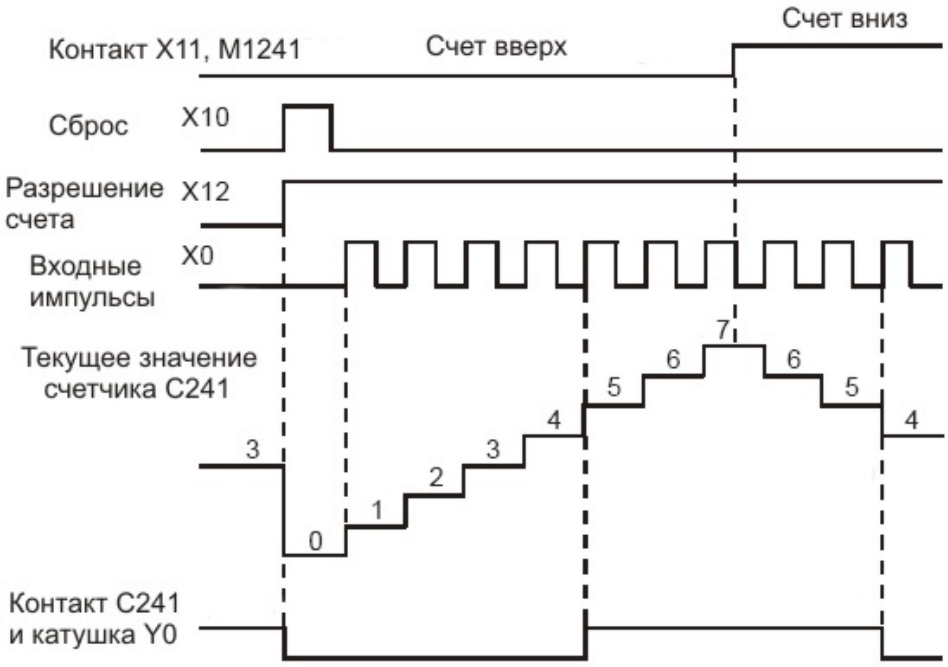
Комментарии

1. Специальное реле M1241 определяет направление счета и активируется входом X11.
2. При включении X10 активируется команда RST и текущее значение счетчика C241 сбросится в ноль, а контакт разомкнется (выключится).
3. Когда вход X12 активен счетчик C241 воспринимает сигналы от своего счетного входа X0 и с каждым импульсом текущее значение увеличивается (уменьшается) на 1.
4. При достижении счета заданной уставки K5 включится контакт C241 и, при наличии импульсов от X0, счет продолжится дальше.
5. В контроллерах ES/EX/SS/SA/SX/SC для сброса счетчика C241 можно использовать входной контакт X1.
6. В контроллерах EH/EH2/SV:
   - Для сброса счетчика C241 используется вход X2, а для запуска X3.
   - Функция сброса счетчика C241 (HHSC0) от внешнего входа (X2) отключается M1264, а функция старта от внешнего входа (X3) отключается M1265.
   - Программный сброс счетчика C241 (HHSC0) осуществляется M1272, а

разрешение на работу (счет входных импульсов) M1273 (возможность аппаратного сброса и старта должны быть отключены, см. предыдущий пункт).

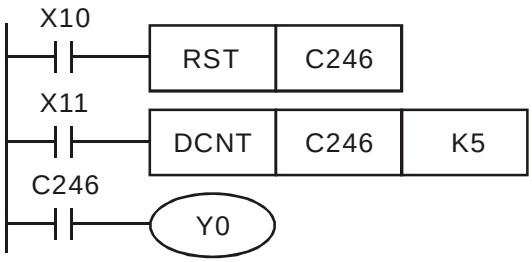
- Режим счета – одинарная или двойная частота – определяется в регистре D1225, по умолчанию стоит двойная частота (K2).

Ниже приведена временная диаграмма работы счетчика C241 в соответствии с указанным выше фрагментом программы.



Пример 2.
Однофазный высокоскоростной счетчик с двумя входами

```
LD      X10
RST     C246
LD      X11
DCNT    C246 K5
LD      C246
OUT     Y0
```



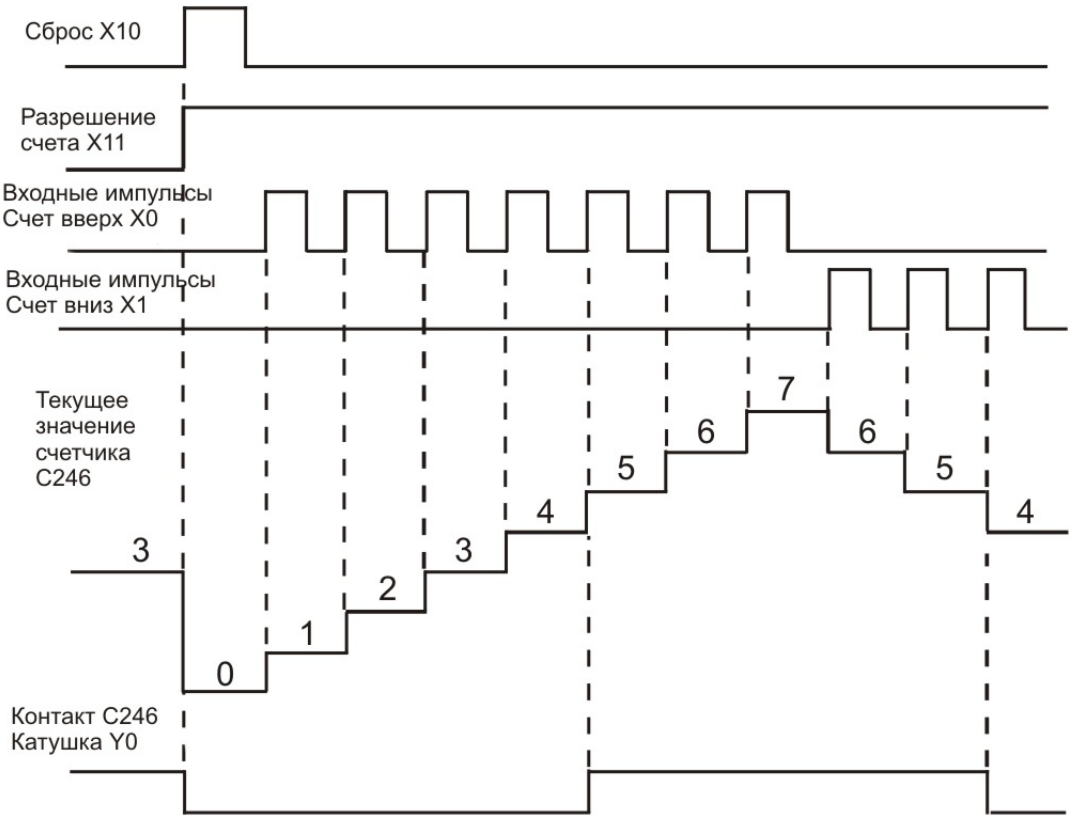
Комментарии

1. При включении X10 активируется команда RST и текущее значение счетчика C246 сбросится в ноль, а контакт разомкнется (выключится).
2. Когда вход X11 активен счетчик C246 воспринимает сигналы от своих счетных входов X0 (счет вверх) и X1 (счет вниз). С каждым импульсом текущее значение увеличивается или уменьшается на 1.
3. При достижении счета заданной уставки K5 включится контакт C246 и, при наличии

импульсов от X0 и X1, счет продолжится дальше.

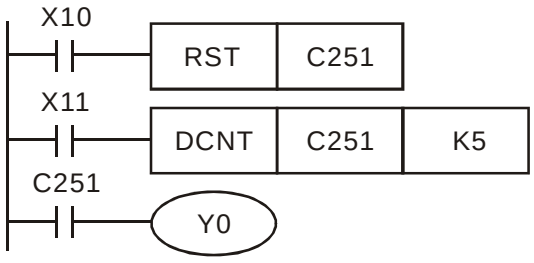
4. В контроллерах EH/EH2/SV:
   - Для сброса счетчика C246 используется вход X2, а для запуска X3.
   - Функция сброса счетчика C246 (HHSC0) от внешнего входа (X2) отключается M1264, а функция старта от внешнего входа (X3) отключается M1265.
   - Программный сброс счетчика C246 (HHSC0) осуществляется M1272, а разрешение на работу (счет входных импульсов) M1273 (возможность аппаратного сброса и старта должны быть отключены, см. предыдущий пункт).
   - Режим счета – одинарная или двойная частота – определяется в регистре D1225, по умолчанию стоит двойная частота (K2).

Ниже приведена временная диаграмма работы счетчика C246 в соответствии с указанным выше фрагментом программы.



Пример 3.
Двухфазный высокоскоростной счетчик с двумя входами

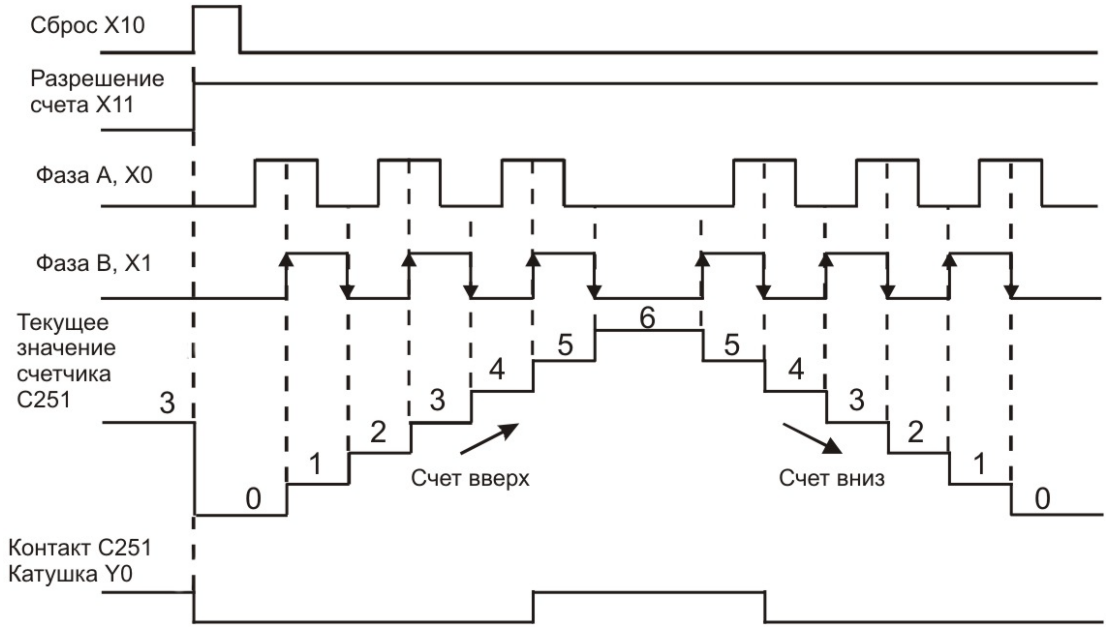
```
LD      X10
RST     C251
LD      X11
DCNT    C251 K5
LD      C251
OUT     Y0
```



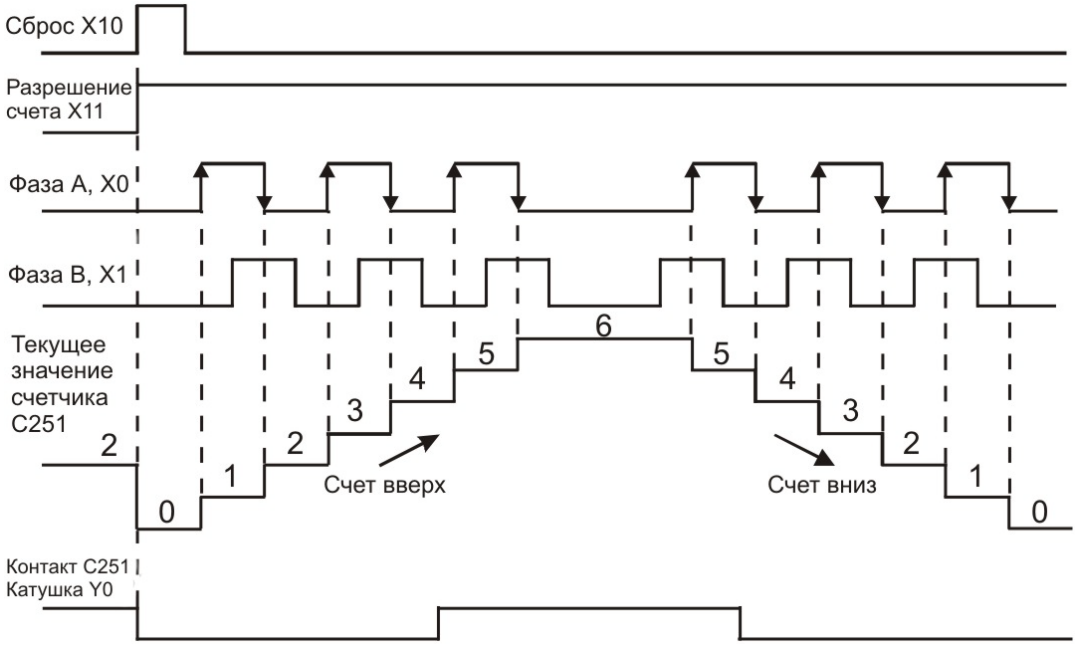
Комментарии

1. При включении X10 активируется команда RST и текущее значение счетчика C251 сбросится в ноль, а контакт разомкнется (выключится).
5. Когда вход X11 активен счетчик C251 воспринимает сигналы от своих счетных входов X0 (фаза A) и X1 (фаза B). С каждым импульсом текущее значение увеличивается или уменьшается на 1.
6. При достижении счета заданной уставки K5 включится контакт C251 и, при наличии импульсов от X0 и X1, счет продолжится дальше.
7. В контроллерах ES/EX/SS/SA/SX/SC режим счета – одинарная, двойная или четырехкратная частота – определяется в регистре D1022, по умолчанию стоит двойная частота (K2).
8. В контроллерах EH/EH2/SV:
   - Для сброса счетчика C251 используется вход X2, а для запуска X3.
   - Функция сброса счетчика C251 (HHSC0) от внешнего входа (X2) отключается M1264, а функция старта от внешнего входа (X3) отключается M1265.
   - Программный сброс счетчика C251 (HHSC0) осуществляется M1272, а разрешение на работу (счет входных импульсов) M1273 (возможность аппаратного сброса и старта должны быть отключены, см. предыдущий пункт).
   - Режим счета – одинарная, двойная, тройная или четырехкратная частота – определяется в регистре D1225, по умолчанию стоит двойная частота (K2).

Ниже приведена временная диаграмма работы счетчика C251 в соответствии с указанным выше фрагментом программы для контроллеров ES/EX/SS/SA/SX/SC, режим двойной частоты.



Временная диаграмма работы счетчика C251 в соответствии с указанным выше фрагментом программы для контроллеров EH/EH2/SV, режим двойной частоты.



2.8 Адресация и назначение регистров [D], [E], [F]

Регистры представляют память данных внутри ПЛК. В регистре можно хранить числовые значения и следующую бит за битом любую двоичную информацию.

Существует пять видов регистров:
1. Общие регистры. При переходе из режима РАБОТА в СТОП или отключении питания не сохраняют данные. При повторном включении ПЛК регистры данного типа будут пусты.
2. Энергонезависимые регистры. При переходе из режима РАБОТА в СТОП или отключении питания данные сохраняются. При повторном включении ПЛК ячейки данного типа будут содержать ранее записанные данные. Для очистки энергонезависимых регистров нужно использовать команды RST и ZRST.
3. Специальные регистры. Предназначены для хранения различной системной информации (текущие значения, настройки, коды ошибок и др.). Использовать для записи обычных данных пользователя категорически запрещается.
4. Индексные регистры E и F. Представляют из себя регистры 16 бит и предназначены для хранения добавочного индекса, который добавляется к адресу операнда при выполнении различных инструкций переноса и сравнения данных. Самостоятельно не применяются. Используются также для организации косвенной адресации.
5. Файловые регистры. Являются внутренней быстродействующей памятью процессора и могут также использоваться для хранения данных пользователя. Имеют разрядность 16 бит. Не имеют прямых адресов, поэтому обозначаются константами "K". Для чтения/записи файловых регистров применяются специальные инструкции API 148 MEMR и API 149 MEMW, а также программатор HPP или среда программирования WPLSoft.

Адресация регистров осуществляется в десятичном формате.

Тип ES/EX/SS

Регистры данных D	Общие	D0 ~ D407, 408 точек	Всего 744 точки
	Энергонезависимые	D408 ~ D599, 192 точки, фиксировано энергонезависимые	
	Специальные	D1000 ~ D1143, 144 точки	
	Индексные E, F	E(=D1028), F(=D1029), 2 точки	

Тип SA/SX/SC

Регистры данных D	Общие	D0 ~ D199, 200 точек, фиксировано энергозависимые	Всего 5000 точек
	Энергонезависимые	D200 ~ D999, D2000 ~ D4999, 3800 точек, по умолчанию энергонезависимые, могуть переопределены в общие	
	Специальные	D1000 ~ D1999, 1000 точек	
	Индексные E, F	E0 ~ E3, F0 ~ F3, 8 точек	
Файловые регистры		K0 ~ K1599, 1600 точек, фиксировано энергонезависимые	1600 точек

Тип EH/EH2/SV

Регистры данных D	Общие	D0 ~ D199, 200 точек, по умолчанию энергозависимые, могут быть переопределены в энергонезависимые	Всего 10000 точек
	Энергонезависимые	D200 ~ D999, D2000 ~ D9999, 8800 точек, по умолчанию энергонезависимые, могут быть переопределены в общие	
	Специальные	D1000 ~ D1999, 1000 точек	
	Индексные E, F	E0 ~ E7, F0 ~ F7, 16 точек	
Файловые регистры		K0 ~ K9999, 10000 точек, по умолчанию энергонезависимые	10000 точек

2.8.1 Регистры данных [D]

Регистры данных позволяют хранить 16-ти разрядное значение от -32768 до +32767. Самый старший бит хранит знак "+" или "-". Два последовательных регистра данных D и D+1 можно объединять в пары. Тогда разрядность повышается до 32 и можно записать значение от -2 147 483 648 до +2 147 483 647. Регистр с меньшим адресом является младшим и хранит 1-16 бит, регистр с адресом "младший адрес + 1" хранит биты 17-32. Например, регистры D10 (1-16 бит) и D11 (17-32 бит). Самый старший бит хранит информацию о знаке числа "+" или "-".

2.8.2 Индексные регистры [E], [F]

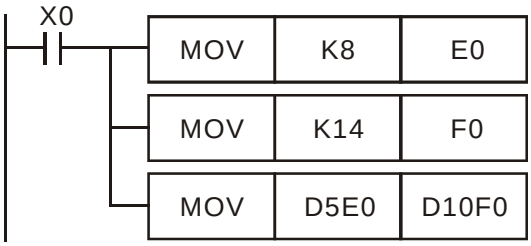
Индексные регистры имеют разрядность 16 бит. Если необходимо использовать индекс с разрядностью 32 бит, то индекс E и индекс F используются совместно. В индексе E будут хранится младшие 16 бит, а в индексе F будут хранится старшие 16 бит. Само 32-х разрядное значение записывается в индекс E, который при этом перекроет индекс F с таким же номером. В данном случае соответствующий индекс F будет уже не доступен. Комбинации 32-х разрядных индексных регистров будут следующие:

(F0, E0)
(F1, E1)
(F2, E2)
.....
(F7, E7)

Для обнуления 32-х разрядного регистра необходимо использовать команду DMOV K0 применительно к индексу E, при этом автоматически обнулиться и индекс F с тем же номером.

Пример использования индексных регистров для изменения адресации регистров D.

Когда X0 замкнется в регистр E0 запишется значение "8", в регистр F0 запишется значение "14". Далее произойдет суммирование адресов:
D5E0=D5 + E0= 5 + 8=13=D13
D10F0=D10 + F0= 10 + 14=24=D24
Таким образом, при текущих значениях индексов E и F произойдет запись содержимого регистра D13 в регистр D24.



Индексные регистры могут использоваться для операций передачи и сравнения данных совместно с байтовыми операндами (KnX, KnY, KnM, KnS, D, T, C) и битовыми операндами (X, Y, M, S).

В контроллерах серии EH можно индексировать также и константы (K, H).

При индексировании констант в командном режиме (IL, список инструкций) WPLSoft необходимо использовать символ @. Например: MOV K10@E0 D0F0.

Внимание!
При использовании индексных регистров для изменения адресов операндов категорически нельзя заходить в диапазон специальных регистров D1000 ~ D1999 и M1000 ~ M1999. В противном случае может произойти серьезная авария.

2.8.3 Файловые регистры

Файловые регистры представляют собой закрытую область памяти. Данные из файловых регистров нельзя использовать напрямую, их надо сначала переписать в обычные регистры данных D. Чтение/запись данных из регистров D в файловые регистры и обратно осуществляется или специальными инструкциями API 148 MEMR/API 149 MEMW при

включении соответствующего условия (см. соответствующий раздел), или автоматически при активации специального реле M1101.

В автоматическом режиме обмен данными с файловыми регистрами происходит следующим образом:

1. Проверяется включено ли реле M1101. Если включено, то проверяется содержимое регистров в следующих пунктах.
2. D1101 – начальный номер файлового регистра, с которого будет осуществляться чтение, задается константой K. Допустимый диапазон для ПЛК типов SA/SX/SC составляет K0 ~ K1600, для ПЛК типов EH/EH2/SV – K0 ~ K9999.
3. D1102 – количество читаемых файловых регистров. Допустимый диапазон для ПЛК типов SA/SX/SC составляет K0 ~ K1600, для ПЛК типов EH/EH2/SV – K0 ~ K8000.
4. D1103 – начальный адрес регистра данных D, начиная с которого будут записываться данные из файловых регистров. Задается константой K, которая должна быть не меньше 2000!

Данные в вышеупомянутые регистры заносятся командой MOV.

Внимание!

1. Если содержимое регистра D1101 превысит для контроллеров типов SA/SX/SC значение 1600, а для контроллеров типов EH/EH2/SV значение 8000, а также содержимое регистра D1103 окажется меньше 2000, то данные из файловых регистров не будет пересланы в регистры данных.
2. Если, при передаче данных диапазон адресов любых из регистров выйдет за пределы допустимого, контроллер прекратит передачу.
3. При попытке прочитать не существующий адрес файлового регистра, то значение в регистре данных D, куда предполагалось записать данные из файлового регистра, будет "0".

2.9 Назначение указателей [N], [P], [I]

Указатели используются в программе для обозначения шагов, по достижении которых должны произойти какие-либо действия. Указатели бывают трех видов: N – указатели номеров вложенности инструкций мастер-контроля, P – указатели перехода к подпрограмме или другому шагу программы, I – указатели перехода к подпрограмме прерываний.

Тип ES/EX/SS

Указатели	N	Для инструкций мастер-контроля MC/MCR		N0 ~ N7, 8 точек	Контрольная точка мастер-контроля
	P	Для инструкций CJ, CALL		P0 ~ P63, 64 точки	Указатель начала подпрограммы или перехода
	I	Прерывание	по времени	I6□□, 1 точка (□□=10 ~ 99 мс, дискретность = 1ms)	Указатель перехода к подпрограмме обработки прерывания
			внешние	I001, I101, I201, I301, 4 точки	
			коммуникационное	I150, 1 точка	

Тип SA/SX/SC

Указатели	N	Для инструкций мастер-контроля MC/MCR		N0~N7, 8 точек	Контрольная точка мастер-контроля
	P	Для инструкций CJ, CALL		P0~P255, 256 точек	Указатель начала подпрограммы или перехода
	I	Прерывание	внешние	I001, I101, I201, I301, I401, I501, 6 точек	Указатель перехода к подпрограмме обработки прерывания
			по времени	I6□□, I7□□, 2 точки (□□=10~99 мс, дискретность = 1 мс)	
			высокоскоростного счетчика	I010, I020, I030, I040, I050, I060, 6 точек	
			коммуникационное	I150, 1 точка	

Примечание:

В каждой из 6 пар прерываний (I001, I010), (I101, I1020), (I1201, I030), (I301, I040), (I401, I050), (I501, I060) можно использовать в программе какое-либо одно из них. Если использовать сразу оба, то может произойти синтаксическая ошибка.

Тип EH/EH2/SV

Указатели	N	Для инструкций мастер-контроля MC/MCR		N0~N7, 8 точек	Контрольная точка мастер-контроля
	P	Для инструкций CJ, CALL		P0~P255, 256 точек	Указатель начала подпрограммы или перехода
	I	Прерывание	внешние	I00□(X0), I10□(X1), I20□(X2), I30□(X3), I40□(X4), I50□(X5), 6 точек (□=1, по переднему фронту ┌┘, □=0, по заднему фронту └┐)	Указатель перехода к подпрограмме обработки прерывания
			по времени	I6□□, I7□□, I8□□, 2 точки (□□=1~99 мс, дискретность = 1 мс) I8□□, 1 точка (□□=0.1~9.9 мс, дискретность = 0.1 мс)	
			высокоскоростного счетчика	I010, I020, I030, I040, I050, I060, 6 точек	
			импульсного выхода	I110, I120, I130, I140, 4 точки	
			коммуникационное	I150, I160, I170, 3 точки	
			карты измерения частоты	I180, 1 точка	

Примечание:

Входы X, задействованные высокоскоростным счетчиком, не могут одновременно с ним использоваться для внешних прерываний. Например, если счетчик C251 занимает входы X0, X1, X2 и X3, то внешние прерывания I00□(X0), I10□(X1), I20□(X2), I30□(X3) нельзя использовать и они будут отключены.

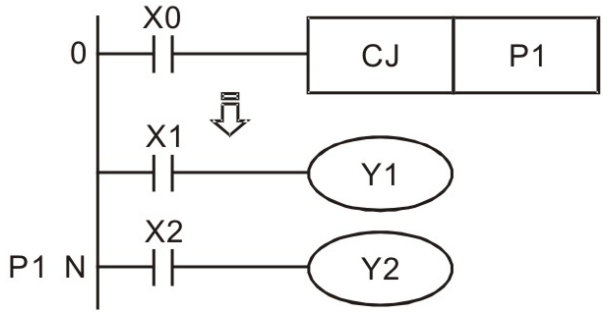
Указатели N

Используются с инструкциями мастер-контроля MC (старт мастер-контроля) MCR (сброс мастер-контроля). Использование мастер-контроля позволяет исключать участки программы из исполнения. Допускает до 8 уровней вложенности. См. также Главу 3.

Указатели P

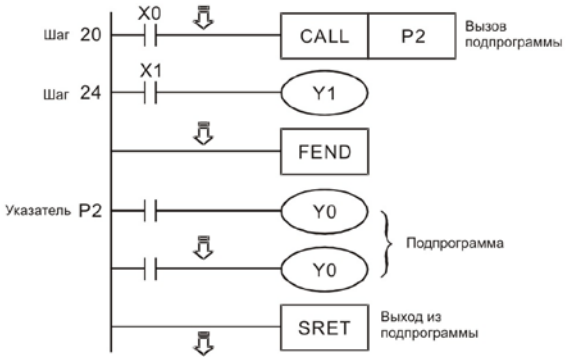
Применяются совместно с инструкциями API 00 CJ, API 01 CALL, API 02 SRET и используются в качестве номерных меток в программе куда должен осуществиться скачкообразный переход или откуда начинается подпрограмма. Указанные инструкции подробно описаны в Главе 6.

Пример использования указателя P с инструкцией CJ (условный скачкообразный переход).



На шаге "0" программы стоит контакт X0, включающий условный переход к указателю P1, находящемуся на шаге программы N.
Когда X0 замкнется, программа осуществит скачкообразный переход к метке P1, т.е. к сразу к шагу N, минуя участок с контактом X1 и катушкой Y1.
Если X0 разомкнут, то программ будет выполняться последовательно, включая участок с контактом X1 и катушкой Y1.

Пример использования указателя P с инструкциями CALL (вызов подпрограммы) и SRET (возвращение к основной программе).



При включении X0 (шаг 20) вызывается подпрограмма, отмеченная указателем P2, и запускается подпрограмма. По окончанию подпрограммы (SRET) процессор возвращается в основную программу на шаг 24 (следующий после строчки с командой CALL).
Если X0 не включен, то переход к подпрограмме не осуществляется.

Указатели I

Используются при организации прерываний в программе и переходу к подпрограмме обработки прерывания. Применяются совместно с инструкциями API 04 EI (активация прерывания), API 05 DI (дезактивация прерывания), API 03 IRET (выход из подпрограммы прерывания). Данные инструкции описаны в Главе 6.
Указатели прерываний "I" бывают 6-ти видов:

1. Внешние прерывания.

Благодаря специальной конструкции аппаратной части ЦПУ, входные сигналы по переднему или заднему фронту от клемм X0 ~ X5 воспринимаются напрямую, минуя время скана. Выполнение основной программы моментально останавливается и происходит переход к подпрограмме обработки прерывания по указателям I00□(X0), I10□(X1), I20□(X2), I30□(X3),

I40□(X4), I50□(X5). Когда исполнение подпрограммы достигнет инструкции IRET, произойдет возвращение к основной программе.
В контроллерах типов SA/SX импульсный вход X0 скоростных счетчиков C235, C251 и C253 работает с прерыванием I401, активирующимся от входа X4. Прерывание I401 позволяет перехватить текущее значения скоростного счетчика, работающего от X0, и записать 32-х битное значение в ячейки D1180/D1181. Подобным образом импульсный вход X1 работает со входом X5, через который активируется прерывание I501, перехватывается текущее значение скоростного счетчика C245 (вход X1) и записывается в ячейки D1198/D1199 (32 бит).
В контроллерах типа SC импульсный вход X10 высокоскоростных счетчиков C243 и C255 работает со входом X4 (внешнее прерывание I401). При активации I401 перехватывается текущее значение высокоскоростного счетчика и записывается в ячейки D1180/D1181 (32 бит). По аналогии вход X11 высокоскоростного счетчика C245 работает с входом X5 прерывания I501, которое перехватывает текущее значение и записывает в регистры D1198/D1199 (32 бит).

2. Прерывания по времени.
Контроллер будет прерывать исполнение основной программы через установленные промежутки времени и переходить к обозначенной указателем подпрограмме прерывания.

3. Прерывания по достижению высокоскоростным счетчиком заданной уставки.
Инструкция высокоскоростного сравнения API 53 DHSCS может быть настроена на таким образом, что при достижении заданной уставки будет осуществлять прерывание основной программы с переходом по указателям I010, I020, I030, I040, I050 или I060 к заданной подпрограмме.

4. Прерывания по импульсному выходу.
Инструкция импульсного выхода API 57 PLSY может быть настроена таким образом, что при выдаче первого импульса синхронно будет выдан сигнал на прерывание путем включения флагов M1342 и M1343, соответствующие прерываниям I130 и I140. Также, можно настроить инструкцию на выдачу сигнала прерывания по последнему импульсу. В данном случае включаются флаги M1340 и M1341, которым соответствуют прерывания I110 и I120.

5. Коммуникационные прерывания.
I150: Используется совместно с инструкцией API 80 RS и позволяет организовать прерывание при получении определенного слова по последовательному каналу связи. Данное слово записывается в младший байт регистра D1168, при получении которого ПЛК отработает подпрограмму прерывания по указателю I150.
I160: Работает также как и I150, но кодовое слово записывается в младший байт регистра D1169. Если в данном регистре записан "0", прерывание не будет отработано.

I170: Используется когда контроллер является Ведомым устройством (Slave) для организации немедленной обработки данных, полученных по последовательному каналу связи, не дожидаясь инструкции END. В обычном режиме полученные данные будут обработаны только в следующем скане, поэтому при длительном цикле программы может пройти существенный промежуток времени. Чтобы избежать этого, можно использовать прерывание I170, которое инициирует обработку данных сразу после окончания сеанса связи не дожидаясь конца скана (инструкции END).

6. Прерывание карты измерения частоты (только для EH/EH2).
I180: При выборе режимов 1 и 3 карты установкой M1019 и D1034 прерывание I180 поддерживается также.

2.10 Специальные регистры и реле

В контроллерах есть регистры и реле, которым жестко присвоены определенные функции. Их наличие существенно облегчает написание программ и работу с прикладными инструкциями. Пользователь получает удобные инструменты для осуществления определенных действий, оперативного получения информации и воздействия на процессы.
Специальные регистры и реле категорически нельзя использовать в программе в качестве регистров и реле общего назначения, так как это приведет к сбоям или непредсказуемым действиям контроллера.
Ниже в таблицах приведен полный список и функциональное назначение специальных регистров и реле. В следующем параграфе (2.11) приведено их описание и примеры применения.
В таблицах используются следующие условные обозначения:

- Значок "○" – реле или регистр присутствует в данном типе контроллеров, пустая ячейка – реле или регистр данным типом контроллеров не поддерживается
- Значок "#" – устанавливается системой в зависимости от состояния контроллера
- Значок "*" – данное реле или регистр более подробно описывается в параграфе 2.11
- Значок "-" – состояние не меняется
- Значок "R" – возможно только чтение текущего состояния
- Значок "R/W" – возможно и чтение и запись пользователем
- Значок "On" – Включается (в колонках реакции на изменение состояния контроллера)
- Значок "Off" – Выключается (в колонках реакции на изменение состояния контроллера)
- Колонки ES/EX/SS, SA/SX/SC, EH/SV (сюда относится и EH2) – типы контроллеров
- Колонка "Off —> On" – реакция на подачу питания на контроллер
- Колонка "STOP —> RUN" – реакция на перевод контроллера в состояние "Работа"
- Колонка "RUN —> STOP" – реакция на перевод контроллера в состояние "Стоп"
- Колонка "Энергонезависимость" – отображает, является ли реле или регистр энергонезависимым, т.е. сохраняющим свое текущее состояние при отключении питания, "да" – является, "нет" – не является
- Колонка "По умолчанию" – отображает исходное состояние реле или регистра, т.е. когда елемент еще не задействовался в программе, "On" – Включено, "Off" – Выключено или какое-либо цифровое значение

Специальные реле

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1000*	Нормально-открытый контакт. Контакт замкнут, когда на ПЛК подано напряжение питания, и он находится в состоянии RUN.	○	○	○	#	On	Off	R	нет	Off
M1001*	Нормально-закрытый контакт. Контакт разомкнут, когда на ПЛК подано напряжение питания, и он находится в состоянии RUN.	○	○	○	#	Off	On	R	нет	On
M1002*	Нормально-открытый контакт. Контакт замыкается при включении ПЛК во время первого цикла выполнения программы на период, равный периоду сканирования. Все остальное время контакт разомкнут.	○	○	○	#	On	Off	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1003*	Нормально-закрытый контакт. Контакт размыкается при включении ПЛК во время первого цикла выполнения программы на период, равный периоду сканирования. Все остальное время контакт замкнут.	○	○	○	#	Off	On	R	нет	On
M1004*	Замыкается при возникновении синтаксической ошибки.	○	○	○	Off	Off	-	R	нет	Off
M1005	Замыкается, когда пароль в карте памяти не совпадает с паролем в ПЛК.			○	Off	Off	-	R	нет	Off
M1006	Замыкается, когда карта памяти не инициализирована.			○	Off	Off	-	R	нет	Off
M1007	Замыкается, когда данные отсутствуют в области программы карты памяти.			○	Off	-	-	R	нет	Off
M1008*	Флаг сторожевого таймера. Включается, когда вышло время WDT.	○	○	○	Off	Off	-	R	нет	Off
M1009	Включается, когда уровень напряжение питания 24 VDC ниже допустимого (LV).	○	○	○	Off	-	-	R	нет	Off
M1010	Выбор режима выполнения инструкции PLSY для импульсного выхода. **ES/EX/SS/SA/SX/SC**: когда M1010=ON импульсы на Y0 идут непрерывно. **EH/EH2/SV**: когда M1010=ON импульсы на Y0, Y1, Y2, Y3 будут идти до выполнения инструкции END.	○	○	○	Off	-	-	R/W	нет	Off
M1011*	Генератор импульсов с периодом 10мс (ON= 5 мс, OFF=5 мс).	○	○	○	Off	-	-	R	нет	Off
M1012*	Генератор импульсов с периодом 100мс (ON= 50 мс, OFF=50 мс).	○	○	○	Off	-	-	R	нет	Off
M1013*	Генератор импульсов с периодом 1 сек (ON= 0.5 сек, OFF=0.5 сек).	○	○	○	Off	-	-	R	нет	Off
M1014*	Генератор импульсов с периодом 1 мин (ON= 30 сек, OFF = 30 сек).	○	○	○	Off	-	-	R	нет	Off
M1015*	Запуск высокоскоростного таймера.		○	○	Off	-	-	R/W	нет	Off
M1016*	Отображение года в часах реального времени: OFF – 2 цифры, ON – 4 цифры.		○	○	Off	-	-	R/W	нет	Off
M1017*	Корректировка секунд в часах реального времени (± 30 сек).		○	○	Off	-	-	R/W	нет	Off
M1018	Флаг: радианы/градусы. M1018 = ON – градусы.		○	○	Off	-	-	R/W	нет	Off
M1019	Флаг запуска работы карты измерения частоты.			○	Off	Off	-	R	нет	Off
M1020	Флаг нуля. Включается, если результат сложения или вычитания равен нулю.	○	○	○	Off	-	-	R	нет	Off
M1021	Флаг заимствования (Borrow). Включается, если результат вычитания меньше самого малого значения.	○	○	○	Off	-	-	R	нет	Off
M1022	Флаг переноса (Carry). Включается при передаче значения числа, при суммировании или при передаче данных, при выполнении инструкции сдвига.	○	○	○	Off	-	-	R	нет	Off
M1023	Выбор режима выполнения инструкции PLSY для импульсного выхода Y1. Когда M1023=ON импульсы идут непрерывно.	○	○		Off	-	-	R/W	нет	Off
M1024	Флаг запроса передачи на COM1.	○	○	○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1025	Когда к ПЛК подключено одно из устройств - HMI, HPP или PC, которое посылает ПЛК нештатный запрос, ПЛК установит M1025=ON и запишет код ошибки в регистр D1025.	○	○	○	Off	-	-	R	нет	Off
M1026	Флаг разрешения инструкции RAMP.		○	○	Off	-	-	R/W	нет	Off
M1027	Флаг PR выхода.		○	○	Off	-	-	R/W	нет	Off
M1028	Флаг режима работы таймеров T64...T126 M1028=ON – дискретность 10мс M1028=OFF - дискретность 100мс	○			Off	-	-	R/W	нет	Off
M1029*	Флаг завершения выполнения инструкции. В т.ч. PLSY и PLSR, а также многих других. ES, EX, SS и SA, SX, SC: Команда PLSY или PLSR для импульсного выхода Y0 полностью выполнена. EH/EH2/SV: Команда PLSY или PLSR для первой группы импульсных выходов CH0 (Y0, Y1) полностью выполнена.	○	○	○	Off	-	-	R	нет	Off
M1030*	Флаг завершения выполнения инструкций PLSY и PLSR, а также ряда других команд. ES, EX, SS и SA, SX, SC: Команда PLSY или PLSR для импульсного выхода Y1 полностью выполнена. EH/EH2/SV: Команда PLSY или PLSR для второй группы импульсных выходов CH1 (Y2, Y3) полностью выполнена.	○	○	○	Off	-	-	R	нет	Off
M1031*	Очистка всей энергозависимой памяти данных.	○	○	○	Off	-	-	R/W	нет	Off
M1032*	Очистка всей энергонезависимой памяти данных.	○	○	○	Off	-	-	R/W	нет	Off
M1033*	Фиксация текущего состояния физических выходов Y (катушек) при переводе контроллера в состояние СТОП.	○	○	○	Off	-	-	R/W	нет	Off
M1034*	Принудительное отключение всех физических выходов Y.	○	○	○	Off	-	-	R/W	нет	Off
M1035*	Разрешение входа X в качестве переключателя RUN/STOP. Номер входа X записывается в D1035 (для типа SA, только X7 может быть использован, для SX только X3, для SC X5).		○	○	-	-	-	R/W	да	Off
M1036*	EH2/SV: Команда PLSY или PLSR для третьей группы импульсных выходов CH2 (Y4, Y5) полностью выполнена.			○	Off	-	-	R	нет	Off
M1037*	EH2/SV: Команда PLSY или PLSR для четвертой группы импульсных выходов CH3 (Y6, Y7) полностью выполнена.			○	Off	-	-	R	нет	Off
M1039*	Включение режима фиксированного времени скана программы.	○	○	○	Off	-	-	R/W	нет	Off
M1040	Флаг запрета передачи управляющего шага. Когда M1040=1 останавливается выполнение шаговой последовательности.	○	○	○	Off	-	-	R/W	нет	Off
M1041	Инициация следующего шага. Флаг инструкции IST.	○	○	○	Off	-	Off	R/W	нет	Off
M1042	Разрешение импульсов. Флаг инструкции IST.	○	○	○	Off	-	-	R/W	нет	Off
M1043	Возвращение в нулевую точку завершено. Флаг инструкции IST.	○	○	○	Off	-	Off	R/W	нет	Off
M1044	Разрешение непрерывного режима работы. Флаг инструкции IST.	○	○	○	Off	-	Off	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1045	Запрещение сброса всех выходов. Флаг инструкции IST.	○	○	○	Off	-	-	R/W	нет	Off
M1046	M1046=1, если какой-либо из шагов включен (режим пошагового управления).	○	○	○	Off	-	-	R	нет	Off
M1047	Разрешение мониторинга выполнения режима пошагового управления (STL).		○	○	Off	-	-	R/W	нет	Off
M1048	Флаг аварийного состояния.		○	○	Off	-	-	R	нет	Off
M1049	Включение мониторинга аварийного состояния.		○	○	Off	-	-	R/W	нет	Off
M1050	Запрет прерывания I001.	○	○		Off	-	-	R/W	нет	Off
M1051	Запрет прерывания I101.	○	○		Off	-	-	R/W	нет	Off
M1052	Запрет прерывания I201.	○	○		Off	-	-	R/W	нет	Off
M1053	Запрет прерывания I301.	○	○		Off	-	-	R/W	нет	Off
M1054	Запрет прерывания I401.		○		Off	-	-	R/W	нет	Off
M1055	Запрет прерывания I501.		○		Off	-	-	R/W	нет	Off
M1056	Запрет прерывания I6□□.	○	○		Off	-	-	R/W	нет	Off
M1057	Запрет прерывания I7□□.		○		Off	-	-	R/W	нет	Off
M1059	Запрет прерываний I010 – I060.		○		Off	-	-	R/W	нет	Off
M1060	Системная ошибка, сообщение № 1.	○	○	○	Off	-	-	R	нет	Off
M1061	Системная ошибка, сообщение № 2.	○	○	○	Off	-	-	R	нет	Off
M1062	Системная ошибка, сообщение № 3.	○	○	○	Off	-	-	R	нет	Off
M1063	Системная ошибка, сообщение № 4.	○	○	○	Off	-	-	R	нет	Off
M1064	Некорректное использование операнда.	○	○	○	Off	Off	-	R	нет	Off
M1065	Синтаксическая ошибка.	○	○	○	Off	Off	-	R	нет	Off
M1066	Общая ошибка цикла программы.	○	○	○	Off	Off	-	R	нет	Off
M1067*	Ошибка алгоритма программы.	○	○	○	Off	Off	-	R	нет	Off
M1068*	Ошибка алгоритма программы зафиксирована в D1068.	○	○	○	Off	-	-	R	нет	Off
M1070	**ES/EX/SS/SA/SX/SC**: Команда PWM (ШИМ) для выхода Y1. M1070=OFF: дискретность задания = 1 мс M1070=ON: дискретность задания = 100 мкс. **EH/EH2/SV**: Команда PWM для 1-й группы импульсных выходов CH0 (Y0, Y1). M1070=OFF: дискретность задания = 1 мс M1070=ON: дискретность задания = 100 мкс.	○	○	○	Off	-	-	R/W	нет	Off
M1071	Команда PWM для 2-й группы импульсных выходов CH1 (Y2, Y3). M1071=OFF: дискретность задания = 1 мс M1071=ON: дискретность задания = 100 мкс.			○	Off	-	-	R/W	нет	Off
M1072	Команда RUN (запуск ПЛК) выполняется.	○	○	○	Off	On	Off	R/W	нет	Off
M1075*	Ошибка записи во Flash-память.			○	Off	-	-	R	нет	Off
M1076*	Ошибка часов реального времени.		○	○	Off	-	-	R	нет	Off
M1077	Низкое напряжение на встроенной батарее, отсутствие или выход из строя батареи.		○	○	Off	-	-	R	нет	Off
M1078	Немедленная остановка выполнения команды PLSY для Y0.	○	○		Off	-	-	R/W	нет	Off
M1079	Немедленная остановка выполнения команды PLSY для Y1.	○	○		Off	-	-	R/W	нет	Off
M1080	Флаг запроса на использование COM2.	○	○	○	Off	-	-	R	нет	Off
M1081	Флаг изменения направления преобразования инструкцией FLT.		○	○	Off	-	-	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1082	Флаг изменения значений в часах реального времени.		○	○	Off	-	-	R	нет	Off
M1083	Разрешение/запрещение выполнения прерываний в режиме FROM/TO.		○	○	Off	-	-	R/W	нет	Off
M1084*	Разрешение определения ширины импульса на входе X0.	○	○		Off	Off	Off	R/W	нет	Off
M1086	Установка переключателя ON/OFF для разрешения функции пароля карты DVP-PCC01	○	○	○	Off	-	-	R/W	нет	Off
M1087	Разрешение выдачи сигнала о низком напряжении (LV).			○	Off	-	-	R/W	нет	Off
M1088	Флаг операции сравнения матриц. M1088=1: матрицы одинаковые M1088=0: матрицы различные		○	○	Off	Off	-	R/W	нет	Off
M1089	Флаг конца сравнения матриц. Когда сравнение достигнет последнего бита, M1089=1.		○	○	Off	Off	-	R	нет	Off
M1090	Флаг начала сравнения матриц. Когда сравнение начнется с бита 0, M1090=1.		○	○	Off	Off	-	R	нет	Off
M1091	Флаг поиска нужного бита при сравнении матриц. Когда бит найден, сравнение сразу же остановится и M1091=1.		○	○	Off	Off	-	R	нет	Off
M1092	Флаг ошибки указателя матрицы. Если указатель Pr превышает заданный диапазон, M1092=1.		○	○	Off	Off	-	R	нет	Off
M1093	Флаг увеличения на 1 значения указателя матрицы. Когда M1093=1, к текущему значению указателя добавляется 1.		○	○	Off	Off	-	R/W	нет	Off
M1094	Флаг очистки значения указателя матрицы. Когда M1094=1, текущее значение указателя становится 0.		○	○	Off	Off	-	R/W	нет	Off
M1095	Флаг переноса при операциях сдвига/вращения матрицы.		○	○	Off	Off	-	R	нет	Off
M1096	Флаг дополнения при операции сдвига матрицы.		○	○	Off	Off	-	R/W	нет	Off
M1097	Флаг направления при операциях сдвига/вращения матрицы.		○	○	Off	Off	-	R/W	нет	Off
M1098	Флаг счетчика единичных или нулевых битов в матрице.		○	○	Off	Off	-	R/W	нет	Off
M1099	M1091=1, если результат подсчета битов матрицы равен 0.		○	○	Off	Off	-	R/W	нет	Off
M1100	Флаг величины выборки при выполнении команды SPD (вычисление скорости).			○	Off	-	-	R/W	нет	Off
M1101*	Разрешение на автоматическую передачу данных из файловых регистров (F) в обычные (D) при подаче питания на ПЛК.		○	○	-	-	-	R/W	Yes	Off
M1102	Флаг окончания выдачи импульсов выходом Y10 контроллера типа SC.		○		Off	-	-	R/W	нет	Off
M1103	Флаг окончания выдачи импульсов выходом Y11 контроллера типа SC.		○		Off	-	-	R/W	нет	Off
M1104*	Состояние DIP-переключателя SW1 функциональной карты DVP-F8ID, или входа AX0 карты дискретных входов DVP-F4IP.			○	Off	Off	-	R	нет	Off
M1105*	Состояние DIP-переключателя SW2 функциональной карты DVP-F8ID, или входа AX1 карты дискретных входов DVP-F4IP.			○	Off	Off	-	R	нет	Off
M1106*	Состояние DIP-переключателя SW3			○	Off	Off	-	R	нет	Off

Номер реле	Функция	ESEXSS	SASXSC	EHSV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
	функциональной карты DVP-F8ID, или входа AX2 карты дискретных входов DVP-F4IP.									
M1107*	Состояние DIP-переключателя SW4 функциональной карты DVP-F8ID, или входа AX3 карты дискретных входов DVP-F4IP.			○	Off	Off	-	R	нет	Off
M1108*	Состояние DIP-переключателя SW5 функциональной карты DVP-F8ID.			○	Off	Off	-	R	нет	Off
M1109*	Состояние DIP-переключателя SW6 функциональной карты DVP-F8ID.			○	Off	Off	-	R	нет	Off
M1110*	Состояние DIP-переключателя SW7 функциональной карты DVP-F8ID.			○	Off	Off	-	R	нет	Off
M1111*	Состояние DIP-переключателя SW8 функциональной карты DVP-F8ID.			○	Off	Off	-	R	нет	Off
M1112*	Состояние выхода AY0 карты транзисторных выходов DVP-F2OT.			○	Off	-	Off	R/W	нет	Off
M1113*	Состояние выхода AY1 карты транзисторных выходов DVP-F2OT.			○	Off	-	Off	R/W	нет	Off
M1115*	Запуск импульсного выхода Y0 с функцией разгона/замедления (недоступно в версии SC_V1.4 и выше).		○		Off	Off	Off	R/W	нет	Off
M1116*	Флаг режима разгона на импульсном выходе Y0 (недоступно в версии SC_V1.4 и выше).		○		Off	Off	Off	R/W	нет	Off
M1117*	Флаг достижения заданной частоты импульсным выходом Y0 (недоступно в версии SC_V1.4 и выше).		○		Off	Off	Off	R/W	нет	Off
M1118*	Флаг режима замедления на импульсном выходе Y0 (недоступно в версии SC_V1.4 и выше).		○		Off	Off	Off	R/W	нет	Off
M1119*	Флаг завершения цикла импульсным выходом Y0 с функцией разгона/торможения (недоступно в версии SC_V1.4 и выше).		○		Off	Off	Off	R/W	нет	Off
M1120	Фиксация протокола для порта COM2 (RS485). Когда M1120=1, изменения в D1120 контроллером не воспринимаются.	○	○	○	Off	Off	-	R/W	нет	Off
M1121	Флаг ожидания передачи данных по RS-485.	○	○	○	Off	On	-	R	нет	Off
M1122	Флаг запроса на передачу данных.	○	○	○	Off	Off	-	R/W	нет	Off
M1123	Флаг окончания приема данных.	○	○	○	Off	Off	-	R/W	нет	Off
M1124	Флаг ожидания приема.	○	○	○	Off	Off	-	R	нет	Off
M1125	Сброс связи.	○	○	○	Off	Off	-	R/W	нет	Off
M1126	Назначение стартового и стоповых символов в режиме ASCII инструкции RS (API-80) пользователем или системой (используются совместно M1126, M1130, D1124, D1125, D1126).	○	○	○	Off	Off	-	R/W	нет	Off
M1127	Завершение приема/передачи коммуникационным инструкциями, за исключением RS (API-80).	○	○	○	Off	Off	-	R/W	нет	Off
M1128	Флаг индикации процесса приема/передачи данных.	○	○	○	Off	Off	-	R	нет	Off
M1129	Флаг истечения времени ожидания приема данных.	○	○	○	Off	Off	-	R/W	нет	Off
M1130	Назначение стартового и стоповых символов в режиме ASCII инструкции RS (API-80) пользователем или системой (используются совместно M1126, M1130, D1124, D1125, D1126).	○	○	○	Off	Off	-	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1131	M1131=1, когда данные , полученные инструкциями MODRD/RDST/MODRW, конвертируются в HEX.	○	○	○	Off	Off	Off	R	нет	Off
M1132	Флаг отсутствия в ПЛК подходящей коммуникационной инструкции.	○	○	○	Off	-	-	R	нет	Off
M1133*	Запуск специального высокоскоростного выхода Y0 (50KHz). Начиная с версии SC_V1.4 и выше, разрешение выхода Y10 при синхронном управлении по 2 осям.		○		Off	Off	Off	R/W	нет	Off
M1134*	Включение режима непрерывной выдачи импульсов специальным высокоскоростным выходом Y0 (50KHz). (недоступно в версии SC_V1.4 и выше).		○		Off	Off	-	R/W	нет	Off
M1135*	Флаг достижения специальным высокоскоростным выходом Y0 (50KHz) заданного числа импульсов. Начиная с версии SC_V1.4 и выше, разрешение выхода Y11 при синхронном управлении по 2 осям.		○		Off	Off	Off	R/W	нет	Off
M1136*	Фиксация протокола связи для COM3.			○	Off	-	-	R/W	нет	Off
M1138*	Фиксация протокола для COM1 (RS-232). Когда M1138=1, изменения в D1036 контроллером не воспринимаются.	○	○	○	Off	-	-	R/W	нет	Off
M1139*	Выбор режима ASCII/RTU для COM1 (RS-232): M1139 = ON – режим RTU M1139 = OFF – режим ASCII	○	○	○	Off	-	-	R/W	нет	Off
M1140	Флаг ошибки при приеме данных инструкциями MODRD/MODWR/MODRW.	○	○	○	Off	Off	-	R	нет	Off
M1141	Флаг ошибки в параметрах инструкций MODRD/MODWR/MODRW.	○	○	○	Off	Off	-	R	нет	Off
M1142	Флаг ошибки приема данных от VFD-A.	○	○	○	Off	Off	-	R	нет	Off
M1143	Выбор режима ASCII/RTU для COM2 (RS-485): M1143 = ON – режим RTU M1143 = OFF – режим ASCII	○	○	○	Off	-	-	R/W	нет	Off
M1144*	Запуск функции выдачи заданного количества импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.		○		Off	Off	Off	R/W	нет	Off
M1145*	Флаг ускорения для импульсного выхода Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.		○		Off	Off	-	R	нет	Off
M1146*	Флаг достижения заданной частоты импульсным выходом Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.		○		Off	Off	-	R	нет	Off
M1147*	Флаг замедления для импульсного выхода Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.		○		Off	Off	-	R	нет	Off
M1148*	Флаг завершения цикла выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков импульсным выходом Y0.		○		Off	Off	Off	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1149*	Флаг временного прекращения подсчета импульсов на выходе Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.		○		Off	Off	-	R/W	нет	Off
M1150	Включение режима группового сравнения для инструкции DHSZ.			○	Off	-	-	R/W	нет	Off
M1151	Окончание выполнения режима группового сравнения для инструкцией DHSZ.			○	Off	-	-	R	нет	Off
M1152	Включение режима контроля частоты для инструкции DHSZ.			○	Off	-	-	R/W	нет	Off
M1153	Окончание выполнения режима контроля частоты инструкцией DHSZ.			○	Off	-	-	R	нет	Off
M1154*	Разрешение режима замедления для импульсного выхода Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.		○		Off	-	-	R/W	нет	Off
M1161	Режим 8/16 бит для коммуникационных инструкций (M1161 = 1 – 8 бит).	○	○	○	Off	-	-	R/W	нет	Off
M1162	Переключение между десятичным целом числом и двоичным с плавающей точкой для инструкции SCLP. M1162=1 двоичное с плавающей точкой M1162=0 целое десятичное	○	○	○	Off	-	-	R/W	нет	Off
M1165	Когда M1165=1, программа и пароль будут переписаны с flash-памяти в основную память при подаче питания на контроллер (недоступно в EH).			○	-	-	-	R/W	да	Off
M1166	Когда M1166=1, рецепты будут переписаны с flash-памяти в основную память контроллера при подаче питания на контроллер (недоступно в EH).			○	-	-	-	R/W	да	Off
M1167	Режим 16 бит для входа HKY.		○	○	Off	-	-	R/W	нет	Off
M1168	Назначение режима работы для инструкции SMOV.		○	○	Off	-	-	R/W	нет	Off
M1169	Назначение режима работы для инструкции PWD.			○	Off	-	-	R/W	нет	Off
M1170*	Разрешение функции пошагового исполнения программы.			○	Off	-	-	R/W	нет	Off
M1171*	Исполнение одного шага программы.			○	Off	-	-	R/W	нет	Off
M1172*	Включение 2-х фазного импульсного выхода, M1172=1 – включен.		○		Off	Off	Off	R/W	нет	Off
M1173*	Когда M1173=1, импульсы идут непрерывно.		○		Off	-	-	R/W	нет	Off
M1174*	Флаг достижения заданного количества выходных импульсов 2-х фазным выходом.		○		Off	Off	Off	R/W	нет	Off
M1175	Потеря контроллером параметров (недоступно в EH).			○	-	-	-	R	да	Off
M1176	Потеря контроллером данных в программе (недоступно в EH).			○	-	-	-	R	да	Off
M1178*	Разрешение функции потенциометра VR0.		○	○	Off	-	-	R/W	нет	Off
M1179*	Разрешение функции потенциометра VR1.		○	○	Off	-	-	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1184*	Разрешение функции модема (недоступно в SV).			○	Off	-	-	R/W	нет	Off
M1185*	Запуск инициализации модема (недоступно в SV).			○	Off	-	-	R/W	нет	Off
M1186*	Инициализация модема не удалась (недоступно в SV).			○	Off	-	-	R/W	нет	Off
M1187*	Инициализация модема прошла успешно (недоступно в SV).			○	Off	-	-	R/W	нет	Off
M1188*	Отображение статуса подключения модема (недоступно в SV). M1188=1, модем подключен.			○	Off	-	-	R/W	нет	Off
M1196	Выбор формата отображения данных на дисплее DVP-SX: M1196 = ON – шестнадцатеричный; M1196 = OFF – десятичный		○		Off	-	-	R/W	нет	Off
M1197	Установка десятичной точки на дисплее DVP-SX между 1 и 2 разрядом.		○		Off	-	-	R/W	нет	Off
M1198	Установка десятичной точки на дисплее DVP-SX в крайне правом положении (десятичный разряд на экране не отображается).		○		Off	-	-	R/W	нет	Off
M1200	Установка режима счета для счетчика C200. M1200=0 счет идет вверх (суммирование), M1200=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1201	Установка режима счета для счетчика C201. M1201=0 счет идет вверх (суммирование), M1201=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1202	Установка режима счета для счетчика C202. M1202=0 счет идет вверх (суммирование), M1202=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1203	Установка режима счета для счетчика C203. M1203=0 счет идет вверх (суммирование), M1203=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1204	Установка режима счета для счетчика C204. M1200=0 счет идет вверх (суммирование), M1200=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1205	Установка режима счета для счетчика C205. M1205=0 счет идет вверх (суммирование), M1205=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1206	Установка режима счета для счетчика C206. M1206=0 счет идет вверх (суммирование), M1206=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1207	Установка режима счета для счетчика C207. M1207=0 счет идет вверх (суммирование), M1207=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1208	Установка режима счета для счетчика C208. M1208=0 счет идет вверх (суммирование), M1208=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1209	Установка режима счета для счетчика C209. M1209=0 счет идет вверх (суммирование), M1209=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1210	Установка режима счета для счетчика C210. M1210=0 счет идет вверх (суммирование), M1210=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1211	Установка режима счета для счетчика C211. M1211=0 счет идет вверх (суммирование), M1211=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1212	Установка режима счета для счетчика C212. M1212=0 счет идет вверх (суммирование), M1212=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1213	Установка режима счета для счетчика C213. M1213=0 счет идет вверх (суммирование), M1213=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1214	Установка режима счета для счетчика C214. M1214=0 счет идет вверх (суммирование), M1214=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1215	Установка режима счета для счетчика C215. M1215=0 счет идет вверх (суммирование), M1215=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1216	Установка режима счета для счетчика C216. M1216=0 счет идет вверх (суммирование), M1216=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1217	Установка режима счета для счетчика C217. M1217=0 счет идет вверх (суммирование), M1217=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1218	Установка режима счета для счетчика C218. M1218=0 счет идет вверх (суммирование), M1218=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1219	Установка режима счета для счетчика C219. M1219=0 счет идет вверх (суммирование), M1219=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1220	Установка режима счета для счетчика C220. M1220=0 счет идет вверх (суммирование), M1220=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1221	Установка режима счета для счетчика C221. M1221=0 счет идет вверх (суммирование), M1221=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1222	Установка режима счета для счетчика C222. M1222=0 счет идет вверх (суммирование), M1222=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1223	Установка режима счета для счетчика C223. M1223=0 счет идет вверх (суммирование), M1223=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1224	Установка режима счета для счетчика C224. M1224=0 счет идет вверх (суммирование), M1224=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1225	Установка режима счета для счетчика C225. M1225=0 счет идет вверх (суммирование), M1225=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1226	Установка режима счета для счетчика C226. M1226=0 счет идет вверх (суммирование), M1226=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1227	Установка режима счета для счетчика C227. M1227=0 счет идет вверх (суммирование), M1227=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1228	Установка режима счета для счетчика C228. M1228=0 счет идет вверх (суммирование), M1228=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1229	Установка режима счета для счетчика C229. M1229=0 счет идет вверх (суммирование), M1229=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1230	Установка режима счета для счетчика C230. M1230=0 счет идет вверх (суммирование), M1230=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1231	Установка режима счета для счетчика C231. M1231=0 счет идет вверх (суммирование), M1231=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1232	Установка режима счета для счетчика C232. M1232=0 счет идет вверх (суммирование), M1232=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1233	Установка режима счета для счетчика C233. M1233=0 счет идет вверх (суммирование), M1233=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1234	Установка режима счета для счетчика C234. M1234=0 счет идет вверх (суммирование), M1234=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1235	Установка режима счета для счетчика C235. M1235=0 счет идет вверх (суммирование), M1235=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R/W	нет	Off
M1236	Установка режима счета для счетчика C236. M1236=0 счет идет вверх (суммирование), M1236=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R/W	нет	Off
M1237	Установка режима счета для счетчика C237. M1237=0 счет идет вверх (суммирование), M1237=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R/W	нет	Off
M1238	Установка режима счета для счетчика C238. M1238=0 счет идет вверх (суммирование), M1238=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R/W	нет	Off
M1239	Установка режима счета для счетчика C239. M1239=0 счет идет вверх (суммирование), M1239=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1240	Установка режима счета для счетчика C240. M1240=0 счет идет вверх (суммирование), M1240=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1241	Установка режима счета для счетчика C241. M1241=0 счет идет вверх (суммирование), M1241=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R/W	нет	Off
M1242	Установка режима счета для счетчика C242. M1242=0 счет идет вверх (суммирование), M1242=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R/W	нет	Off
M1243	Установка режима счета для счетчика C243. M1243=0 счет идет вверх (суммирование), M1243=1 счет идет вниз (вычитание).		○	○	Off	-	-	R/W	нет	Off
M1244	Установка режима счета для счетчика C244. M1244=0 счет идет вверх (суммирование), M1244=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R/W	нет	Off
M1245	Установка режима счета для счетчика C245. M1245=0 счет идет вверх (суммирование), M1245=1 счет идет вниз (вычитание).		○		Off	-	-	R/W	нет	Off
M1246	Установка режима счета для счетчика C246. M1246=0 счет идет вверх (суммирование), M1246=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R	нет	Off
M1247	Установка режима счета для счетчика C247. M1247=0 счет идет вверх (суммирование), M1247=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1248	Установка режима счета для счетчика C248. M1248=0 счет идет вверх (суммирование), M1248=1 счет идет вниз (вычитание).			○	Off	-	-	R	нет	Off
M1249	Установка режима счета для счетчика C249. M1249=0 счет идет вверх (суммирование), M1249=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R	нет	Off
M1250	Установка режима счета для счетчика C250. M1250=0 счет идет вверх (суммирование), M1250=1 счет идет вниз (вычитание).		○		Off	-	-	R	нет	Off
M1251	Установка режима счета для счетчика C251. M1251=0 счет идет вверх (суммирование), M1251=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R	нет	Off
M1252	Установка режима счета для счетчика C252. M1252=0 счет идет вверх (суммирование), M1252=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R	нет	Off
M1253	Установка режима счета для счетчика C253. M1253=0 счет идет вверх (суммирование), M1253=1 счет идет вниз (вычитание).			○	Off	-	-	R	нет	Off
M1254	Установка режима счета для счетчика C254. M1254=0 счет идет вверх (суммирование), M1254=1 счет идет вниз (вычитание).	○	○	○	Off	-	-	R	нет	Off
M1256	Флаг ошибки EF (внешнее отключение).			○	Off	Off	-	R	нет	Off
M1258	Реверсирование импульсного сигнала на Y0 для PWM-инструкции.			○	Off	-	-	R/W	нет	Off
M1259	Реверсирование импульсного сигнала на Y2 для PWM-инструкции.			○	Off	-	-	R/W	нет	Off
M1260	Разрешение контакта X5 в качестве общей точки сброса для всех высокоскоростных счетчиков.		○		Off	-	-	R/W	нет	Off
M1261	Флаг инструкции DHSCR для высокоскоростных счетчиков			○	Off	Off	Off	R/W	нет	Off
M1264	Отключение функции запуска (S) счетчика HHSC0 от внешнего входа X3.			○	Off	-	-	R/W	нет	Off
M1265	Отключение функции запуска (S) счетчика HHSC0 от внешнего входа X3.			○	Off	-	-	R/W	нет	Off
M1266	Отключение функции сброса (R) счетчика HHSC1 от внешнего входа X6.			○	Off	-	-	R/W	нет	Off
M1267	Отключение функции запуска (S) счетчика HHSC1 от внешнего входа X7.			○	Off	-	-	R/W	нет	Off
M1268	Отключение функции сброса (R) счетчика HHSC2 от внешнего входа X12.			○	Off	-	-	R/W	нет	Off
M1269	Отключение функции запуска (S) счетчика HHSC2 от внешнего входа X13.			○	Off	-	-	R/W	нет	Off
M1270	Отключение функции сброса (R) счетчика HHSC3 от внешнего входа X16.			○	Off	-	-	R/W	нет	Off
M1271	Отключение функции запуска (S) счетчика HHSC3 от внешнего входа X17.			○	Off	-	-	R/W	нет	Off
M1272	Программный сброс (R) счетчика HHSC0 (M1272=1).			○	Off	-	-	R/W	нет	Off
M1273	Программный запуск (S) счетчика HHSC0 (M1273=1).			○	Off	-	-	R/W	нет	Off
M1274	Программный сброс (R) счетчика HHSC1 (M1274=1).			○	Off	-	-	R/W	нет	Off
M1275	Программный запуск (S) счетчика HHSC1 (M1275=1).			○	Off	-	-	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1276	Программный сброс (R) счетчика HHSC2 (M1276=1).			○	Off	-	-	R/W	нет	Off
M1277	Программный запуск (S) счетчика HHSC2 (M1277=1).			○	Off	-	-	R/W	нет	Off
M1278	Программный сброс (R) счетчика HHSC3 (M1278=1).			○	Off	-	-	R/W	нет	Off
M1279	Программный запуск (S) счетчика HHSC3 (M1279=1).			○	Off	-	-	R/W	нет	Off
M1280	Запрет прерывания I00.			○	Off	-	-	R/W	нет	Off
M1281	Запрет прерывания I10.			○	Off	-	-	R/W	нет	Off
M1282	Запрет прерывания I20.			○	Off	-	-	R/W	нет	Off
M1283	Запрет прерывания I30.			○	Off	-	-	R/W	нет	Off
M1284	Запрет прерывания I40.			○	Off	-	-	R/W	нет	Off
M1285	Запрет прерывания I50.			○	Off	-	-	R/W	нет	Off
M1286	Запрет прерывания I6.			○	Off	-	-	R/W	нет	Off
M1287	Запрет прерывания I7.			○	Off	-	-	R/W	нет	Off
M1288	Запрет прерывания I8.			○	Off	-	-	R/W	нет	Off
M1289	Запрет прерывания высокоскоростного счетчика I010.			○	Off	-	-	R/W	нет	Off
M1290	Запрет прерывания высокоскоростного счетчика I020.			○	Off	-	-	R/W	нет	Off
M1291	Запрет прерывания высокоскоростного счетчика I030.			○	Off	-	-	R/W	нет	Off
M1292	Запрет прерывания высокоскоростного счетчика I040.			○	Off	-	-	R/W	нет	Off
M1293	Запрет прерывания высокоскоростного счетчика I050.			○	Off	-	-	R/W	нет	Off
M1294	Запрет прерывания высокоскоростного счетчика I060.			○	Off	-	-	R/W	нет	Off
M1295	Запрет прерывания I110.			○	Off	-	-	R/W	нет	Off
M1296	Запрет прерывания I120.			○	Off	-	-	R/W	нет	Off
M1297	Запрет прерывания I130.			○	Off	-	-	R/W	否	Off
M1298	Запрет прерывания I140.			○	Off	-	-	R/W	否	Off
M1299	Запрет прерывания I150.		○	○	Off	-	-	R/W	否	Off
M1300	Запрет прерывания I160.			○	Off	-	-	R/W	нет	Off
M1301	Запрет прерывания I170.			○	Off	-	-	R/W	нет	Off
M1302	Запрет прерывания I180.			○	Off	-	-	R/W	нет	Off
M1303	Перестановка местами старшего и младшего битов для инструкции XCH.		○	○	Off	-	-	R/W	нет	Off
M1304*	Разрешение программного вкл./выкл. входов X (WPLSoft или программатором HPP).		○	○	Off	-	-	R/W	нет	Off
M1305	Включение реверсивного режима первой группы импульсных выходов CH0 (Y0, Y1) для инструкций PLSV, DPLSV, DRVI, DDRVI, DRVA, DDRVA.			○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1306	Включение реверсивного режима первой группы импульсных выходов CH0 (Y0, Y1) для инструкций PLSV, DPLSV, DRVI, DDRVI, DRVA, DDRVA.			○	Off	-	-	R	нет	Off
M1310*	Немедленное отключение импульсного выхода Y10 (для версии SC_V1.4 и выше).		○		Off	Off	-	R/W	нет	Off
M1311*	Немедленное отключение импульсного выхода Y11 (для версии SC_V1.4 и выше).		○		Off	Off	-	R/W	нет	Off
M1312	Запуск счетчика C235			○	Off	-	-	R/W	нет	Off
M1313	Запуск счетчика C236			○	Off	-	-	R/W	нет	Off
M1314	Запуск счетчика C237			○	Off	-	-	R/W	нет	Off
M1315	Запуск счетчика C238			○	Off	-	-	R/W	нет	Off
M1316	Запуск счетчика C239			○	Off	-	-	R/W	нет	Off
M1317	Запуск счетчика C240			○	Off	-	-	R/W	нет	Off
M1320	Сброс счетчика C235			○	Off	-	-	R/W	нет	Off
M1321	Сброс счетчика C236			○	Off	-	-	R/W	нет	Off
M1322	Сброс счетчика C237			○	Off	-	-	R/W	нет	Off
M1323	Сброс счетчика C238			○	Off	-	-	R/W	нет	Off
M1324	Сброс счетчика C239			○	Off	-	-	R/W	нет	Off
M1325	Сброс счетчика C240			○	Off	-	-	R/W	нет	Off
M1328	Разрешение функции запуска/сброса счетчика C235			○	Off	-	-	R/W	нет	Off
M1329	Разрешение функции запуска/сброса счетчика C236			○	Off	-	-	R/W	нет	Off
M1330	Разрешение функции запуска/сброса счетчика C237			○	Off	-	-	R/W	нет	Off
M1331	Разрешение функции запуска/сброса счетчика C238			○	Off	-	-	R/W	нет	Off
M1332	Разрешение функции запуска/сброса счетчика C239			○	Off	-	-	R/W	нет	Off
M1333	Разрешение функции запуска/сброса счетчика C240			○	Off	-	-	R/W	нет	Off
M1334	**EH/EH2/SV:** Остановка первой группы импульсных выходов CH0 (Y0, Y1). **SC:** начиная с версии SC_V1.4 и выше – остановка импульсного выхода Y10.		○	○	Off	-	-	R/W	нет	Off
M1335	**EH/EH2/SV:** Остановка второй группы импульсных выходов CH1 (Y2, Y3). **SC:** начиная с версии SC_V1.4 и выше – остановка импульсного выхода Y11.		○	○	Off	-	-	R/W	нет	Off
M1336	Флаг окончания выдачи заданного количества импульсов первой группой CH0 (Y0, Y1).			○	Off	Off	Off	R	нет	Off
M1337	Флаг окончания выдачи заданного количества импульсов второй группой CH1 (Y2, Y3).			○	Off	Off	Off	R	нет	Off
M1338	Разрешение выдачи импульсов со смещением группой CH0 (Y0, Y1).			○	Off	-	-	R/W	нет	Off
M1339	Разрешение выдачи импульсов со смещением группой CH1 (Y2, Y3).			○	Off	-	-	R/W	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1340	Включение прерывания I110 сразу после выдачи импульсов группой CH0 (Y0, Y1).			○	Off	-	-	R/W	нет	Off
M1341	Включение прерывания I120 сразу после выдачи импульсов группой CH1 (Y2, Y3).			○	Off	-	-	R/W	нет	Off
M1342	Включение прерывания I130 одновременно с выдачей импульсов группой CH0 (Y0, Y1).			○	Off	-	-	R/W	нет	Off
M1343	Включение прерывания I140 одновременно с выдачей импульсов группой CH1 (Y2, Y3).			○	Off	-	-	R/W	нет	Off
M1344	Разрешение смещения группе CH0 (Y0, Y1).			○	Off	-	-	R/W	нет	Off
M1345	Разрешение смещения группе CH1 (Y2, Y3).			○	Off	-	-	R/W	нет	Off
M1346	Разрешение выходного сигнала "Очистить" (CLEAR) для инструкции ZRN.			○	Off	-	-	R/W	нет	Off
M1347	Разрешение авто-сброса первой импульсной группы CH0 (Y0, Y1) после выдачи заданного количества импульсов инструкцией PLSY.			○	Off	-	-	R/W	нет	Off
M1348	Разрешение авто-сброса второй импульсной группы CH1 (Y2, Y3) после выдачи заданного количества импульсов инструкцией PLSY.			○	Off	-	-	R/W	нет	Off
M1350*	Запуск цикличного обмена данными по коммуникационной технологии PLC LINK.		○	○	Off	-	-	R/W	нет	Off
M1351*	Запуск PLC LINK в автоматическом режиме.		○	○	Off	-	-	R/W	нет	Off
M1352*	Запуск PLC LINK в режиме с заданным количеством циклов опроса и внешним перезапуском.		○	○	Off	-	-	R/W	нет	Off
M1353*	Запуск PLC LINK в режиме до 32-х Ведомых станций и до 100 регистров записи/чтения в каждом Ведомом в одном цикле обмена данными.			○	Off	-	-	R/W	нет	Off
M1354*	Разрешение функции одновременного чтения и записи при работе PLC LINK.		○	○	Off	-	-	R/W	нет	Off
M1360*	PLC LINK SLAVE ID1, связь установлена.		○	○	Off	-	-	R	нет	Off
M1361*	PLC LINK SLAVE ID2, связь установлена.		○	○	Off	-	-	R	нет	Off
M1362*	PLC LINK SLAVE ID3, связь установлена.		○	○	Off	-	-	R	нет	Off
M1363*	PLC LINK SLAVE ID4, связь установлена.		○	○	Off	-	-	R	нет	Off
M1364*	PLC LINK SLAVE ID5, связь установлена.		○	○	Off	-	-	R	нет	Off
M1365*	PLC LINK SLAVE ID6, связь установлена.		○	○	Off	-	-	R	нет	Off
M1366*	PLC LINK SLAVE ID7, связь установлена.		○	○	Off	-	-	R	нет	Off
M1367*	PLC LINK SLAVE ID8, связь установлена.		○	○	Off	-	-	R	нет	Off
M1368*	PLC LINK SLAVE ID9, связь установлена.		○	○	Off	-	-	R	нет	Off
M1369*	PLC LINK SLAVE ID10, связь установлена.		○	○	Off	-	-	R	нет	Off
M1370*	PLC LINK SLAVE ID11, связь установлена.		○	○	Off	-	-	R	нет	Off
M1371*	PLC LINK SLAVE ID12, связь установлена.		○	○	Off	-	-	R	нет	Off
M1372*	PLC LINK SLAVE ID13, связь установлена.		○	○	Off	-	-	R	нет	Off
M1373*	PLC LINK SLAVE ID14, связь установлена.		○	○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1374*	PLC LINK SLAVE ID15, связь установлена.		○	○	Off	-	-	R	нет	Off
M1375*	PLC LINK SLAVE ID16, связь установлена.		○	○	Off	-	-	R	нет	Off
M1376*	PLC LINK SLAVE ID1, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1377*	PLC LINK SLAVE ID2, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1378*	PLC LINK SLAVE ID3, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1379*	PLC LINK SLAVE ID4, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1380*	PLC LINK SLAVE ID5, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1381*	PLC LINK SLAVE ID6, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1382*	PLC LINK SLAVE ID7, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1383*	PLC LINK SLAVE ID8, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1384*	PLC LINK SLAVE ID9, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1385*	PLC LINK SLAVE ID10, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1386*	PLC LINK SLAVE ID11, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1387*	PLC LINK SLAVE ID12, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1388*	PLC LINK SLAVE ID13, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1389*	PLC LINK SLAVE ID14, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1390*	PLC LINK SLAVE ID15, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1391*	PLC LINK SLAVE ID16, идет передача данных.		○	○	Off	-	-	R	нет	Off
M1392*	PLC LINK SLAVE ID1, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1393*	PLC LINK SLAVE ID2, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1394*	PLC LINK SLAVE ID3, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1395*	PLC LINK SLAVE ID4, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1396*	PLC LINK SLAVE ID5, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1397*	PLC LINK SLAVE ID6, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1398*	PLC LINK SLAVE ID7, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1399*	PLC LINK SLAVE ID8, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1400*	PLC LINK SLAVE ID9, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1401*	PLC LINK SLAVE ID10, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1402*	PLC LINK SLAVE ID11, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1403*	PLC LINK SLAVE ID12, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1404*	PLC LINK SLAVE ID13, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1405*	PLC LINK SLAVE ID14, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1406*	PLC LINK SLAVE ID15, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1407*	PLC LINK SLAVE ID16, ошибка чтения/записи.		○	○	Off	-	-	R	нет	Off
M1408*	PLC LINK SLAVE ID1, чтение завершено.		○	○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1409*	PLC LINK SLAVE ID2, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1410*	PLC LINK SLAVE ID3, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1411*	PLC LINK SLAVE ID4, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1412*	PLC LINK SLAVE ID5, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1413*	PLC LINK SLAVE ID6, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1414*	PLC LINK SLAVE ID7, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1415*	PLC LINK SLAVE ID8, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1416*	PLC LINK SLAVE ID9, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1417*	PLC LINK SLAVE ID10, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1418*	PLC LINK SLAVE ID11, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1419*	PLC LINK SLAVE ID12, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1420*	PLC LINK SLAVE ID13, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1421*	PLC LINK SLAVE ID14, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1422*	PLC LINK SLAVE ID15, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1423*	PLC LINK SLAVE ID16, чтение завершено.		○	○	Off	-	-	R	нет	Off
M1424*	PLC LINK SLAVE ID1, запись завершена.		○	○	Off	-	-	R	нет	Off
M1425*	PLC LINK SLAVE ID2, запись завершена.		○	○	Off	-	-	R	нет	Off
M1426*	PLC LINK SLAVE ID3, запись завершена.		○	○	Off	-	-	R	нет	Off
M1427*	PLC LINK SLAVE ID4, запись завершена.		○	○	Off	-	-	R	нет	Off
M1428*	PLC LINK SLAVE ID5, запись завершена.		○	○	Off	-	-	R	нет	Off
M1429*	PLC LINK SLAVE ID6, запись завершена.		○	○	Off	-	-	R	нет	Off
M1430*	PLC LINK SLAVE ID7, запись завершена.		○	○	Off	-	-	R	нет	Off
M1431*	PLC LINK SLAVE ID8, запись завершена.		○	○	Off	-	-	R	нет	Off
M1432*	PLC LINK SLAVE ID9, запись завершена.		○	○	Off	-	-	R	нет	Off
M1433*	PLC LINK SLAVE ID10, запись завершена.		○	○	Off	-	-	R	нет	Off
M1434*	PLC LINK SLAVE ID11, запись завершена.		○	○	Off	-	-	R	нет	Off
M1435*	PLC LINK SLAVE ID12, запись завершена.		○	○	Off	-	-	R	нет	Off
M1436*	PLC LINK SLAVE ID13, запись завершена.		○	○	Off	-	-	R	нет	Off
M1437*	PLC LINK SLAVE ID14, запись завершена.		○	○	Off	-	-	R	нет	Off
M1438*	PLC LINK SLAVE ID15, запись завершена.		○	○	Off	-	-	R	нет	Off
M1439*	PLC LINK SLAVE ID16, запись завершена.		○	○	Off	-	-	R	нет	Off
M1440*	PLC LINK SLAVE ID17, связь установлена.			○	Off	-	-	R	нет	Off
M1441*	PLC LINK SLAVE ID18, связь установлена.			○	Off	-	-	R	нет	Off
M1442*	PLC LINK SLAVE ID19, связь установлена.			○	Off	-	-	R	нет	Off
M1443*	PLC LINK SLAVE ID20, связь установлена.			○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1444*	PLC LINK SLAVE ID21, связь установлена.			○	Off	-	-	R	нет	Off
M1445*	PLC LINK SLAVE ID22, связь установлена.			○	Off	-	-	R	нет	Off
M1446*	PLC LINK SLAVE ID23, связь установлена.			○	Off	-	-	R	нет	Off
M1447*	PLC LINK SLAVE ID24, связь установлена.			○	Off	-	-	R	нет	Off
M1448*	PLC LINK SLAVE ID25, связь установлена.			○	Off	-	-	R	нет	Off
M1449*	PLC LINK SLAVE ID26, связь установлена.			○	Off	-	-	R	нет	Off
M1450*	PLC LINK SLAVE ID27, связь установлена.			○	Off	-	-	R	нет	Off
M1451*	PLC LINK SLAVE ID28, связь установлена.			○	Off	-	-	R	нет	Off
M1452*	PLC LINK SLAVE ID29, связь установлена.			○	Off	-	-	R	нет	Off
M1453*	PLC LINK SLAVE ID30, связь установлена.			○	Off	-	-	R	нет	Off
M1454*	PLC LINK SLAVE ID31, связь установлена.			○	Off	-	-	R	нет	Off
M1455*	PLC LINK SLAVE ID32, связь установлена.			○	Off	-	-	R	нет	Off
M1456*	PLC LINK SLAVE ID17, идет передача данных.			○	Off	-	-	R	нет	Off
M1457*	PLC LINK SLAVE ID18, идет передача данных.			○	Off	-	-	R	нет	Off
M1458*	PLC LINK SLAVE ID19, идет передача данных.			○	Off	-	-	R	нет	Off
M1459*	PLC LINK SLAVE ID20, идет передача данных.			○	Off	-	-	R	нет	Off
M1460*	PLC LINK SLAVE ID21, идет передача данных.			○	Off	-	-	R	нет	Off
M1461*	PLC LINK SLAVE ID22, идет передача данных.			○	Off	-	-	R	нет	Off
M1462*	PLC LINK SLAVE ID23, идет передача данных.			○	Off	-	-	R	нет	Off
M1463*	PLC LINK SLAVE ID24, идет передача данных.			○	Off	-	-	R	нет	Off
M1464*	PLC LINK SLAVE ID25, идет передача данных.			○	Off	-	-	R	нет	Off
M1465*	PLC LINK SLAVE ID26, идет передача данных.			○	Off	-	-	R	нет	Off
M1466*	PLC LINK SLAVE ID27, идет передача данных.			○	Off	-	-	R	нет	Off
M1467*	PLC LINK SLAVE ID28, идет передача данных.			○	Off	-	-	R	нет	Off
M14688	PLC LINK SLAVE ID29, идет передача данных.			○	Off	-	-	R	нет	Off
M1469*	PLC LINK SLAVE ID30, идет передача данных.			○	Off	-	-	R	нет	Off
M1470*	PLC LINK SLAVE ID31, идет передача данных.			○	Off	-	-	R	нет	Off
M1471*	PLC LINK SLAVE ID32, идет передача данных.			○	Off	-	-	R	нет	Off
M1472*	PLC LINK SLAVE ID17, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1473*	PLC LINK SLAVE ID18, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1474*	PLC LINK SLAVE ID19, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1475*	PLC LINK SLAVE ID20, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1476*	PLC LINK SLAVE ID21, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1477*	PLC LINK SLAVE ID22, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1478*	PLC LINK SLAVE ID23, ошибка чтения/записи.			○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1479*	PLC LINK SLAVE ID24, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1480*	PLC LINK SLAVE ID25, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1481*	PLC LINK SLAVE ID26, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1482*	PLC LINK SLAVE ID27, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1483*	PLC LINK SLAVE ID28, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1484*	PLC LINK SLAVE ID29, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1485*	PLC LINK SLAVE ID30, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1486*	PLC LINK SLAVE ID31, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1487*	PLC LINK SLAVE ID32, ошибка чтения/записи.			○	Off	-	-	R	нет	Off
M1488*	PLC LINK SLAVE ID17, чтение завершено.			○	Off	-	-	R	нет	Off
M1489*	PLC LINK SLAVE ID18, чтение завершено.			○	Off	-	-	R	нет	Off
M1490*	PLC LINK SLAVE ID19, чтение завершено.			○	Off	-	-	R	нет	Off
M1491*	PLC LINK SLAVE ID20, чтение завершено.			○	Off	-	-	R	нет	Off
M1492*	PLC LINK SLAVE ID21, чтение завершено.			○	Off	-	-	R	нет	Off
M1493*	PLC LINK SLAVE ID22, чтение завершено.			○	Off	-	-	R	нет	Off
M1494*	PLC LINK SLAVE ID23, чтение завершено.			○	Off	-	-	R	нет	Off
M1495*	PLC LINK SLAVE ID24, чтение завершено.			○	Off	-	-	R	нет	Off
M1496*	PLC LINK SLAVE ID25, чтение завершено.			○	Off	-	-	R	нет	Off
M1497*	PLC LINK SLAVE ID26, чтение завершено.			○	Off	-	-	R	нет	Off
M1498*	PLC LINK SLAVE ID27, чтение завершено.			○	Off	-	-	R	нет	Off
M1499*	PLC LINK SLAVE ID28, чтение завершено.			○	Off	-	-	R	нет	Off
M1500*	PLC LINK SLAVE ID29, чтение завершено.			○	Off	-	-	R	нет	Off
M1501*	PLC LINK SLAVE ID30, чтение завершено.			○	Off	-	-	R	нет	Off
M1502*	PLC LINK SLAVE ID31, чтение завершено.			○	Off	-	-	R	нет	Off
M1503*	PLC LINK SLAVE ID32, чтение завершено.			○	Off	-	-	R	нет	Off
M1504*	PLC LINK SLAVE ID17, запись завершена.			○	Off	-	-	R	нет	Off
M1505*	PLC LINK SLAVE ID18, запись завершена.			○	Off	-	-	R	нет	Off
M1506*	PLC LINK SLAVE ID19, запись завершена.			○	Off	-	-	R	нет	Off
M1507*	PLC LINK SLAVE ID20, запись завершена.			○	Off	-	-	R	нет	Off
M1508*	PLC LINK SLAVE ID21, запись завершена.			○	Off	-	-	R	нет	Off
M1509*	PLC LINK SLAVE ID22, запись завершена.			○	Off	-	-	R	нет	Off
M1510*	PLC LINK SLAVE ID23, запись завершена.			○	Off	-	-	R	нет	Off
M1511*	PLC LINK SLAVE ID24, запись завершена.			○	Off	-	-	R	нет	Off
M1512*	PLC LINK SLAVE ID25, запись завершена.			○	Off	-	-	R	нет	Off

Номер реле	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
M1513*	PLC LINK SLAVE ID26, запись завершена.			○	Off	-	-	R	нет	Off
M1514*	PLC LINK SLAVE ID27, запись завершена.			○	Off	-	-	R	нет	Off
M1515*	PLC LINK SLAVE ID28, запись завершена.			○	Off	-	-	R	нет	Off
M1516*	PLC LINK SLAVE ID29, запись завершена.			○	Off	-	-	R	нет	Off
M1517*	PLC LINK SLAVE ID30, запись завершена.			○	Off	-	-	R	нет	Off
M1518*	PLC LINK SLAVE ID31, запись завершена.			○	Off	-	-	R	нет	Off
M1519*	PLC LINK SLAVE ID32, запись завершена.			○	Off	-	-	R	нет	Off

Специальные регистры

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1000*	Текущее значение сторожевого таймера (WDT), ед.: 1 мс	○	○	○	200	-	-	R/W	нет	200
D1001	Версия встроенного программного обеспечения. Например, D1001 = HXX10 означает версию 1.0.	○	○	○	#	-	-	R	нет	#
D1002*	Максимальный объем программы (количество шагов).	○	○	○	#	-	-	R	нет	#
D1003	Занятый объем памяти ПЛК текущей загруженной программой.	○	○	○	#	-	-	R	нет	#
D1004*	Код синтаксической ошибки.	○	○	○	0	0	-	R	нет	0
D1008*	Адрес шага ошибки, когда сторожевой таймер WDT включен.	○	○	○	0	-	-	R	нет	0
D1009	ES/EX/SS/SA/SX/SC: количество сигналов о низком напряжении (LV) накопительным итогом. EH/EH2/SV: код ошибки потери данных в SRAM	○	○		-	-	-	R	да	0
D1010*	Текущее время скана загруженной программы. ед.: 0.1 мс.	○	○	○	0	-	-	R	нет	0
D1011*	Минимальное время скана загруженной программы. ед.: 0.1 мс.	○	○	○	0	-	-	R	нет	0
D1012*	Максимальное время скана загруженной программы. ед.: 0.1 мс.	○	○	○	0	-	-	R	нет	0
D1015*	Текущее значение высокоскоростного таймера, диапазон 0~32767, ед.: 0.1 мс.		○	○	0	-	-	R/W	нет	0
D1018*	Число π PI, младший байт.	○	○	○	H0FDB	H0FDB	H0FDB	R/W	нет	H0FDB
D1019*	Число π PI, старший байт.	○	○	○	H4049	H4049	H4049	R/W	нет	H4049
D1020*	Входной фильтр для входов X0~X7, ед.: мс.	○	○	○	10	-	-	R/W	нет	10
D1021*	Входной фильтр для входов X10~X17, ед.: мс.	○	○	○	10	-	-	R/W	нет	10
D1022	Коэффициент умножения частоты двухфазных счетчиков.	○	○		0	-	-	R/W	нет	0
D1023*	Значение ширины текущего импульса на входе X0, ед. 0,1 мс. Диапазон: 0,1 мс ~ 10000 мс.	○	○		0	-	-	R/W	нет	0
D1025*	Код ошибки коммуникационного запроса.	○	○	○	0	-	-	R	нет	0
D1028	Индексный регистр E0.	○	○	○	0	-	-	R/W	нет	0
D1029	Индексный регистр F0.	○	○	○	0	-	-	R/W	нет	0
D1030	Число выходных импульсов для Y0 накопительным итогом при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков (младшее слово).		○		0	-	-	R	нет	0
D1031	Число выходных импульсов для Y0 накопительным итогом при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков (старшее слово).		○		0	-	-	R	нет	0
D1032	Число выходных импульсов для Y1 (младшее слово).	○	○		0	-	-	R	нет	0

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1033	Число выходных импульсов для Y1 (старшее слово).	○	○		0	-	-	R	нет	0
D1034	Режим работы карты измерения частоты.			○	-	-	-	R	да	1
D1035*	Назначение номера входного контакта X для перевода контроллера RUN/STOP.			○	-	-	-	R/W	да	0
D1036*	Параметры протокола для COM1 (RS-232).	○	○	○	H86	-	-	R/W	нет	H86
D1037	Время сканирования для HKY (мс).			○	-	-	-	R/W	да	0
D1038*	Время задержки для ответа, когда ПЛК – ведомый в сети RS-485 (ед.: 0.1 мс)	○	○	○	-	-	-	R/W	да	0
D1039*	Значение фиксированного времени скана (мс).	○	○	○	0	-	-	R/W	нет	0
D1040	Номер реле Sn предыдущего шага 1 (режим STL).	○	○	○	0	-	-	R	нет	0
D1041	Номер реле Sn предыдущего шага 2 (режим STL).	○	○	○	0	-	-	R	нет	0
D1042	Номер реле Sn предыдущего шага 3 (режим STL).	○	○	○	0	-	-	R	нет	0
D1043	Номер реле Sn предыдущего шага 4 (режим STL).	○	○	○	0	-	-	R	нет	0
D1044	Номер реле Sn предыдущего шага 5 (режим STL).	○	○	○	0	-	-	R	нет	0
D1045	Номер реле Sn предыдущего шага 6 (режим STL).	○	○	○	0	-	-	R	нет	0
D1046	Номер реле Sn предыдущего шага 7 (режим STL).	○	○	○	0	-	-	R	нет	0
D1047	Номер реле Sn предыдущего шага 8 (режим STL).	○	○	○	0	-	-	R	нет	0
D1049	Номер включившегося сигнала аварии.		○	○	0	-	-	R	нет	0
D1050 ↓ D1055	Используются инструкциями Modbus в режиме ASCII. Данные, полученные в регистры D1070~D1085, будут автоматически конвертироваться в HEX и сохраняться в D1050~D1055.	○	○	○	0	-	-	R	нет	0
D1056*	Текущее значение аналогового входа CH0 EX/SX и карты расширения EH/EH2.	○	○	○	0	-	-	R	нет	0
D1057*	Текущее значение аналогового входа CH1 EX/SX и карты расширения EH/EH2.	○	○	○	0	-	-	R	нет	0
D1058*	Текущее значение аналогового входа CH2 EX.	○			0	-	-	R	нет	0
D1059*	Текущее значение аналогового входа CH3 EX.	○			0	-	-	R	нет	0
D1061	Сообщение о системной ошибке.	○	○		-	-	-	R	да	0
D1062*	Количество замеров при осреднении значения на аналоговых входах контроллеров SX. Диапазон 2 ~ 4 замера.		○		2	-	-	R/W	нет	2
D1067*	Код ошибки алгоритма.	○	○	○	0	0	-	R	нет	0
D1068*	Фиксация шага программы с ошибкой. Сброс реле RST M1068.	○	○	○	0	-	-	R	нет	0
D1069	Номер шага программы с ошибкой, связанной с флагами M1065~M1067.	○	○	○	0	-	-	R	нет	0
D1070 ↓ D1085	Используются инструкциями Modbus для сохранения принятых данных (ответного сообщения). В режиме ASCII данные будут сохранены в кодах ASCII, в режиме RTU в HEX.	○	○	○	0	-	-	R	нет	0

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1086	Старшее слово пароля для DVP-PCC01. Отображается в HEX в соответствии с символами ASCII.	○	○	○	0	-	-	R/W	нет	0
D1087	Младшее слово пароля для DVP-PCC01. Отображается в HEX в соответствии с символами ASCII.	○	○	○	0	-	-	R/W	нет	0
D1088	Установка количества копий для DVP-PCC01.	○	○	○	0	-	-	R/W	нет	0
D1089 ↓ D1099	Используются инструкциями Modbus для сохранения текущего отправленного сообщения.	○	○	○	0	-	-	R	нет	0
D1100	При появлении сигнала пониженного напряжения (LV) выходы Y0~Y17 отработают в соответствии со значением в D1100.			○	0	-	-	R/W	нет	0
D1101*	Начальный номер файлового регистра, с которого будет осуществляться чтение.		○	○	-	-	-	R/W	да	0
D1102*	Количество читаемых файловых регистров.		○	○	-	-	-	R/W	да	1600
D1103*	Начальный адрес регистра данных D, начиная с которого будут записываться данные из файловых регистров (адрес с D2000 и выше).		○	○	-	-	-	R/W	да	2000
D1104*	Начальный регистр для задания параметров импульсного выхода Y0 с функцией разгона/замедления (недоступно в версии SC_V1.4 и выше).		○		0	-	-	R/W	нет	0
D1109*	Параметры протокола для COM3.			○	0	-	-	R/W	нет	0
D1110*	Усредненное значение на аналоговом входе CH0 EX/SX и функциональной карты EH/EH2.	○	○	○	0	-	-	R	нет	0
D1111*	Усредненное значение на аналоговом входе CH1 EX/SX и функциональной карты EH/EH2.	○	○	○	0	-	-	R	нет	0
D1112*	Усредненное значение на аналоговом входе CH2 EX.	○			0	-	-	R	нет	0
D1113*	Усредненное значение на аналоговом входе CH3 EX.	○			0	-	-	R	нет	0
D1116*	Текущее значение на аналоговом выходе CH0 EX/SX и функциональной карты EH/EH2.	○	○	○	0	0	0	R/W	нет	0
D1117*	Текущее значение на аналоговом выходе CH1 EX/SX и функциональной карты EH/EH2.	○	○	○	0	0	0	R/W	нет	0
D1118*	SX/EX/EH время дискретизации АЦП (мс).	○	○	○	5	-	-	R/W	нет	5
D1120	Параметры протокола для COM2 (RS-485).	○	○	○	H86	-	-	R/W	нет	H86
D1121	Коммуникационный адрес контроллера (в сети Modbus).	○	○	○	-	-	-	R/W	да	1
D1122	Остаточные слова после передачи данных.	○	○	○	0	0	0	R	нет	0
D1123	Остаточные слова после приема данных.	○	○	○	0	0	0	R	нет	0
D1124	Определение стартового символа (STX) для режима ASCII.	○	○	○	H3A	-	-	R/W	нет	H3A
D1125	Определение первого стопового символа (ETX1) для режима ASCII.	○	○	○	H0D	-	-	R/W	нет	H0D
D1126	Определение второго стопового символа (ETX2) для режима ASCII.	○	○	○	H0A	-	-	R/W	нет	H0A
D1127	Слово, по получению которого по инструкции RS, произойдет прерывание I150.	○			0	-	-	R/W	нет	0
D1129	Предельное время ожидания ответа для RS-485	○	○	○	0	-	-	R/W	нет	0

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
	(мс).									
D1130	Код ошибки в ответном сообщении MODBUS.	○	○	○	0	-	-	R	нет	0
D1133*	SA/SX: начальный регистр для задания параметров импульсного выхода Y0 (50 кГц). SC_V1.4 и выше: начальный регистр для задания параметров импульсного выхода Y10, 2-х осевого синхронного управления.		○		0	-	-	R/W	нет	0
D1134*	Число участков для импульсного выхода Y10, 2-х осевого синхронного управления.		○		0	-	-	R/W	нет	0
D1135*	SC_V1.4 и выше: начальный регистр для задания параметров импульсного выхода Y11, 2-х осевого синхронного управления.		○		0	-	-	R/W	нет	0
D1136*	Число участков для импульсного выхода Y11, 2-х осевого синхронного управления.		○		0	-	-	R/W	нет	0
D1137*	Шаг программы, где выявлено некорректное использование операнда.	○	○	○	0	0	-	R	нет	0
D1140*	Количество присоединенных правосторонних модулей расширения (макс. 8).	○	○	○	0	-	-	R	нет	0
D1142	Количество входов X модулей расширения.	○	○	○	0	-	-	R	нет	0
D1143	Количество выходов Y модулей расширения.	○	○	○	0	-	-	R	нет	0
D1144*	Начальный адрес регистра D для параметров импульсного выхода Y0 при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков.		○		0	-	-	R/W	нет	0
D1145*	Количество присоединенных левосторонних модулей расширения (только для SV, макс. 8).			○	0	-	-	R	нет	0
D1147	Состояние флэш-карты памяти b0=0: карты нет (H0000) b0=1: флэш-карта определена (H0001) b8=0: выключатель флэш-карты выкл. (H0001) b8=1: выключатель флэш-карты вкл. (H0101)			○	#	-	-	R	нет	0
D1149	Тип карты расширения EH/EH2: 0: нет карты, 1: RS-232, DU-01, 2: RS-422, 3: COM3, 4: потенциометры, 5: DIP-переключатель, 6: транзисторные выходы, 8: 2 аналоговых входа AD, 9: 2 аналоговых выхода DA, 10: измерения частоты			○	#	-	-	R	нет	0
D1150	Таблица счета инструкции сравнения DHSZ в мультигруппном режиме.			○	0	0	0	R	нет	0
D1151	Таблица счета инструкции сравнения DHSZ в режиме измерения частоты.			○	0	0	0	R	нет	0
D1152	Старшее слово текущего значения инструкции сравнения DHSZ.			○	0	0	0	R	нет	0
D1153	Младшее слово текущего значения инструкции сравнения DHSZ.			○	0	0	0	R	нет	0

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1154*	Рекомендованный интервал шага времени замедления (10 ~ 32767 ms) импульсного выхода Y0 при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков, и когда M1154=1.		○		200	-	-	R/W	нет	200
D1155*	Рекомендованный интервал шага частоты замедления (-1 ~ -32700 ms) импульсного выхода Y0 при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков, и когда M1154=1.		○		-1000	-	-	R/W	нет	-1000
D1156 ↓ D1165	Регистры для инструкций RTMU, RTMD (K0~K9).			○	0	-	-	R/W	нет	0
D1166	SC: выбор режима для входа X10 – работа по переднему или заднему фронту.		○		0	-	-	R/W	нет	0
D1167	SC: выбор режима для входа X11 – работа по переднему или заднему фронту.		○		0	-	-	R/W	нет	0
D1168	Слово, по получении которого по инструкции RS, произойдет прерывание I150.		○	○	0	-	-	R/W	нет	0
D1169	Длина сообщения, по получении которого по инструкции RS, произойдет прерывание I160.			○	0	-	-	R/W	нет	0
D1170*	Номер текущего исполняемого шага программы в режиме пошагового выполнения программы.			○	0	0	0	R	нет	0
D1172*	Частоты двухфазного импульсного выхода (12 Гц ~ 20 кГц)		○		0	-	-	R/W	нет	0
D1173*	Выбор режима двухфазного импульсного выхода (K1 или K2).		○		0	-	-	R/W	нет	0
D1174*	Младшие 16 бит заданного числа импульсов двухфазного выхода.		○		0	-	-	R/W	нет	0
D1175*	Старшие 16 бит заданного числа импульсов двухфазного выхода.		○		0	-	-	R/W	нет	0
D1176*	Младшие 16 бит текущего числа импульсов двухфазного выхода.		○		0	-	-	R/W	нет	0
D1177*	Старшие 16 бит текущего числа импульсов двухфазного выхода.		○		0	-	-	R/W	нет	0
D1178*	Текущее значение потенциометра VR0.		○	○	0	-	-	R	нет	0
D1179*	Текущее значение потенциометра VR1.		○	○	0	-	-	R	нет	0
D1180*	Младшие 16 бит мгновенного значения высокоскоростного счетчика, полученного при срабатывании прерывания I401.		○		0	-	-	R	нет	0
D1181*	Старшие 16 бит мгновенного значения высокоскоростного счетчика, полученного при срабатывании прерывания I401.		○		0	-	-	R	нет	0
D1182	Индексный регистр E1		○	○	0	-	-	R/W	нет	0
D1183	Индексный регистр F1		○	○	0	-	-	R/W	нет	0
D1184	Индексный регистр E2		○	○	0	-	-	R/W	нет	0
D1185	Индексный регистр F2		○	○	0	-	-	R/W	нет	0
D1186	Индексный регистр E3		○	○	0	-	-	R/W	нет	0
D1187	Индексный регистр F3		○	○	0	-	-	R/W	нет	0

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1188	Индексный регистр E4			○	0	-	-	R/W	нет	0
D1189	Индексный регистр F4			○	0	-	-	R/W	нет	0
D1190	Индексный регистр E5			○	0	-	-	R/W	нет	0
D1191	Индексный регистр F5			○	0	-	-	R/W	нет	0
D1192	Индексный регистр E6			○	0	-	-	R/W	нет	0
D1193	Индексный регистр F6			○	0	-	-	R/W	нет	0
D1194	Индексный регистр E7			○	0	-	-	R/W	нет	0
D1195	Индексный регистр F7			○	0	-	-	R/W	нет	0
D1196	SX: Регистр, в который записывается информация для отображения на дисплеи.		○		0	-	-	R/W	нет	0
D1200*	Начальный адрес определения энергонезависимого диапазона адресов реле M0~M999.		○	○	-	-	-	R/W	да	#
D1201*	Конечный адрес определения энергонезависимого диапазона адресов реле M0~M999.		○	○	-	-	-	R/W	да	999
D1202*	Начальный адрес определения энергонезависимого диапазона адресов реле M2000~M4095.		○	○	-	-	-	R/W	да	2000
D1203*	Конечный адрес определения энергонезависимого диапазона адресов реле M2000~M4095.		○	○	-	-	-	R/W	да	4095
D1204*	Начальный адрес определения энергонезависимого диапазона адресов таймеров T0~T199.			○	-	-	-	R/W	да	-1
D1205*	Конечный адрес определения энергонезависимого диапазона адресов таймеров T0~T199.			○	-	-	-	R/W	да	-1
D1206*	Начальный адрес определения энергонезависимо диапазона адресов таймеров T200~T239.			○	-	-	-	R/W	да	-1
D1207*	Начальный адрес определения энергонезависимо диапазона адресов таймеров T200~T239.			○	-	-	-	R/W	да	-1
D1208*	Начальный адрес определения энергонезависимого диапазона адресов счетчиков C0~C199.		○	○	-	-	-	R/W	да	#
D1209*	Конечный адрес определения энергонезависимого диапазона адресов счетчиков C0~C199.		○	○	-	-	-	R/W	да	199
D1210*	Начальный адрес определения энергонезависимого диапазона адресов счетчиков C200~C234.		○	○	-	-	-	R/W	да	#
D1211*	Конечный адрес определения энергонезависимого диапазона адресов счетчиков C200~C234.		○	○	-	-	-	R/W	да	234
D1212*	Начальный адрес определения энергонезависимого диапазона адресов счетчиков C235~C255.		○	○	-	-	-	R/W	да	235
D1213*	Конечный адрес определения энергонезависимого диапазона адресов счетчиков C235~C255.		○	○	-	-	-	R/W	да	255

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1214*	Начальный адрес определения энергонезависимого диапазона адресов шаговых реле S0~S1023.		○	○	-	-	-	R/W	да	#
D1215*	Конечный адрес определения энергонезависимого диапазона адресов шаговых реле S0~S1023.		○	○	-	-	-	R/W	да	#
D1216*	Начальный адрес определения энергонезависимого диапазона регистров D0~D999.		○	○	-	-	-	R/W	да	200
D1217*	Конечный адрес определения энергонезависимого диапазона регистров D0~D999.		○	○	-	-	-	R/W	да	999
D1218*	Начальный адрес определения энергонезависимого диапазона регистров D2000~D9999.		○	○	-	-	-	R/W	да	2000
D1219*	Конечный адрес определения энергонезависимого диапазона регистров D2000~D9999.		○	○	-	-	-	R/W	да	#
D1220	Фаза первой импульсной группы CH0 (Y0, Y1).			○	0	-	-	R/W	нет	0
D1221	Фаза второй импульсной группы CH1 (Y2, Y3).			○	0	-	-	R/W	нет	0
D1222	Установка разницы по времени между сигналом направления и выходными импульсами первой группы CH0 (Y0, Y1) для инструкций DRVI, DDRVI, DRVA, DDRVA, PLSV, DPLSV.			○	0	-	-	R/W	нет	0
D1223	Установка разницы по времени между сигналом направления и выходными импульсами второй группы CH1 (Y2, Y3) для инструкций DRVI, DDRVI, DRVA, DDRVA, PLSV, DPLSV.			○	0	-	-	R/W	нет	0
D1225	Выбор режима счета для HHSC0.			○	2	-	-	R/W	нет	2
D1226	Выбор режима счета для HHSC1.			○	2	-	-	R/W	нет	2
D1227	Выбор режима счета для HHSC2.			○	2	-	-	R/W	нет	2
D1228	Выбор режима счета для HHSC3.			○	2	-	-	R/W	нет	2
D1256 ↓ D1295	Регистры для хранения данных, отправленных инструкцией MODRW.	○	○	○	0	-	-	R	нет	0
D1296 ↓ D1311	Данные, полученные инструкцией MODRW, будут автоматически конвертироваться из кодов ASCII в HEX и сохраняться в D1296 – D1311.	○	○	○	0	-	-	R	нет	0
D1313*	Текущее значение секунд в часах реального времени (RTC), диапазон 00~59.		○	○	#	-	-	R/W	нет	0
D1314*	Текущее значение минут в часах реального времени (RTC), диапазон 00~59.		○	○	#	-	-	R/W	нет	0
D1315*	Текущее значение часов в часах реального времени (RTC), диапазон 00~23.		○	○	#	-	-	R/W	нет	0
D1316*	Текущее значение дней в часах реального времени (RTC), диапазон 01~31.		○	○	#	-	-	R/W	нет	1
D1317*	Текущее значение месяцев в часах реального времени (RTC), диапазон 01~12.		○	○	#	-	-	R/W	нет	1
D1318*	Текущее значение недель в часах реального времени (RTC), диапазон 1~7.		○	○	#	-	-	R/W	нет	6

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1319*	Текущее значение лет в часах реального времени (RTC), диапазон 00~99.		○	○	#	-	-	R/W	нет	0
D1320*	EH/EH2: Идентификационный номер первого специального модуля расширения.			○	0	-	-	R	нет	0
D1321*	EH/EH2: Идентификационный номер второго специального модуля расширения.			○	0	-	-	R	нет	0
D1322*	EH/EH2: Идентификационный номер третьего специального модуля расширения.			○	0	-	-	R	нет	0
D1323*	EH/EH2: Идентификационный номер четвертого специального модуля расширения.			○	0	-	-	R	нет	0
D1324*	EH/EH2: Идентификационный номер пятого специального модуля расширения.			○	0	-	-	R	нет	0
D1325*	EH/EH2: Идентификационный номер шестого специального модуля расширения.			○	0	-	-	R	нет	0
D1326*	EH/EH2: Идентификационный номер седьмого специального модуля расширения.			○	0	-	-	R	нет	0
D1327*	EH/EH2: Идентификационный номер восьмого специального модуля расширения.			○	0	-	-	R	нет	0
D1328	Младшее слово значения смещения первой импульсной группы CH0 (Y0,Y1).			○	0	-	-	R/W	нет	0
D1329	Старшее слово значения смещения первой импульсной группы CH0 (Y0,Y1).			○	0	-	-	R/W	нет	0
D1330	Младшее слово значения смещения второй импульсной группы CH1 (Y2,Y3).			○	0	-	-	R/W	нет	0
D1331	Старшее слово значения смещения второй импульсной группы CH1 (Y2,Y3).			○	0	-	-	R/W	нет	0
D1332	Младшее слово значения оставшегося количества импульсов первой импульсной группы CH0 (Y0,Y1).			○	0	-	-	R	нет	0
D1333	Старшее слово значения оставшегося количества импульсов первой импульсной группы CH0 (Y0,Y1).			○	0	-	-	R	нет	0
D1334	Младшее слово значения оставшегося количества импульсов второй импульсной группы CH1 (Y2,Y3).			○	0	-	-	R	нет	0
D1335	Старшее слово значения оставшегося количества импульсов второй импульсной группы CH1 (Y2,Y3).			○	0	-	-	R	нет	0
D1336	Младшее слово текущего количества выданных импульсов первой импульсной группой CH0 (Y0,Y1).			○	-	0	0	R	нет	0
D1337	Старшее слово текущего количества выданных импульсов первой импульсной группой CH0 (Y0,Y1).			○	-	0	0	R	нет	0
D1338	Младшее слово текущего количества выданных импульсов второй импульсной группой CH1 (Y2,Y3).			○	-	0	0	R	нет	0
D1339	Старшее слово текущего количества выданных импульсов второй импульсной группой CH1 (Y2,Y3).			○	-	0	0	R	нет	0
D1340	EH/EH2/SV: частота первого и последнего шага первой импульсной группы CH0 (Y0,Y1). SC: частота старта/останова выхода Y10.		○	○	- 200	-	-	R/W	да нет	200

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1341	Младшее слово значения максимальной частоты высокоскоростных выходов, фиксировано 200 кГц.			○	H04D 0	-	-	R	да	H04D0
D1342	Старшее слово значения максимальной частоты высокоскоростных выходов, фиксировано 200 кГц.			○	3	-	-	R	да	3
D1343	Время разгона/торможения: EH/EH2/SV: для первой импульсной группы CH0 (Y0, Y1) SC: для выхода Y10.		○	○	- 200	-	-	R/W	да нет	100 200
D1344	Младшее слово количества компенсационных импульсов первой группы CH0 (Y0,Y1).			○	-	-	-	R/W	да	0
D1345	Старшее слово количества компенсационных импульсов первой группы CH0 (Y0,Y1).			○	-	-	-	R/W	да	0
D1346	Младшее слово количества компенсационных импульсов второй группы CH1 (Y2,Y3).			○	-	-	-	R/W	да	0
D1347	Старшее слово количества компенсационных импульсов второй группы CH1 (Y2,Y3).			○	-	-	-	R/W	да	0
D1348	SC: Младшее слово текущего числа выданных импульсов выходом Y10.		○		0	-	-	R	нет	0
D1349	SC: Старшее слово текущего числа выданных импульсов выходом Y10.		○		0	-	-	R	нет	0
D1350	SC: Младшее слово текущего числа выданных импульсов выходом Y11.		○		0	-	-	R	нет	0
D1351	SC: Старшее слово текущего числа выданных импульсов выходом Y11.		○		0	-	-	R	нет	0
D1352	EH/EH2/SV: частота первого и последнего шага первой импульсной группы CH0 (Y0,Y1). SC: частота старта/останова выхода Y10.		○	○	- 200	-	-	R/W	да нет	200
D1353	Время разгона/торможения: EH/EH2/SV: для второй импульсной группы CH1 (Y2, Y3) SC: для выхода Y11.		○	○	- 200	-	-	R/W	да нет	100 200
D1355*	Адрес начального регистра в Ведомом-1, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H106 4	-	-	R/W	нет	H1064
D1356*	Адрес начального регистра в Ведомом-2, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H106 4	-	-	R/W	нет	H1064
D1357*	Адрес начального регистра в Ведомом-3, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H106 4	-	-	R/W	нет	H1064
D1358*	Адрес начального регистра в Ведомом-4, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H106 4	-	-	R/W	нет	H1064
D1359*	Адрес начального регистра в Ведомом-5, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H106 4	-	-	R/W	нет	H1064
D1360*	Адрес начального регистра в Ведомом-6, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H106 4	-	-	R/W	нет	H1064
D1361*	Адрес начального регистра в Ведомом-7, откуда		○	○	H106	-	-	R/W	нет	H1064

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
	будут считываться данные Мастером в режиме EASY PLC LINK.				4					
D1362*	Адрес начального регистра в Ведомом-8, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1363*	Адрес начального регистра в Ведомом-9, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1364*	Адрес начального регистра в Ведомом-10, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1365*	Адрес начального регистра в Ведомом-11, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1366*	Адрес начального регистра в Ведомом-12, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1367*	Адрес начального регистра в Ведомом-13, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1368*	Адрес начального регистра в Ведомом-14, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1369*	Адрес начального регистра в Ведомом-15, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1370*	Адрес начального регистра в Ведомом-16, откуда будут считываться данные Мастером в режиме EASY PLC LINK.		○	○	H1064	-	-	R/W	нет	H1064
D1371	Единицы времени для инструкции PWM выход Y0, когда M1070=1.			○	1	-	-	R/W	нет	1
D1372	Единицы времени для инструкции PWM выход Y2, когда M1071=1.			○	1	-	-	R/W	нет	1
D1373	EH2/SV: Единицы времени для инструкции PWM выход Y4, когда M1530=1.			○	1	-	-	R/W	нет	1
D1374	EH2/SV: Единицы времени для инструкции PWM выход Y6, когда M1531=1.			○	1	-	-	R/W	нет	1
D1375	EH2/SV: Младшее слово текущего значения выданного количества импульсов третьей группой CH2 (Y4, Y5).			○	-	-	-	R/W	да	0
D1376	EH2/SV: Старшее слово текущего значения выданного количества импульсов третьей группой CH2 (Y4, Y5).			○	-	-	-	R/W	да	0
D1377	EH2/SV: Младшее слово текущего значения выданного количества импульсов четвертой группой CH3 (Y6, Y7).			○	-	-	-	R/W	да	0
D1378	EH2/SV: Старшее слово текущего значения выданного количества импульсов четвертой группой CH3 (Y6, Y7).			○	-	-	-	R/W	да	0
D1379	EH2/SV: Частота старта/останова третьей импульсной группы CH2 (Y4, Y5).			○	-	-	-	R/W	да	200
D1380	EH2/SV: Частота старта/останова четвертой импульсной группы CH3 (Y6, Y7).			○	-	-	-	R/W	да	200
D1381	EH2/SV: Время ускорения/замедления третьей			○	-	-	-	R/W	да	100

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
	импульсной группы CH2 (Y4, Y5).									
D1382	EH2/SV: Время ускорения/замедления четвертой импульсной группы CH3 (Y6, Y7).			○	-	-	-	R/W	да	100
D1386*	SV: Идентификационный номер первого левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1387*	SV: Идентификационный номер второго левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1388*	SV: Идентификационный номер третьего левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1389*	SV: Идентификационный номер четвертого левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1390*	SV: Идентификационный номер пятого левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1391*	SV: Идентификационный номер шестого левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1392*	SV: Идентификационный номер седьмого левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1393*	SV: Идентификационный номер восьмого левостороннего модуля расширения.			○	0	-	-	R	нет	0
D1399	Сетевой адрес (в сети Modbus) первого Ведомого устройства (в десятичном формате), которому в рамках режима EASY PLC LINK присваивается идентификационный номер "Ведомый-1".		○	○	1	-	-	R/W	нет	1
D1415*	Адрес начального регистра в Ведомом-1, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1416*	Адрес начального регистра в Ведомом-2, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1417*	Адрес начального регистра в Ведомом-3, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1418*	Адрес начального регистра в Ведомом-4, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1419*	Адрес начального регистра в Ведомом-5, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1420*	Адрес начального регистра в Ведомом-6, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1421*	Адрес начального регистра в Ведомом-7, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1422*	Адрес начального регистра в Ведомом-8, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1423*	Адрес начального регистра в Ведомом-9, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1424*	Адрес начального регистра в Ведомом-10, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1425*	Адрес начального регистра в Ведомом-11, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1426*	Адрес начального регистра в Ведомом-12, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1427*	Адрес начального регистра в Ведомом-13, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1428*	Адрес начального регистра в Ведомом-14, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1429*	Адрес начального регистра в Ведомом-15, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1430*	Адрес начального регистра в Ведомом-16, куда будут записываться данные Мастером в режиме EASY PLC LINK.		○	○	H10C 8	-	-	R/W	нет	H10C8
D1431*	Количество циклов опроса в режиме PLC LINK.		○	○	0	-	-	R/W	нет	0
D1432*	Текущее значение отработанных циклов режима EASY PLC LINK.		○	○	0	-	-	R/W	нет	0
D1433*	Заданное количество Ведомых, опрашиваемых Мастером в режиме EASY PLC LINK.		○	○	0	-	-	R/W	нет	0
D1434*	Количество читаемых регистров Мастером в Ведомом 1 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1435*	Количество читаемых регистров Мастером в Ведомом 2 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1436*	Количество читаемых регистров Мастером в Ведомом 3 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1437*	Количество читаемых регистров Мастером в Ведомом 4 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1438*	Количество читаемых регистров Мастером в Ведомом 5 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1439*	Количество читаемых регистров Мастером в Ведомом 6 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1440*	Количество читаемых регистров Мастером в Ведомом 7 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1441*	Количество читаемых регистров Мастером в Ведомом 8 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1442*	Количество читаемых регистров Мастером в Ведомом 9 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1443*	Количество читаемых регистров Мастером в Ведомом 10 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1444*	Количество читаемых регистров Мастером в Ведомом 11 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1445*	Количество читаемых регистров Мастером в Ведомом 12 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1446*	Количество читаемых регистров Мастером в Ведомом 13 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1447*	Количество читаемых регистров Мастером в Ведомом 14 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1448*	Количество читаемых регистров Мастером в Ведомом 15 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1449*	Количество читаемых регистров Мастером в Ведомом 16 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1450*	Количество записываемых регистров Мастером в Ведомый 1 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1451*	Количество записываемых регистров Мастером в Ведомый 2 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1452*	Количество записываемых регистров Мастером в Ведомый 3 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1453*	Количество записываемых регистров Мастером в Ведомый 4 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1454*	Количество записываемых регистров Мастером в Ведомый 5 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1455*	Количество записываемых регистров Мастером в Ведомый 6 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1456*	Количество записываемых регистров Мастером в Ведомый 7 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1457*	Количество записываемых регистров Мастером в Ведомый 8 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1458*	Количество записываемых регистров Мастером в Ведомый 9 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1459*	Количество записываемых регистров Мастером в Ведомый 10 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1460*	Количество записываемых регистров Мастером в Ведомый 11 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1461*	Количество записываемых регистров Мастером в Ведомый 12 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1462*	Количество записываемых регистров Мастером в Ведомый 13 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1463*	Количество записываемых регистров Мастером в Ведомый 14 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1464*	Количество записываемых регистров Мастером в Ведомый 15 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1465*	Количество записываемых регистров Мастером в Ведомый 16 в режиме EASY PLC LINK.		○	○	16	-	-	R/W	нет	16
D1466	Младшее слово значения необходимого количества импульсов на одни оборот двигателя для канала CH0.			○	-	-	-	R	да	2000
D1467	Старшее слово значения необходимого количества импульсов на одни оборот двигателя для канала CH0.			○	-	-	-	R	да	0
D1468	Младшее слово значения необходимого количества импульсов на одни оборот двигателя для канала CH1.			○	-	-	-	R	да	2000
D1469	Старшее слово значения необходимого количества импульсов на одни оборот двигателя для канала CH1.			○	-	-	-	R	да	0
D1470	Младшее слово значения линейного перемещения на один оборот двигателя для канала CH0.			○	-	-	-	R	да	1000
D1471	Старшее слово значения линейного перемещения на один оборот двигателя для канала CH0.			○	-	-	-	R	да	0
D1472	Младшее слово значения линейного перемещения на один оборот двигателя для			○	-	-	-	R	да	1000

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
	канала CH1.									
D1473	Старшее слово значения линейного перемещения на один оборот двигателя для канала CH1.			○	-	-	-	R	да	0
D1474	Младшее слово значения единиц измерения механического движения канала CH0.			○	-	-	-	R	да	0
D1475	Старшее слово значения единиц измерения механического движения канала CH0.			○	-	-	-	R	да	0
D1476	Младшее слово значения единиц измерения механического движения канала CH1.			○	-	-	-	R	да	0
D1477	Старшее слово значения единиц измерения механического движения канала CH1.			○	-	-	-	R	да	0
D1480* ↓ D1495*	Когда M1353=0 данные, прочитанные Мастером в Ведомом 1 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, адреса начальных регистров Мастера, куда будут записываться данные, прочитанные с Ведомых 1 ~ 16 в текущем цикле.		○	○	0	-	-	R	нет	0
D1496* ↓ D1511*	Когда M1353=0 данные, которые будут записаны Мастером в Ведомый 1 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, адреса начальных регистров Мастера, откуда данные будут записаны в Ведомые 1 ~ 16 в текущем цикле.		○	○	0	-	-	R/W	нет	0
D1512* ↓ D1527*	Когда M1353=0 данные, прочитанные Мастером в Ведомом 2 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, адреса начальных регистров Ведомых 17 ~ 32, откуда Мастер будет читать данные.		○	○	0	-	-	R	нет	0
D1528* ↓ D1543*	Когда M1353=0 данные, которые будут записаны Мастером в Ведомый 2 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, адреса начальных регистров Ведомых 17 ~ 32, куда Мастер будет записывать данные.		○	○	0	-	-	R/W	нет	0
D1544* ↓ D1559*	Когда M1353=0 данные, прочитанные Мастером в Ведомом 3 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, количество регистров в Ведомых 17 ~ 32, которое будет считывать Мастер.		○	○	0	-	-	R	нет	0
D1560* ↓ D1575*	Когда M1353=0 данные, которые будут записаны Мастером в Ведомый 3 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, количество регистров в Ведомых 17 ~ 32, в которые Мастер будет записывать данные.		○	○	0	-	-	R/W	нет	0
D1576* ↓ D1591*	Когда M1353=0 данные, прочитанные Мастером в Ведомом 4 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, адреса начальных регистров Мастера для хранения данных,		○	○	0	-	-	R	нет	0

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
	принятых от Ведомых 17 ~ 32.									
D1592* ↓ D1607*	Когда M1353=0 данные, которые будут записаны Мастером в Ведомый 4 в текущем цикле опроса EASY PLC LINK . EH/EH2/SV: когда M1353=1, адреса начальных регистров Мастера, откуда данные будут записаны в Ведомые 17 ~ 32.		○	○	0	-	-	R/W	нет	0
D1608* ↓ D1623*	Данные, прочитанные Мастером в Ведомом 5 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1624* ↓ D1639*	Данные, которые будут записаны Мастером в Ведомый 5 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1640* ↓ D1655*	Данные, прочитанные Мастером в Ведомом 6 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1656* ↓ D1671*	Данные, которые будут записаны Мастером в Ведомый 6 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1672* ↓ D1687*	Данные, прочитанные Мастером в Ведомом 7 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1688* ↓ D1703*	Данные, которые будут записаны Мастером в Ведомый 7 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1704* ↓ D1719*	Данные, прочитанные Мастером в Ведомом 8 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1720* ↓ D1735*	Данные, которые будут записаны Мастером в Ведомый 8 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1736* ↓ D1751*	Данные, прочитанные Мастером в Ведомом 9 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1752* ↓ D1767*	Данные, которые будут записаны Мастером в Ведомый 9 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1768* ↓ D1783*	Данные, прочитанные Мастером в Ведомом 10 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1784* ↓ D1799*	Данные, которые будут записаны Мастером в Ведомый 10 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1800* ↓ D1815*	Данные, прочитанные Мастером в Ведомом 11 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1816* ↓ D1831*	Данные, которые будут записаны Мастером в Ведомый 11 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1832* ↓ D1847*	Данные, прочитанные Мастером в Ведомом 12 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0

Номер регистра	Функция	ES EX SS	SA SX SC	EH SV	Off —> On	STOP —> RUN	RUN —> STOP	Чтение/ Запись	Энерго-незави-симость	По умолча нию
D1848* ↓ D1863*	Данные, которые будут записаны Мастером в Ведомый 12 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1864* ↓ D1879*	Данные, прочитанные Мастером в Ведомом 13 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1880* ↓ D1895*	Данные, которые будут записаны Мастером в Ведомый 13 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1896* ↓ D1911*	Данные, прочитанные Мастером в Ведомом 14 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1912* ↓ D1927*	Данные, которые будут записаны Мастером в Ведомый 14 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1928* ↓ D1943*	Данные, прочитанные Мастером в Ведомом 15 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1944* ↓ D1959*	Данные, которые будут записаны Мастером в Ведомый 15 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0
D1960* ↓ D1975*	Данные, прочитанные Мастером в Ведомом 16 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R	нет	0
D1976* ↓ D1991*	Данные, которые будут записаны Мастером в Ведомый 16 в текущем цикле опроса EASY PLC LINK .		○	○	0	-	-	R/W	нет	0

2.11 Описание специальных реле и регистров

2.11.1 Флаги состояния ПЛК (M1000 – M1003)

M1000: Отображает состояние ПЛК – контакт всегда замкнут в режиме RUN (работа) и разомкнут в состояние STOP (стоп).

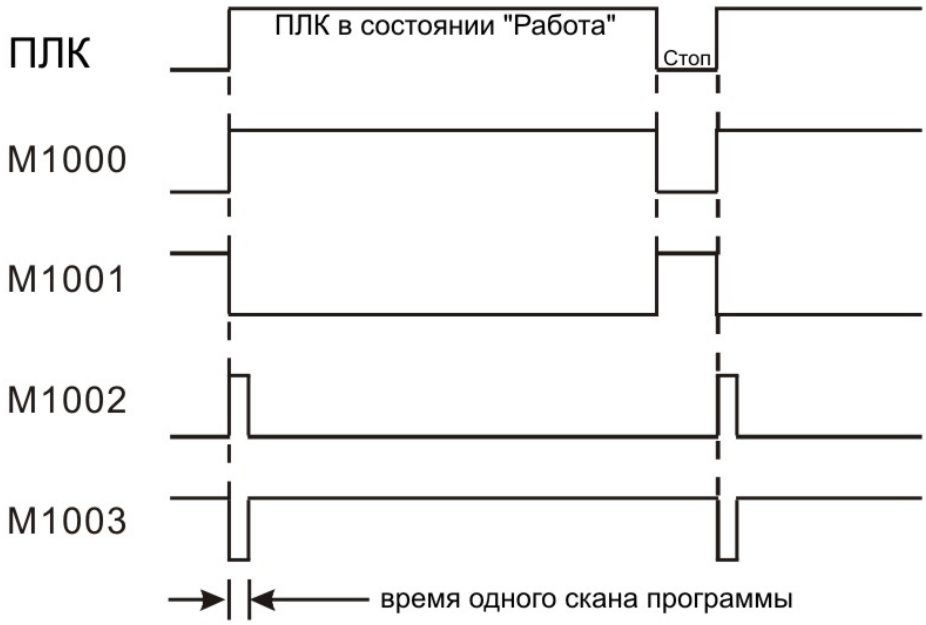
M1000, а значит и Y0 будет замкнут всегда, когда программа выполняется, т.е. ПЛК находится в состоянии RUN.

M1001: Отображает состояние ПЛК – контакт всегда разомкнут в режиме RUN (Работа) и замкнут в состояние STOP (Стоп).

M1002 замкнут в течение первого цикла сканирования, когда ПЛК запускается, и разомкнут в течении остального времени. Реле M1002 удобно использовать в программе в качестве импульса начальной инициализации.

M1003 разомкнут в течение первого цикла сканирования, когда ПЛК запускается, и замкнут в течении остального времени.

Ниже приведена временная диаграмма работы специальных реле M1000 – M1003:



2.11.2 Регистр текущей уставки сторожевого таймера (D1000)

Используется для контроля времени цикла. Если время одного цикла превысит заданную уставку сторожевого таймера, загорится светодиод "ERROR" и все выходы будут сброшены. Настраивается в милисекундах. Значение по умолчанию 200 мс. Для изменения времени

можно использовать команду MOV (в данном примере уставка будет 300 мс):

Максимальное значение сторожевого таймера: 32767 мс. Внимательно устанавливайте значение таймера, т.к. большое значение может привести к большой задержке обнаружения аварийной ситуации. Время сканирования может быть очень длинным за счет использования в программе сложных вычислений или большого количества специальных модулей расширения. Для корректной установки времени сторожевого таймера посмотрите время сканирования в регистрах D1010 - D1012.

Вы также можете использовать в программе для контроля времени цикла инструкцию WDT (API 07), которая может разделять цикл программы на отрезки со сбросом времени сторожевого таймера в конце каждого отрезка программы.

2.11.3 Максимальное количество шагов программы (D1002)

В различных типах ПЛК это значение будет разным:

1. ES, EX, SS: 3792 шагов
2. SX, SA, SC: 7920 шагов
3. EH, EH2, SV: 15872 шагов

2.11.4 Проверка синтаксических ошибок в программе (M1004, D1004, D1137)

При выявлении синтаксической ошибки в программе, светодиод ERROR начнет мигать и реле M1004 включится. Это может произойти при неправильном использовании операнда или грамматических ошибках в теле программы.
Проверка синтаксиса осуществляется при переводе контроллера из состояния СТОП в состояние РАБОТА, при загрузке программы, а также при использовании on-line программирования в контроллерах типов SA/SX/SC/EH/EH2/SV и WPLSoft.
При обнаружении ошибки, ее код записывается в регистр D1004, а шаг программы, где обнаружена ошибка, в регистр D1137. В случае общей ошибки программы (всего цикла), регистр D1137 будет недоступен.
Коды ошибок можно посмотреть в конце данной Главы в параграфе 2.12.

2.11.5 Карта памяти резервирования данных (M1005 – M1007)

Данные реле доступны только в контроллерах типов EH/EH2, когда в них вставлена соответствующая карта расширения (карта памяти) и переключатель ON/OFF на ней включен.

Контроллер осуществляет инициализацию карты памяти, проводит операцию сравнения данных на ней и в своей памяти и, при отсутствии каких-либо ошибок, осуществляет копирование данных с карты.
Если в ходе операции сравнения были выявлены какие-либо ошибки, то копирование данных не осуществляется и включаются соответствующие реле:

M1005=1, если пароль в карте памяти не совпадает с паролем в ПЛК

M1006=1, если карта памяти не прошла инициализацию

M1007=1, если данные отсутствуют в области программы карты памяти

2.11.6 Флаги сторожевого таймера (M1008, D1008)

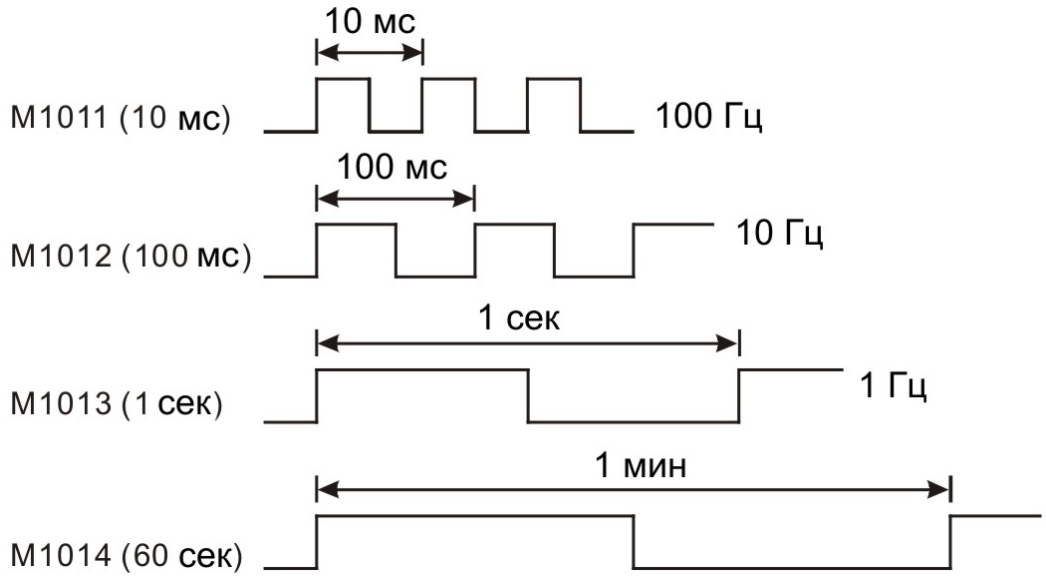
Если время прохождения цикла программы превысит заданное время, загорится светодиод ERROR и реле M1008 замкнется. В регистр D1008 будет записан номер шага программы, на котором была вызвана задержка выполнения программы.

2.11.7 Мониторинг времени прохождения цикла программы (D1010 – D1012)

В данных регистрах отображается текущее (D1010), минимальное (D1011) и максимальное время (D1012) выполнения программы.

2.11.8 Встроенные генераторы тактовых импульсов (M1011 – M1014)

Контроллер может автоматически выдавать в программе импульсы с четырьмя различными периодами:



Встроенные генераторы импульсов начинают работать автоматически при подаче питания на контроллер. Соответствующие контакты (M1011~M1014) будут замыкаться также

автоматически при использовании их в программе на протяжении всего времени работы ПЛК, в т.ч. в состоянии СТОП.
При переводе ПЛК в состояние RUN (Работа) тактовые импульсы с началом выполнения программы не синхронизируются.

2.11.9 Высокоскоростной таймер (M1015, D1015)

Используется для подсчета времени с дискретностью 0.1 мс. У контроллеров EH/EH2/SV работает только в состоянии RUN, у SA/SX/SC и в состоянии СТОП тоже.
Когда M1015 = 1, начнется работа высокоскоростного таймера сразу после выполнения инструкции END в текущем скане. Текущее время записывается в специальном регистре D1015. Диапазон D1015: 0 … 32767, единица - 100 мкс. Счет идет по кругу, т.е. при достижении текущего значения 32767 счет снова начнется с ноля. Когда M1015 = 0, счет времени немедленно прекратится.
Для работы с высокоскоростным таймером в EH/EH2/SV может использоваться инструкция HST (API 196).

2.11.10 Часы реального времени (M1016, M1017, M1076, D1313~D1319)

Часы задействуют несколько специальных регистров и реле:

M1016	Отображение года	Когда M1016=OFF, будут отображаться две младших цифры. Когда M1016=ON, будут отображаться две младших цифры + 2000
M1017	Корректировка секунд (± 30 сек)	При переключении с OFF на ON секунды будут обнулены и минуты не изменятся, если секунды находились в диапазоне от 0 до 29; и будет добавлена 1 минута, если секунды находились в диапазоне 30 … 59 сек.
M1076	Ошибка календаря (часов реального времени)	Контакт включится если будет превышен установленный диапазон в настройках или батарея будет иметь низкий заряд.
M1082	Смена настроек	Контакт включится при изменении текущих настроек часов реального времени.
D1313	Часы реального времени: отображение и коррекция секунд	(00…59)
D1314	Отображение и коррекция минут	(00…59)
D1315	Отображение и коррекция часов	(00…23)
D1316	Отображение и коррекция дня месяца	(01…31)

D1317	Отображение и коррекция месяца	(01…12)
D1318	Отображение и коррекция дня недели	(1…7)
D1319	Отображение и коррекция года	(00…99)

При сбое часов реального времени произойдет сброс на 1 января 2000 г. 00:00 суббота. Корректировка часов реального времени в может быть выполнена с помощью инструкции TWR (API 167), а также WPLSoft или панельки DU-01.

2.11.11 Число **π** (ПИ) (D1018 ~ D1019)

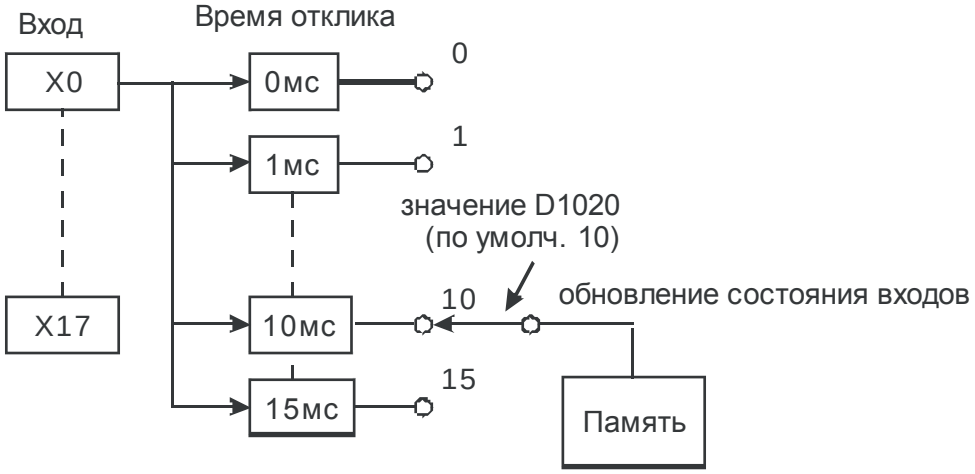
Число π записано в двух последовательных регистрах в формате числа с плавающей точкой (32 бита).
Значение в шестнадцатеричном формате с плавающей точкой = H40490FBD.

2.11.12 Настройка времени реакции входов (D1020 ~ D1021)

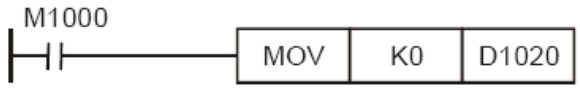
При помощи записи значения в регистр D1020 настраиваются входы X0~X7 всех типов контроллеров. Диапазон 0~20 у SS/ES/EX/SA/SX/SC, и 0~60 EH/EH2/SV, ед. мс.
При помощи записи значения в ячейку D1021 можно изменять реакцию входов X10~X17 контроллеров типов ES, диапазон 0~20, ед. мс, и EH/EH2/SV, диапазон 0~60, ед. мс.

В контроллерах SC время реакции высокоскоростных импульсных входов X10~X11 настраивается путем записи значения в ячейку D1021. Диапазон 0~1000, ед. время цикла (скана) программы. Например, если время скана 10 мс, а ячейка D1021=2, то время реакции будет 20 мс.

При включении ПЛК содержимое регистров D1020, D1021 будет автоматически установлено на значение 10 мс.



Если D1020 = 0, время отклика будет зависеть только от последовательного RC-фильтра и будет максимально быстрым (50 мкс).



Нет необходимости корректировать время отклика дискретных входов, когда в программе используются высокоскоростные счетчики, внешние прерывания или импульсные ловушки. Для корректировки времени отклика дискретных входов также можно использовать команду REFF (API 51).

2.11.13 Флаги окончания выполнения команд (M1029, M1030, M1036, M1037, M1102, M1103)

1. При завершении выполнения инструкций MTR (API 52), HKY (API 71), DSW (API 72), SEGL (API 74), PR (API 77) специальное реле M1029 будет включено в течение одного скана.
2. При завершении выполнения инструкций API 57 PLSY, API 59 PLSR:
   - У контроллеров SA/SX/SC/ES/EX/SS реле M1029 включится, когда выход Y0 закончит выдачу установленного числа импульсов. Реле M1030 включится, когда выход Y1 закончит выдачу установленного числа импульсов. Когда инструкции PLSY и PLSR выключатся, реле M1029 и M1030 сбросятся. Если необходимо продолжить работу с инструкциями без их выключения, реле M1029 и M1030 необходимо сбросить командой RST.
   - У контроллеров EH/EH2/SV реле M1029 включится, когда выходы Y0 и Y1 закончат выдачу импульсов. Реле M1030 включится, когда выходы Y2 и Y3 закончат выдачу импульсов. У контроллеров EH2/SV реле M1036 включится, когда выходы Y4 и Y5 закончат выдачу импульсов. Реле M1037 включится, когда выходы Y6 и Y7 закончат выдачу импульсов. Когда инструкции PLSY и PLSR выключатся, реле M1029, M1030, M1036 и M1037 сбросятся. Если инструкции будут задействованы снова, то указанные реле снова включатся по достижении соответствующими выходами заданного числа импульсов.
3. При выполнении инструкции API 63 INCD реле M1029 будет включено в течение одного скана по достижении последней уставки многодиапазонного счетчика.
4. При завершении выполнения инструкций API 67 RAMP и API 69 SORT реле M1029 будет включено в течение одного скана. Когда инструкции RAMP и SORT выключатся, реле M1029 и M1030 сбросятся. Если необходимо продолжить работу с инструкциями без их выключения, реле M1029 и M1030 необходимо сбросить командой RST.
5. При завершении выполнения инструкций API 155 DABSR, API 156 ZRN, API 158 DRVI, API 159 DRVA для контроллеров EH/EH2/SV:
   - Реле M1029 включится при достижении первой импульсной группой (Y0, Y1) заданного числа импульсов. Реле M1030 включится при достижении второй импульсной группой (Y2, Y3) заданного числа импульсов.
   - Реле M1036 включится при достижении третьей импульсной группой (Y4, Y5) заданного числа импульсов. Реле M1037 включится при достижении четвертой импульсной группой (Y6, Y7) заданного числа импульсов.
   - Если инструкции будут задействованы снова, то указанные реле снова включатся по достижении соответствующими выходами заданного числа импульсов.
6. При завершении выполнения инструкций API 155 DABSR, API 156 DZRN, API 158

DDRVI, API 159 DDRVA для контроллеров SC:

- Реле M1102 включится при достижении выходом Y10 заданного числа импульсов.
- Реле M1103 включится при достижении выходом Y11 заданного числа импульсов.
- По окончании выполнения инструкции PLSY реле M1102 и M1103 сбросятся. После окончания выполния инструкций DZRN, DDRVI и DDRVA реле M1102 и M1103 сбросятся только при следующей активации данных инструкций.

2.11.14 Код ошибки при обмене данными (M1025, D1025)

Когда к ПЛК подключено одно из устройств - HMI, HPP или PC, которое посылает ПЛК нештатный запрос, ПЛК установит M1025=ON и запишет код ошибки в регистр D1025.
Расшифровка кодов ошибок коммуникационных запросов приведены ниже:

01 – неправильный код команды
02 – неправильный адрес регистра
03 – запрашиваемые данные выходят за допустимый диапазон
07 – ошибка при проверке контрольной суммы

2.11.15 Очистка регистров памяти (M1031, M1032)

При включении реле M1031 происходит очистка общих (энергозависимых) регистров. При включении реле M1032 происходит очистка энергонезависимых регистров.

Реле	Операнды
M1031	Сброс контактов Y, контактов общего назначения M и S Сброс контактов общего назначения T и выходных катушек T Сброс контактов общего назначения C и выходных катушек C Очистка текущего значения регистров памяти общего назначения D Очистка текущего значения таймеров общего назначения T Очистка текущего значения счетчиков общего назначения C
M1032	Сброс контактов энергонезависимых M и S Сброс контактов и выходных катушек аккумулятивных таймеров T Сброс контактов и выходных катушек энергонезависимых счетчиков C Очистка текущего значения регистров энергонезависимой памяти D Очистка текущего значения аккумулятивных таймеров T Очистка текущего значения энергонезависимых счетчиков C

2.11.16 Фиксация состояния выходов Y при останове ПЛК (M1033)

Если в программе M1033=1, то при переводе контроллера из режима РАБОТА в режим СТОП будет сохранено текущее состояние физических выходов Y (катушек). Т.е., если выход был включен, то он так и останется включен, а если был выключен, то так и будет выключен.
Данная функция может быть полезна, когда необходимо обеспечить непрерывность техпроцесса во время внесения изменений в программу.

2.11.17 Принудительное отключение всех физических выходов Y (M1034)

Если включить реле M1034, то все физические выходы Y принудительно отключатся и будут недоступны пока включено реле M1034.

2.11.18 Внешний переключатель РАБОТА/СТОП (M1035, D1035)

При M1035=1 активируется функция пуска/останова контроллера от внешнего сигнала, который подается на определенный физический вход ПЛК. У контроллеров EH/EH2/SV номер физического входа определяется в регистре D1035, диапазон K0 ~ K15, входы X0 ~ X17. У контроллеров SA фиксировано выделяется вход X7, у контроллеров SX вход X3, у контроллеров SC вход X5.

2.11.19 Определение ширины импульса на входа X0 (M1084, D1023)

Когда M1084=1, то при каждом переходе входа X0 с ВКЛ на ВЫКЛ, значение промежутка времени, в течение которого вход X0 был включен, записывается в регистр D1023. Таким образом, определяется ширина входного импульса. Единица измерения – 0,1 мс. Диапазон от 0,1 мс до 10000 мс.
Данная функция поддерживается в следующих версиях встроенного ПО:
ES/EX/SS_V6.4/SA/SX_V1.6/SC_V1.4 и выше.

2.11.20 Установка протокола связи для COM-портов (M1120, M1136, M1138, M1139, M1143, D1036, D1109, D1120)

Контроллеры ES/EX/SS/SA/SX/SC/SV оснащены портом COM1 (RS232) и портом COM2 (RS485). Поддерживаются скорости до 115200 бит/сек, режимы RTU/ASCII, любые форматы длины данных, четности и стоповых битов. Порты COM1 и COM2 могут использоваться одновременно.

Контроллеры EH/EH2 оснащены портами COM1 (RS232), COM2 (RS232/RS422/RS485) и, при использовании соответствующей карты расширения, COM3 (RS232/RS485). Порты COM1 и COM2 поддерживают скорости до 115200 бит/сек, режимы RTU/ASCII, любые форматы длины данных, четности и стоповых битов. Порты COM1 и COM2 могут использоваться одновременно. Порт COM3 может работать только в режиме ASCII 7, E, 1 до скорости 38400 бит/сек.

Описание COM портов:

COM1 Работает только в режиме Ведомого. Поддерживает скорости до 115200 бит/сек, режимы RTU/ASCII, любые форматы длины данных, четности и стоповых битов.

COM2 Работает как в режиме Мастера так и Ведомого. Поддерживает скорости до 115200 бит/сек, режимы RTU/ASCII, любые форматы длины данных, четности и стоповых

битов.

COM3 Работает только в режиме Ведомого. Поддерживает только режим ASCII 7 (длина данных), E (по четному биту), 1 (один стоповый бит) до скорости 38400 бит/сек. В режиме Ведомых порты COM2 и COM3 не могут использоваться одновременно.

Установка протокола связи:

COM1
1. Параметры протокола записываются в регистр D1036. Биты b8 ~ b15 не используются.
2. Фиксация протокола осуществляется реле M1138.
3. Включением реле M1139 выбирается режим RTU/ASCII (M1139=1, RTU)

COM2
1. Параметры протокола записываются в регистр D1120.
2. Фиксация протокола осуществляется реле M1120.
3. Включением реле M1143 выбирается режим RTU/ASCII (M1143=1, RTU)

COM3
1. Параметры протокола записываются в регистр D1109. Биты b8 ~ b15 не используются.
2. Фиксация протокола осуществляется реле M1136.

Выбор параметров протокола осуществляется записью битов соответствующего номера согласно нижеприведенной таблицы:

Бит	Значение	0	1
b0	Длина данных	b0=0, длина 7 бит	b0=1, длина 8 бит
b1 b2	бит четности	b2, b1=00 : нет b2, b1=01 : нечетный (odd) b2, b1=11 : четный (even)	
b3	стоповый бит	b3=0 - 1 бит	b3=1 - 2 бита
b4 b5 b6 b7	b7~b4=0001 (H1) : 110 бит/сек b7~b4=0010 (H2) : 150 бит/сек b7~b4=0011 (H3) : 300 бит/сек b7~b4=0100 (H4) : 600 бит/сек b7~b4=0101 (H5) : 1200 бит/сек b7~b4=0110 (H6) : 2400 бит/сек b7~b4=0111 (H7) : 4800 бит/сек b7~b4=1000 (H8) : 9600 бит/сек b7~b4=1001 (H9) : 19200 бит/сек b7~b4=1010 (HA) : 38400 бит/сек b7~b4=1011 (HB) : 57600 бит/сек b7~b4=1100 (HC) : 115200 бит/сек		
b8	Стартовый символ	b8=0: нет	b8=1 из D1124
b9	Первый стоповый символ	b9=0: нет	b9=1 из D1125
b10	Второй стоповый символ	b10=0: нет	b10=1 из D1126
b15~b11	не используются		

Значение каждого бита записывается справа налево (младший бит b0 справа, старший b15 слева) согласно вышеприведенной таблицы. Получается последовательность нулей и единиц,

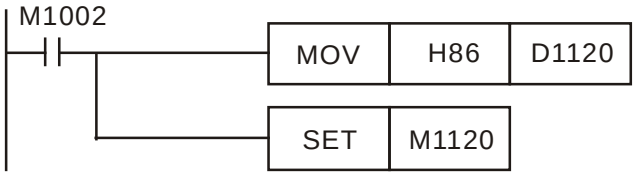
которые необходимо перевести в шестнадцатеричный формат и записать в соответствующий регистр хранения параметров COM порта.
Примеры наиболее распространенных вариантов параметров протокола связи: H86 (9600, 7, E, 1); H87 (9600, 8, E, 1); H96 (19200, 7, E, 1); H97 (19200, 8, E, 1); HA6 (38400, 7, E, 1); HA7 (38400, 8, E, 1).

К контроллерах семейства Delta DVP для всех портов по умолчанию стоит протокол H86 (9600, 7, E, 1).
Шестнадцатеричное значение H86 в двоичном коде будет выглядеть следующим образом:
(b7) 10000110 (b0). Заполняется согласно таблицы выше.

Далее приведены примеры установки различных протоколов связи для коммуникационных портов.

Пример 1.
Установка протокола для порта COM2.

Для установки протокола поместите нижеприведенный блок в самом начале программы:



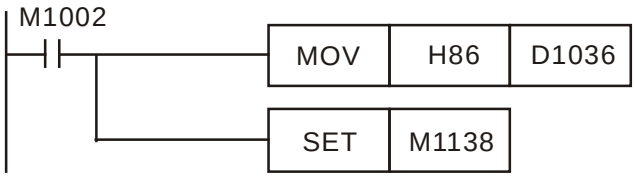
При переводе контроллера из СТОПа в режим РАБОТА, программа определит специальное реле M1120, считает протокол связи из регистра D1120, изменит параметры порта COM2 и зафиксирует их.

Важные замечания:

1. Если порт COM2 используется в качестве Ведомого, то в программе не должно содержаться каких-либо коммуникационных инструкций. В противном случае возникнет конфликт в сети с Мастером и связь не будет установлена.
2. После фиксации протокола он останется без изменений и после перевода контроллера из режима РАБОТА в СТОП.
3. При отключении питания от контроллера и повторном его подаче, протокол вернется к состоянию по умолчанию, т.е. H86. Для повторной установки параметров, указанных в регистре D1120, контроллер необходимо перевести в режим РАБОТА.

Пример 2.
Установка протокола для порта COM1.

Для установки протокола поместите нижеприведенный блок в самом начале программы:



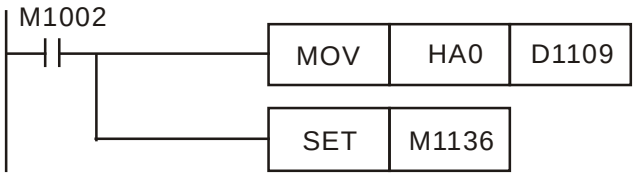
При переводе контроллера из СТОПа в режим РАБОТА, программа определит специальное реле M1138, считает протокол связи из регистра D1036, изменит параметры порта COM1 и зафиксирует их.

Важные замечания:

1. После фиксации протокола он останется без изменений и после перевода контроллера из режима РАБОТА в СТОП.
2. При отключении питания от контроллера и повторном его подаче, протокол вернется к состоянию по умолчанию, т.е. H86. Для повторной установки параметров, указанных в регистре D1120, контроллер необходимо перевести в режим РАБОТА.

Пример 3.
Установка протокола для порта COM3.

Для установки протокола поместите нижеприведенный блок в самом начале программы:



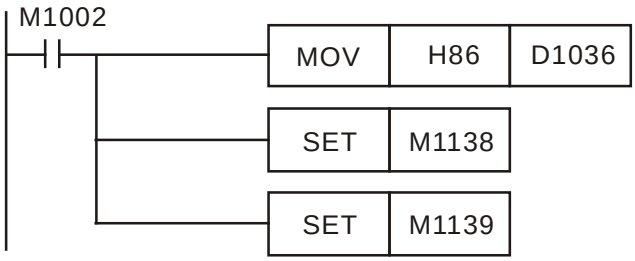
HA0 = 38400, 7, N, 1

Важные замечания:

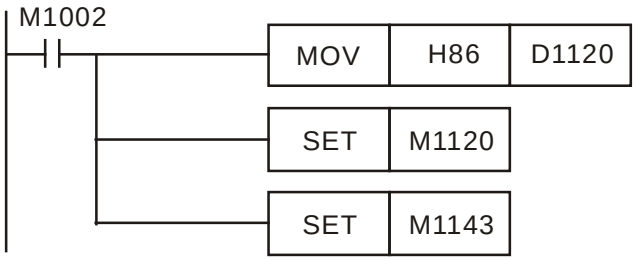
1. После фиксации протокола он останется без изменений и после перевода контроллера из режима РАБОТА в СТОП.
3. При отключении питания от контроллера и повторном его подаче, протокол вернется к состоянию по умолчанию, т.е. H86. Для повторной установки параметров, указанных в регистре D1120, контроллер необходимо перевести в режим РАБОТА.

Пример 4.
Установка режима RTU для портов COM1 и COM2.

Порты COM1 и COM2 поддерживают оба режима Modbus – RTU и ASCII. Для перевода COM1 в режим RTU необходимо включить реле M1139, а для порта COM2 реле M1143. Когда данные реле включены, коммуникационные порты находятся в режиме RTU, когда выключены – в режиме ASCII.
COM1: RTU, 9600, 8, E, 1

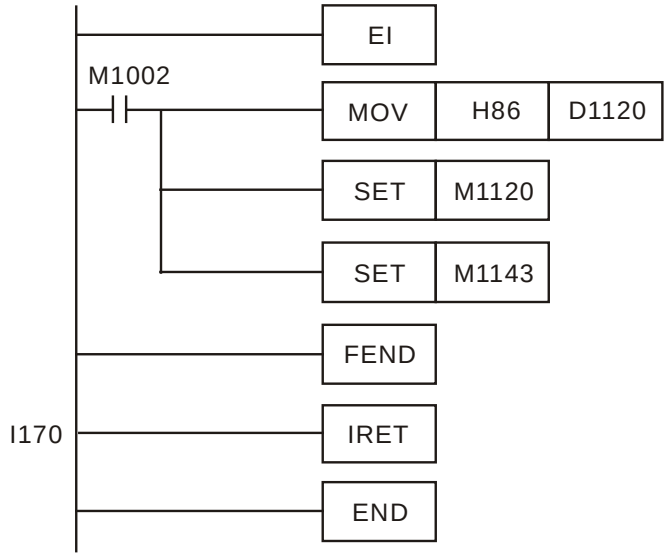


COM2: RTU, 9600, 8, E, 1



Пример 5.
Использование прерывания I170 для порта COM2 контроллеров типов EH/EH2/SV.

Коммуникационное прерывание I170 может быть использовано только контроллерами EH/EH2/SV, работающими в режиме Ведомого, и позволяет осуществить обработку полученных данных сразу по их получению, а не после исполнения команды END в следующем цикле.
Данная функция полезна при большом времени цикла программы, когда данные слишком долго будут стоять в очереди, дожидаясь начала следующего скана после отработки команды END. Использование прерывания I170 позволяет обработать данные сразу же в текущем скане в данном месте программы.



Важные замечания:

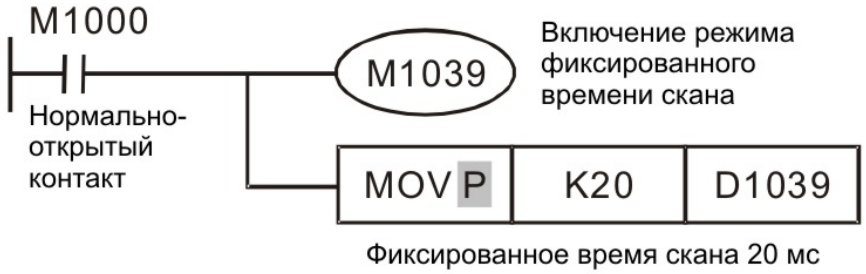
1. Не исправляйте программу в режиме on-line, когда в ней содержится прерывание I170.
4. При использовании прерывания I170 время скана немного увеличится.

2.11.21 Время задержки коммуникационного ответа (D1038)

Когда ПЛК является Ведомым в сети, для порта COM2 (RS485) можно установить задержку ответа Мастеру по времени. Диапазон 0 ~ 10000, ед. 0,1 мс.
Время задержки должно быть меньше, чем уставка сторожевого таймера WDT, записанная в регистр D1000.
В режиме EASY PLC LINK в данном регистре можно выставить задержку передачи следующего пакета данных, ед. время одного скана для SA/SX/SC, и 0,1 мс для EH/EH2/SV.

2.11.22 Фиксированное время скана (M1039, D1039)

Если M1039 включено, то время исполнения одного цикла программы (скана) будет всегда одинаковым. Время скана определяется значением в регистре D1039. Если реальное время скана больше того, что указано в регистре D1039, то продолжительность скана будет соответствовать реальному, а не записаному в D1039. Значение записывается в виде константы с шагом в 1 мс импульсным вариантом команды MOV (MOVP).



Фиксированное время скана 20 мс

Инструкции, связанные со временем скана – RAMP (API 67), HKY (API 71), SEGL (API 74), ARWS (API 75) PR и (API 77) – должны использоваться с фиксированным временем скана или постоянным прерыванием. В частности, для инструкции HKY при использовании 16-ти клавишной клавиатуры получается матрица 4x4 и время скана должно быть более 20 мс.
Время, отображаемое в регистрах D1010 ~ D1012, включает в себя фиксированное время скана.

2.11.23 Аналоговые сигналы (D1056~D1059, D1062, D1110~D1113, D1116~D1118)

Ряд контроллеров оснащены аналоговыми входами и выходами, которые имею свою разрядность. Текущее значение на аналоговых входах и выходах отображается в соответствующих специальных регистрах.

Разрядность АЦП (входы) составляет:

1. У контроллеров EX 10 бит, что соответствует следующим диапазонам: 0 ~ +/-10 V (-512 ~ +511) или 0 ~ +/-20 mA (-512 ~ +511).
2. У контроллеров SX для потенциального режима 12 бит: 0 ~ +/-10 V (-2000 ~ +2000), для токового режима 11 бит: 0 ~ +/-20 mA (-1000 ~ +1000).
3. У карты расширения контроллеров EH/EH2 для потенциального режима 12 бит: 0 ~ 10 V (0 ~ +4000), для токового режима 11 бит 0 ~ 20 mA (0 ~ +2000).

Разрядность ЦАП (выходы) составляет:

1. У контроллеров EX 8 бит, что соответствует следующим диапазонам: 0 ~ 10 V (0 ~ +255) или 0 ~ 20 mA (0 ~ 255).
2. У контроллеров SX 12 бит, что соответствует следующим диапазонам: 0 ~ +/-10 V (-2000 ~ +2000), 0 ~ +/-20 mA (-2000 ~ +2000).
3. У карты расширения контроллеров EH/EH2 12 бит: 0 ~ 10 V (0 ~ +4000), 0 ~ 20 mA (0 ~ +2000).

Время дискретизации АЦП задается в регистре D1118, ед. мс. По умолчанию стоит значение 5 мс. Если задать меньше 5 мс, то автоматически будет задано значение 5 мс.

Сводная таблица специальных регистров, в которых отображается текущее значение на аналоговых входах или задается значение для аналоговых выходов:

Регистр	Функция
D1056	Текущее значение на аналоговом входе CH0 контроллеров EX/SX и карты расширения EH/EH2.
D1057	Текущее значение на аналоговом входе CH1 контроллеров EX/SX и карты расширения EH/EH2.
D1058	Текущее значение на аналоговом входе CH2 контроллеров EX.
D1059	Текущее значение на аналоговом входе CH3 контроллеров EX.
D1062	Количество замеров при осреднении значения на аналоговых входах контроллеров SX. Диапазон 2 ~ 4 замера.
D1110	Среднее значение на аналоговом входе CH0 контроллеров EX/SX и карты расширения EH/EH2.
D1111	Среднее значение на аналоговом входе CH1 контроллеров EX/SX и карты расширения EH/EH2.
D1112	Среднее значение на аналоговом входе CH2 контроллеров EX.
D1113	Среднее значение на аналоговом входе CH3 контроллеров EX.
D1116	Текущее значение на аналоговом выходе CH0 контроллеров EX/SX и карты расширения EH/EH2.
D1117	Текущее значение на аналоговом выходе CH1 контроллеров EX/SX и карты расширения EH/EH2.
D1118	Время дискретизации АЦП контроллеров EX/SX/EH/EH2.

2.11.24 Флаги ошибок алгоритма программы (M1067~M1068, D1067~D1068)

Флаги ошибок алгоритма

Устройство	Описание	STOP →RUN	RUN →STOP
M1067	Флаг ошибки алгоритма программы	Сброс	Сохраняется
M1068	Флаг фиксации ошибки алгоритма программы	Сохраняется	Сохраняется
D1067	Код ошибки алгоритма программы	Сброс	Сохраняется
D1068	Шаг ошибки алгоритма программы	Сохраняется	Сохраняется

Коды ошибок алгоритма

Код ошибки в D1067	Описание
0E18	Ошибка преобразования BCD
0E19	Деление на ноль
0E1A	Значение выходит за границы диапазона (включая E/F).
0E1B	Значение квадратного корня отрицательное
0E1C	Ошибка коммуникации FROM/TO

2.11.25 Сигнал о низком напряжении (M1087, D1100)

Если контроллер обнаружит сигнал о низком напряжении (LV – Low Voltage), и если реле M1087 будет включено, то содержимое регистра D1100 будет передано по битам на выходы Y0~Y17. Младший бит (bit 0) будет соответствовать выходу Y0, бит 1 выходу Y1, бит 8 выходу Y10 и т.д.

2.11.26 Файловые регистры (M1101, D1101~D1103)

При подаче питания, контроллер проверяет разрешение на автоматическую передачу данных из файловых регистров (F) в обычные регистры для хранения данных (D) в соответствии с условиями, обозначенными в нижеприведенных реле и регистрах:

M1101	Разрешение на автоматическую передачу данных из файловых регистров в обычные (M1101=1, разрешено)
D1101	Адрес начального файлового регистра K0 ~ K1600 для SA/SX/SC, K0 ~ K8000 для EH/EH2/SV
D1102	Количество считываемых файловых регистров K0 ~ K1600 для SA/SX/SC, K0 ~ K8000 для EH/EH2/SV
D1103	Начальный адрес регистра D для хранения данных, скопированных из файловых регистров: K2000 ~ K4999 для SA/SX/SC, K2000 ~ K9999 для EH/EH2/SV

См. также инструкции API 148 MEMR и API 149 MEMW.

2.11.27 Функциональная карта с DIP-переключателями (M1104 ~ M1111)

Если контроллер DVP-EH/EH2 используется с функциональной картой DVP-F8ID, имеющей 8 микропереключателей, то их состояние фиксируется в специальных реле M1104 – M1111.

Подробнее см. описание инструкции SWRD (API 109).

Если контроллер DVP-EH/EH2 используется с функциональной картой дискретных оптоизолированных входов DVP-F4IP, то состояние входов AX0 ~ AX3 будут отображать реле M1104 ~ M1107.

2.11.28 Функциональная карта транзисторных выходов (M1112, M1113)

Если контроллер DVP-EH/EH2 используется с функциональной картой дискретных выходов DVP-F2OT, то состояние выхода AY0 отображает реле M1112, а выхода AY1 реле M1113.

2.11.29 Импульсный выход с функцией ускорения/замедления (M1115~M1119, D1104)

Данная функция действует только для контроллеров SA/SX и SC до версии SC_V1.4.
Ниже приведена таблица с расшифровкой назначения регистров и реле для выдачи

импульсов с функцией ускорения/замедления (разгона/торможения):

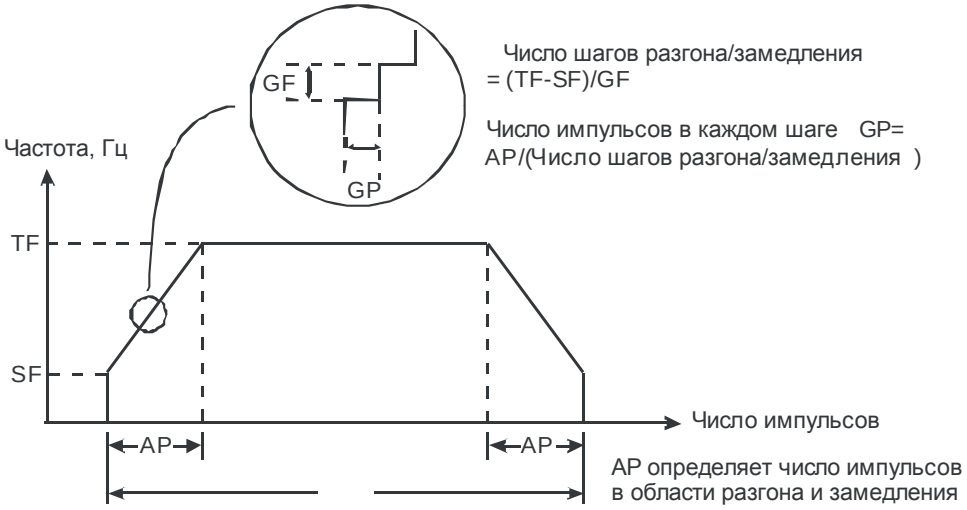
Операнд	Функция
M1115	Запуск импульсного выхода
M1116	Флаг режима ускорения
M1117	Флаг достижения заданной частоты
M1118	Флаг режима замедления
M1119	Флаг завершения одного цикла выдачи импульсов
D1104	Адрес начального регистра D для задания параметров ускорения/замедления

Параметры импульсного выхода с функцией ускорения/замедления задаются в семи последовательных регистрах D с начальным адресом, указанным в D1104. Допустимый диапазон 25 Гц ~ 10 кГц.

Регистр	Функция	
Начальный, заданный в D1104	Начальная частота (SF)	
+1	Шаг частоты при ускорении/замедлении (GF)	
+2	Заданная частота в устоявшемся режиме (TF)	
+3	Младшие 16 бит полного количества импульсов за 1 цикл	(TP)
+4	Старшие 16 бит полного количества импульсов за 1 цикл	
+5	Младшие 16 бит количества импульсов в зоне ускорения/замедления	(AP)
+6	Старшие 16 бит количества импульсов в зоне ускорения/замедления	

Примечание.
32-х разрядные параметры (количество импульсов) задаются командой DMOV и записываются в младший регистр 0-15 бит, а старший регистр 16-31 бит при этом заполняется автоматически.

В режиме импульсного выхода с ускорением/замедлением может работать только Y0 и для этого не требуется применение специальных инструкций. Контроллер должен быть в режиме РАБОТА, нужно задать параметры согласно вышеуказанной таблицы и включить реле M1115, после чего начнется генерация импульсов на выходе Y0.
Ниже приведен рисунок, иллюстрирующий смысл параметров на рабочей характеристике:



Для начала работы данной функции необходимо установить все параметры (записать в регистры) и обязательно соблюсти при этом следующие условия:

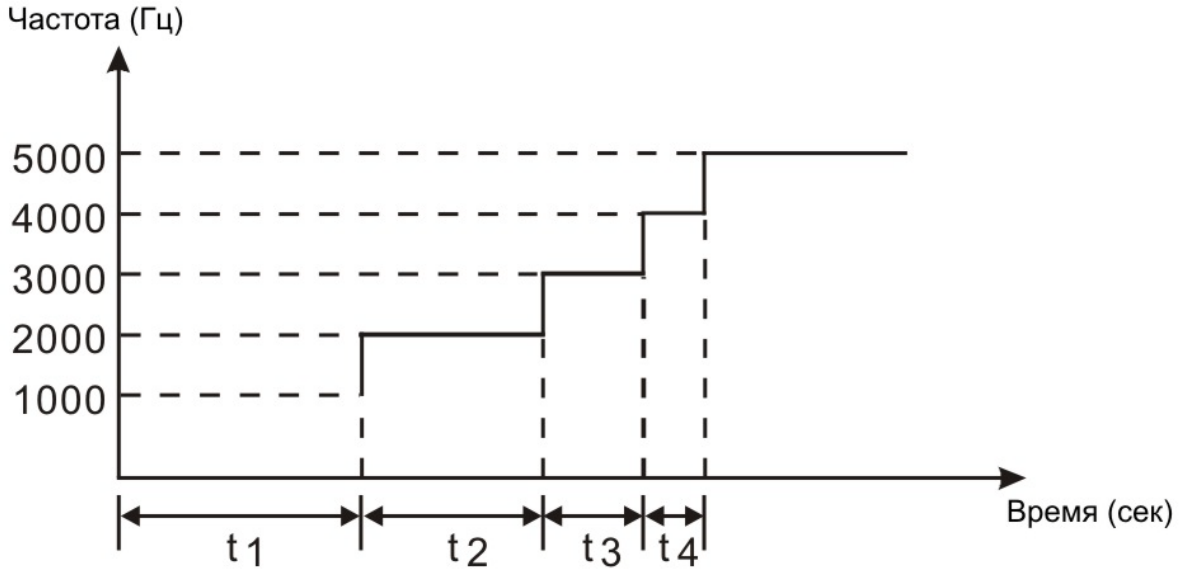
- Начальная частота (SF) должна быть меньше заданной (TF)
- Шаг частоты при ускорении/замедлении (GF) должен быть меньше или равен разности заданной и начальной частот
- Полное количество импульсов должно быть больше двойного количества импульсов в режиме ускорения/замедления.
- Допустимый диапазон для начальной и заданной частоты: минимально 25 Гц, максимально 10 кГц.
- Число импульсов ускорения/замедления должно быть больше, чем число шагов ускорения/замедления.

Когда M1115 выключается, M1119 сбрасывается, а M1116-M1118 остаются без изменений. При переводе контроллера из СТОПа в режим РАБОТА реле M1115-M1119 сбрасываются. Регистр D1104 сбрасывается на ноль только при снятии и повторной подаче питания на ПЛК.

Если функция "импульсный выход с ускорением/замедлением" и команда PLSY Y0 используются одновременно, то будет выполняться только одна из инструкций, запущенная первой.

Продолжительность каждого этапа выдачи импульсов можно вычислить следующим образом:

Предположим начальная частота задана в 1 кГц, заданная частота 5 кГц, шаг частоты при ускорении/замедлении 1 кГц, полное число импульсов 100, число импульсов при ускорении/замедлении 40. Рисунок ниже иллюстрирует данные параметры:



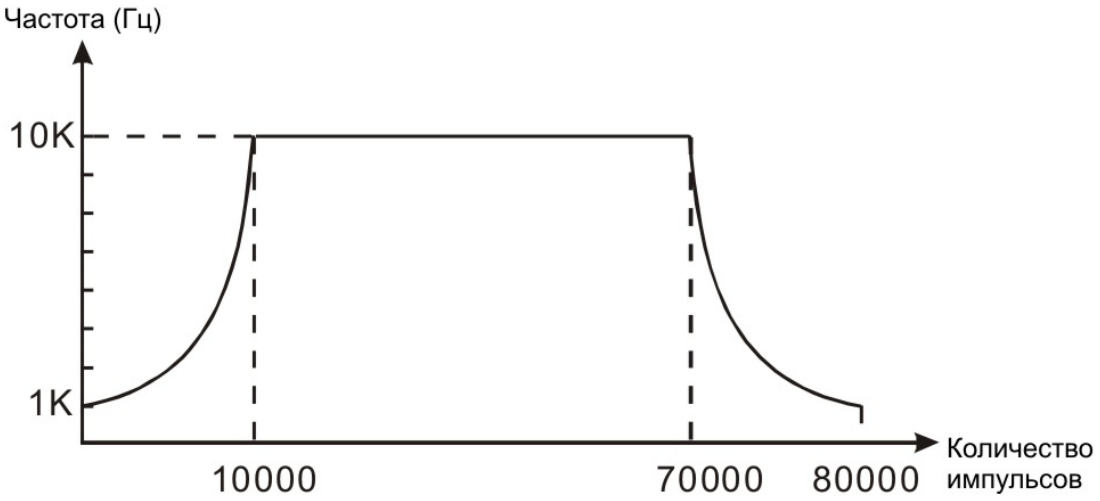
Расчет промежутков времени для заданных выше параметров:
Количество шагов при ускорении/замедлении – (5К – 1К)/1К = 4.
Число импульсов в каждом шаге – 40/4 = 10.

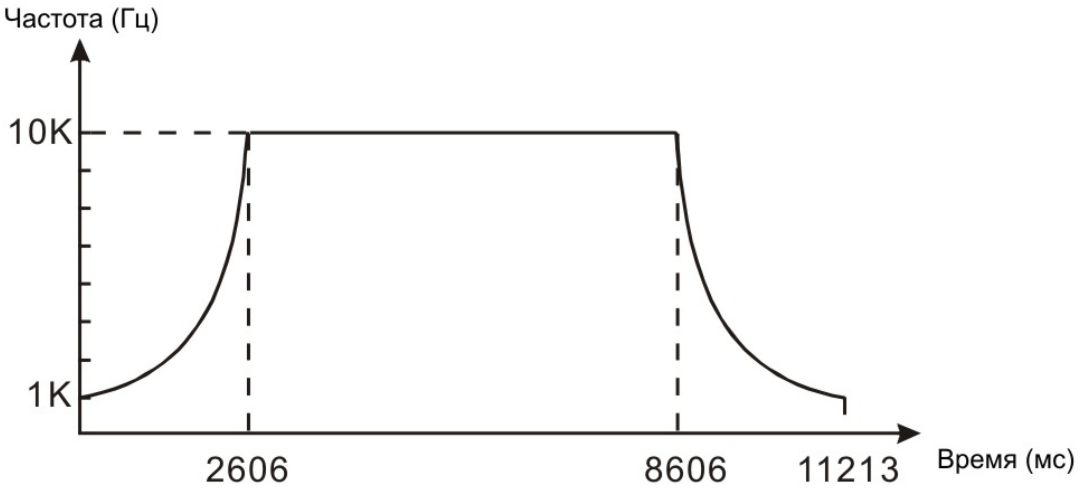
Таким образом, промежутки времени получаются:
t1 = (1/1К)x10 = 10 мс, t2 = (1/2К)x10 = 5 мс, t3 = (1/3К)x10 = 3,33 мс, t4 = (1/4К)x10 = 2,5 мс

Пример 1
Управление шаговым двигателем с ускорением/замедлением

Комментарии.
Когда контроллер переводят в режим РАБОТА, замыкается реле M1002 и указанные параметры записываются в отведенный диапазон регистров памяти, в данном примере D500 ~ D506. При активации реле M1115 на выходе Y0 появляются импульсы согласно заданных параметров. По отработке цикла реле M1115 сбрасывается, поэтому для повторного запуска цикла необходимо предусмотреть в программе отдельное условие включение M1115.
В ходе исполнения цикла будут последовательно включатся следующие реле:
M1116 – в ходе ускорения, M1117 при достижении заданной частоты, M1118 – в ходе замедления, M1119 – по завершению цикла.

Далее приведены графики, иллюстрирующие данный процесс:





Время разгона ~ 2,6 сек, работа в устоявшемся режиме 6,0 сек, замедление 2,6 сек, общее время цикла ~ 11,2 сек.

2.11.29 Специальный высокоскоростной выход (M1133 ~ M1135, D1133)

Данная функция действует только для выхода Y0 (до 50 кГц) контроллеров SA/SX и SC до версии SC_V1.4. Ниже приведена таблица с расшифровкой назначения регистров и реле для организации высокоскоростного выхода:

Операнд	Функция
M1133	Запуск высокоскоростного импульсного выхода
M1134	Флаг разрешения непрерывной выдачи импульсов высокоскоростным выходом
M1135	Флаг окончания выдачи заданного количества импульсов
D1133	Адрес начального регистра D для задания параметров высокоскоростного выхода

Параметры высокоскоростного выхода задаются в шести последовательных регистрах D с начальным адресом, указанным в D1133. Максимально 50 кГц.

Регистр	Функция
Начальный, заданный в D1133	Младшие 16 бит заданной частоты
+1	Старшие 16 бит заданной частоты
+2	Младшие 16 бит количества импульсов за 1 цикл
+3	Старшие 16 бит количества импульсов за 1 цикл
+4	Младшие 16 бит текущего количества импульсов
+5	Старшие 16 бит текущего количества импульсов

Примечание.
32-х разрядные параметры задаются командой DMOV и записываются в младший регистр 0-15 бит, а старший регистр 16-31 бит при этом заполняется автоматически.

Параметры, указанные в таблице выше, можно менять при включенных реле M1133 и M1135.

Однако, новые уставки вступят в действие только со следующего цикла выдачи импульсов. При включении реле M1133 программа считывает параметры из регистров, а потом обнуляет их значение. Когда реле M1133 выключается, в регистрах будет указано последнее значение количества выданных импульсов.

Функция высокоскоростного выхода применима только к Y0 и контроллер должен быть в состоянии РАБОТА (RUN). В программе допускается использование инструкции PLSY, но одновременное выполнение с функцией высокоскоростного выхода невозможно. Выполняться будет та инструкция, которая была запущена раньше. Преимущество функции высокоскоростного выхода перед инструкцией PLSY заключается в большей выходной частоте.

Когда работает функция высокоскоростного выхода, основная функция Y0 как дискретного выхода будет недоступна. Выходы Y1 ~ Y7 будут работать как обычно.

2.11.30 Синхронное перемещение по 2-м осям (M1133, M1135, D1133 ~ D1136)

Данная функция действует только для выходов Y10 и Y11 контроллеров SC с версии SC_V1.4 и выше. Доступно линейное и дуговое синхронное перемещение. Ниже приведена таблица с расшифровкой назначения регистров и реле для организации синхронного управления по двум осям:

Операнд	Функция
M1102	Флаг окончания выдачи импульсов выходом Y10
M1103	Флаг окончания выдачи импульсов выходом Y11
M1133	Запуск выхода Y10
M1135	Запуск выхода Y11
D1133	Адрес начального регистра D для задания параметров выхода Y10
D1134	Количество участков для выхода Y10
D1135	Адрес начального регистра D для задания параметров выхода Y11
D1136	Количество участков для выхода Y11

Перемещение по каждому выходу задается путем разбиения траектории на участки, и для каждого участка задаются свои параметры – выходная частота и количество импульсов.
Максимальное количество участков – 50, минимальное – 1. При других значениях количества участков функция синхронного перемещения отключится.

Перемещение по оси X:
Выход Y0 – определяет направление перемещения, а выход Y10 выдает требуемое количество импульсов с заданной частотой.

Перемещение по оси Y:
Выход Y1 – определяет направление перемещения, а выход Y11 выдает требуемое количество импульсов с заданной частотой.

Параметры для перемещения задаются отдельно по каждой оси в последовательных

регистрах, начальный адрес которых задается в D1133 (ось X) и в D1135 (ось Y) в формате 32-х разрядного числа. Следовательно, описание одной секции одной оси требует 4 регистра памяти D (два регистра для частоты и два для количества импульсов).

Смысловая последовательность регистров будет следующая:

Последовательность регистров	Что записывается
0	Младшие 16 бит частоты на выходе Y10 (Y11) на таком-то участке
1	Старшие 16 бит частоты на выходе Y10 (Y11) на таком-то участке
2	Младшие 16 бит количества импульсов на выходе Y10 (Y11) на таком-то участке
3	Старшие 16 бит количества импульсов на выходе Y10 (Y11) на таком-то участке

Примечание.
32-х разрядные параметры задаются командой DMOV и записываются в младший регистр 0-15 бит, а старший регистр 16-31 бит при этом заполняется автоматически.

Подобным образом последовательно описывается каждый участок по каждой оси.

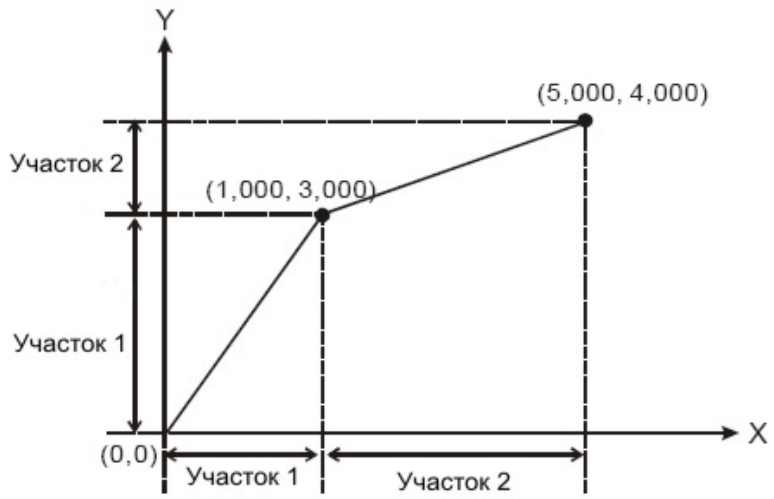
Например, запишем в качестве начального регистра в D1133 число K100 (регистр памяти D100), а количество секций в D1134 определим как K3 (кривая перемещения разбивается на 3 участка). Тогда установка параметров (при помощи команды DMOV) для каждого участка будет выглядеть следующим образом:

Номер участка	Регистр памяти	Частота на выходе Y10	Регистр памяти	Количество импульсов	Действия контроллера по выходу Y10
1	D101, D100	K10000	D103, D102	K1000	Выдаст 1000 импульсов с частотой 10 кГц
2	D105, D104	K15000	D107, D106	K2000	Выдаст 2000 импульсов с частотой 15 кГц
3	D109, D108	K5000	D111, D110	K3000	Выдаст 3000 импульсов с частотой 5 кГц

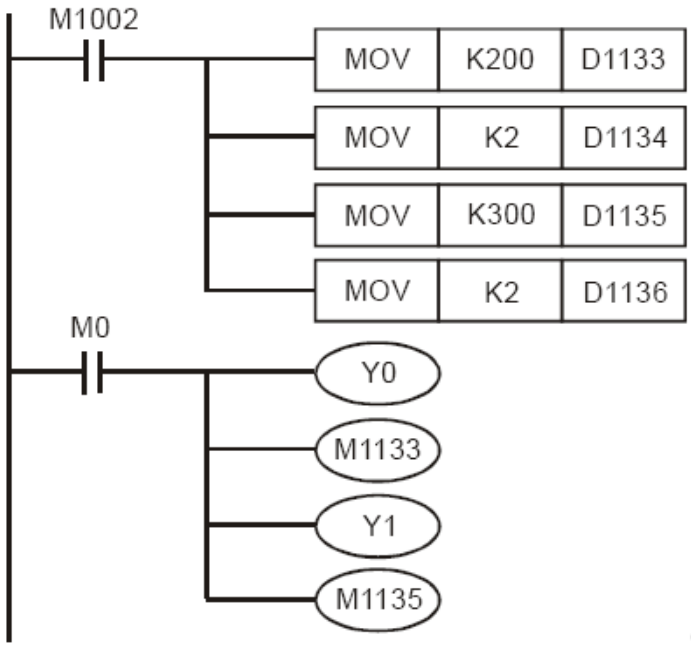
Условия, необходимые для корректной работы функции синхронного перемещения по 2-м осям:

- Все параметры должны быть записаны в соответствующие регистры до запуска данной функции.
- Менять параметры в ходе исполнения перемещений нельзя.
- После исполнения одного цикла перемещений данную функцию необходимо запускать заново (при необходимости).
- Для синхронной работы выходов Y10 и Y11 реле M1133 и M1135 должны быть активированы в одном скане.
- Частота на выходах не может быть ниже 10 0 Гц. Если задать меньше 1 0 0Гц, то выходы отработают с частотой 100 Гц.
- Частота на отдельном выходе не может быть выше 100 кГц. Если задать больше, то выход отработает с частотой 100 кГц.
- Для задания параметров можно использовать регистры D0 ~ D999 и D2000 ~ D4999. Т.е. нельзя использовать специальные регистры и задавать несуществующие номера регистров.

Пример 1
Перемещение по двум линейным участкам.



Перемещением по оси X управляют выходы Y0 и Y10, перемещением по оси Y управляют выходы Y1 и Y11. Слева приведен рисунок с результатом выполнения фрагмента программы, приведенного ниже. Значения параметров показаны в таблице.
На Участке 1 перемещение по оси X – 1000 имп., по оси Y – 3000 имп. На Участке 2 по X – 4000 имп., по Y – 1000 имп. Всего по оси X перемещение составляет 5000 имп., а по Y 4000 имп.



Комментарии:
Начальный регистр D200, (для параметров оси X)

Количество участков по оси X - 2

Начальный регистр D300, (для параметров по оси Y)
Количество участков по оси Y - 2

Движение вперед по оси X

Активация выхода Y10

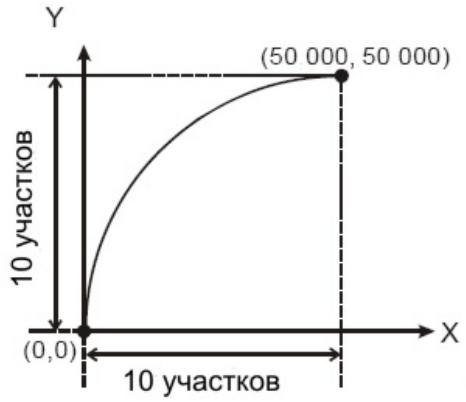
Движение вперед по оси Y

Активация выхода Y11

Ось	Участок	Регистр	Выход	Частота на выходе	Регистр	Количество импульсов
X	1	D201, D200	Y10	1000	D203, D202	1000
	2	D205, D204		4000	D207, D206	4000
Y	1	D301, D300	Y11	3000	D303, D302	3000
	2	D305, D304		1000	D307, D306	1000

Пример 2.
Построение дуги 90^{o} по двум осям в одном квадранте.

Для получения дуги необходимо перемещаться по осям X и Y путем последовательного и непрерывного построения ряда прямых участков небольшой длины, чтобы в итоге получить фигуру закругленной формы близкой по виду к дуге.

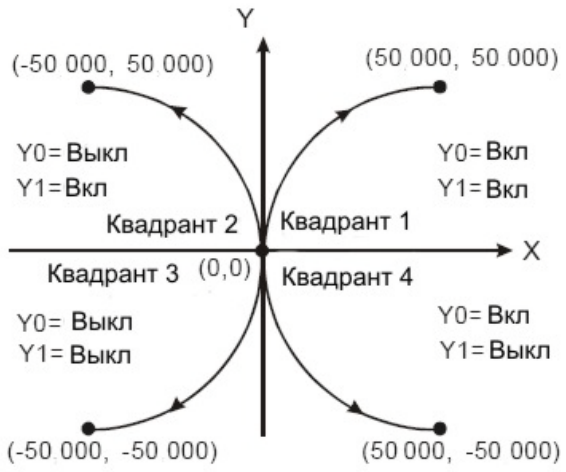


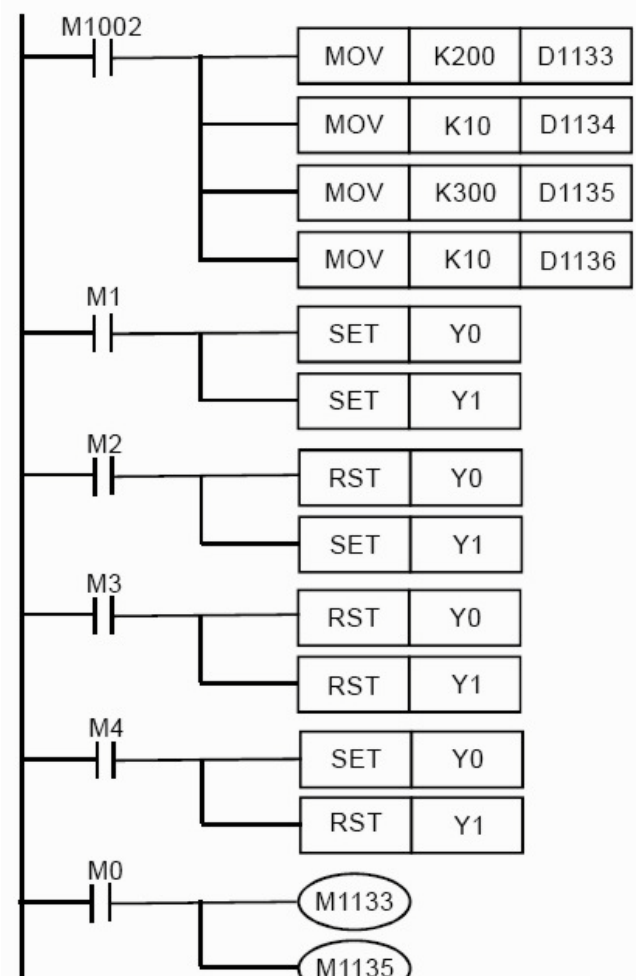
С этой целью можно использовать фрагмент программы из Примера 1, исправив количество участков в D1134 и D1136 на K10. Заполнение таблицы и рисунок с получившимся результатом приведены ниже:

Ось	Участок	Регистр	Выход	Частота на выходе	Регистр	Количество импульсов
X	1	D201, D200	Y10	1230	D203, D202	615
	2	D205, D204		3664	D207, D206	1832
	3	D209, D208		6004	D211, D210	3002
	4	D213, D212		8200	D215, D214	4100
	5	D217, D216		10190	D219, D218	5095
	6	D221, D220		11932	D223, D222	5966
	7	D225, D224		13380	D227, D226	6690
	8	D229, D228		14498	D231, D230	7249
	9	D233, D232		15258	D235, D234	7629
	10	D237, D236		15644	D239, D238	7822
Y	1	D301, D300	Y11	15644	D303, D302	7822
	2	D305, D304		15258	D307, D306	7629
	3	D309, D308		14498	D311, D310	7249
	4	D313, D312		13380	D315, D314	6690
	5	D317, D316		11932	D319, D318	5966
	6	D321, D320		10190	D323, D322	5095
	7	D325, D324		8200	D327, D326	4100
	8	D329, D328		6004	D331, D330	3002
	9	D333, D332		3664	D335, D334	1832
	10	D337, D336		1230	D339, D338	615

Пример 3.
Построение 4х- дуг по 90° в четырех квадрантах.

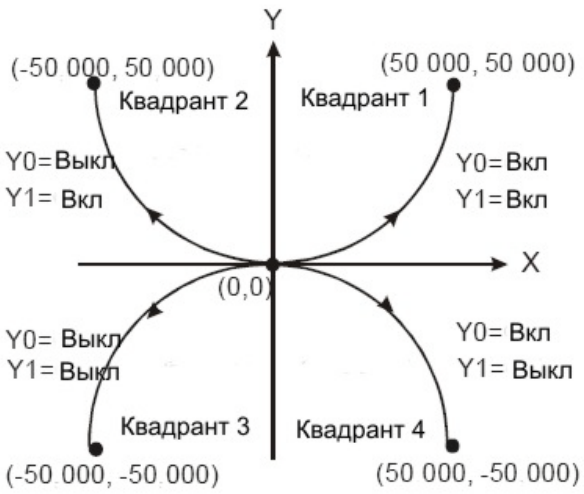
Для построения будем использовать параметры из Примера 2, но в программу необходимо добавить 4 цикла, в каждом из которых будет выбираться разное направление по осям X и Y путем включения/выключение выходов Y0 и Y1 соответственно.
Данный пример графически проиллюстрирован ниже:





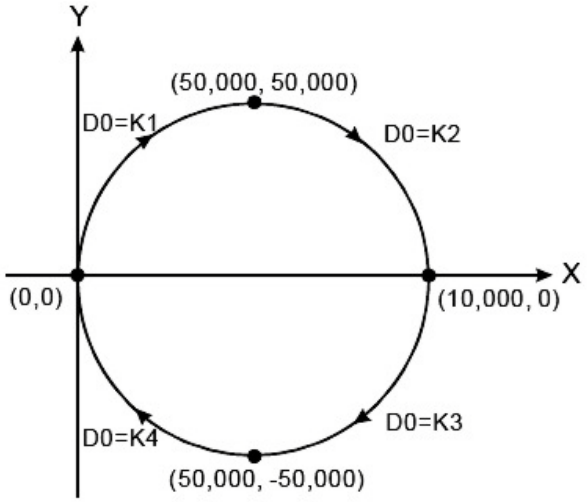
Комментарии:
Данные берутся из таблицы Примера 2.
Реле M0 активирует выходы Y10 Y11 через специальные реле M1133 и M1135 соответственно.
Реле M1-M4 определяют направление по осям X и Y путем включение/выключения выходов Y0 и Y1 соответственно.
При одновременном замыкании M0 и M1 будет нарисована дуга 90° в квадранте 1, при замыкании M0 и M2 в квадранте 2, M0 и M3 в квадранте 3, M0 и M4 в квадранте 4.

В разобранном варианте по оси X с каждым шагом идет увеличение частоты и количества импульсов (ускорение), а по оси Y наоборот (замедление). Если поменять местами данные, т.е. в регистр D1133 записать K300, а в регистр D1134 записать K200, то дуги построятся с ускорением по оси Y и замедлением по оси X, приняв вид как на рисунке справа.

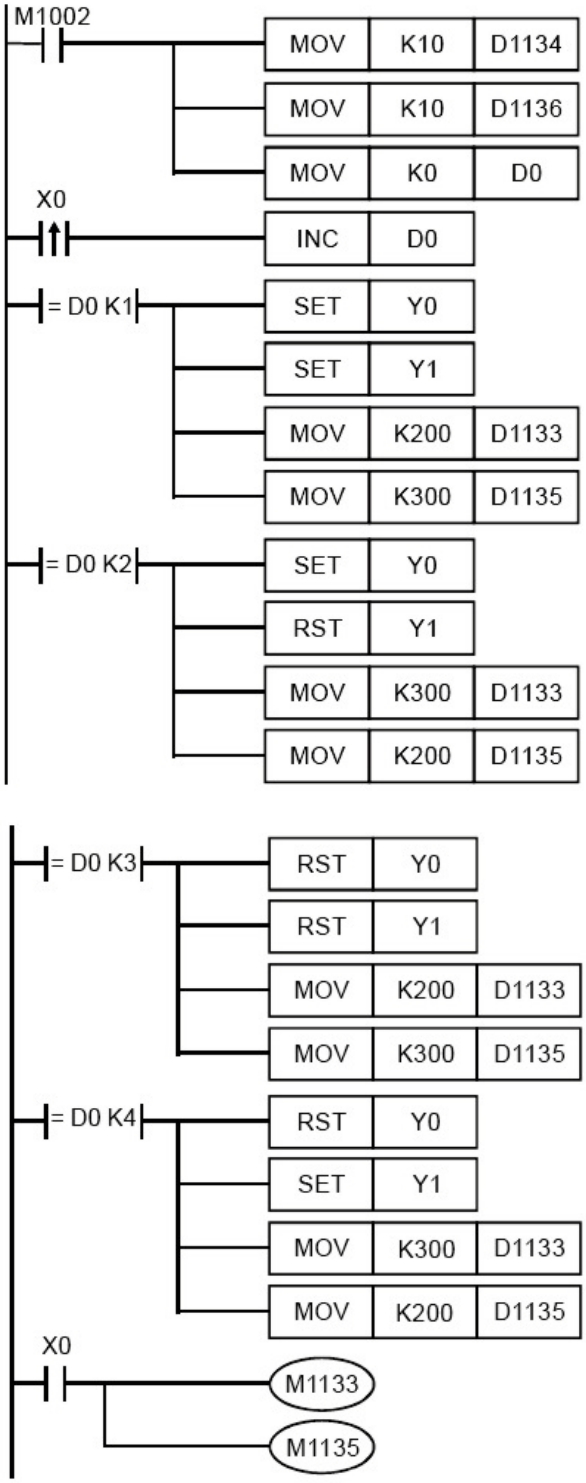


Пример 4
Построение окружности.

Параметры берутся из таблицы Примера 2. Окружность строится по четырем дугам 90° путем включения/выключения выходов Y0 и Y1 для определения направления по осям X и Y соответственно. Также, меняется ускорение/замедление по осям X и Y как в Примере 3.
Ниже приведен рисунок, иллюстрирующий данный пример, а также фрагмент программы с комментариями:

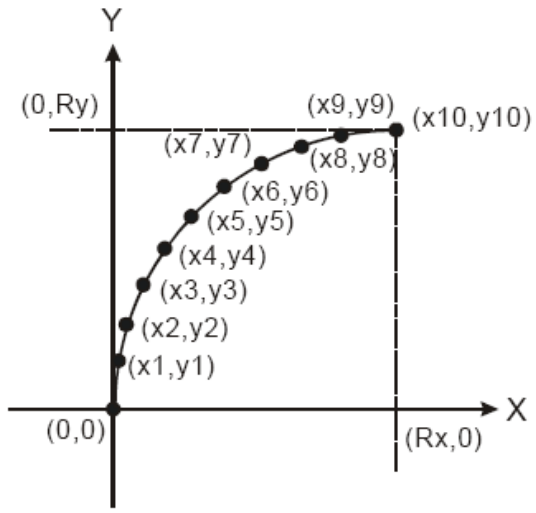


Комментарии:
Замыкание контакта X0 увеличивает значение в D0 на единицу, включая соответствующий участок программы, который строит свою дугу 90°.
В каждом программном блоке свое направление по осям X и Y, а также чередуются начальные регистры параметров. Таким образом получается окружность.



Пример 5

Расчет количества импульсов и частоты для построения дуги по 10-ти точкам в направлении по часовой стрелке в первом квадранте. Предельное значение по оси X - Rx=50000 импульсов, предельное значение по оси Y - Ry=50000 импульсов, количество точек обозначим как N, число ПИ=3,1416.



Шаг 1. Вычисление координаты каждой точки по осям X и Y

$x_1 = Rx - Rx \times \sin [(N-1) \times \pi \div (2 \times N)]$
$x_2 = Rx - Rx \times \sin [(N-2) \times \pi \div (2 \times N)]$ …. See table 3
$y_1 = Ry \times \sin [1 \times \pi \div (2 \times N)]$
$y_2 = Ry \times \sin [2 \times \pi \div (2 \times N)]$…. See table 4

Таблица 1

Координата	x_1	x_2	x_3	x_4	x_5
с дробной част.	615.55	2,447.12	5,449.61	9,549.08	14,464.59
без дробн. част.	615	2,447	5,449	9,549	14,464

Координата	x_6	x_7	x_8	x_9	x_{10} (R_x)
с дробной част.	20,610.67	27,300.42	34,549.11	42,178.25	50,000
без дробн. част.	20,610	27,300	34,549	42,178	50,000

Таблица 2

Координата	y_1	y_2	y_3	y_4	y_5
с дробной част.	7,821.74	15,450.88	22,699.57	29,389.32	35,355.40
без дробн. част.	7,821	15,450	22,699	29,389	35,355

Координата	y_6	y_7	y_8	y_9	y_{10} (R_y)
с дробной част.	40,450.91	44,550.38	47,552.87	49,384.44	50,000
без дробн. част.	40,450	44,550	47,552	49,384	50,000

Шаг 2. Расчет количества импульсов между точками

По оси X: $x_1 = x_1 – 0$, $x_2 = x_2 – x_1$, ... $x_{10} = x_{10} – x_9$ (см. Таблицу 3)
По оси Y: $y_1 = y_1 – 0$, $y_2 = y_2 – y_1$, … $y_{10} = y_{10} – y_9$ (см. Таблицу 3)

Таблица 3

Точка	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
Имп.	615	1,832	3,002	4,100	5,095	5,966	6,690	7,249	7,629	7,822
Точка	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
Имп.	7,821	7,629	7,249	6,690	5,966	5,095	4,100	3,002	1,832	616

Шаг 3. Расчет частоты для каждого участка при перемещении от точки к точке

Предположим, что прохождение от точки к точке должно осуществляться за 500 мс, тогда формулой для расчета частоты в Гц для прохождения каждого участка будет следующее уравнение:
$fx_1 = 1 \div 0.5 \times x_1$
$fx_2 = 1 \div 0.5 \times x_2$…(см. Таблицу 4)

Точка	fx_1	fx_2	fx_3	fx_4	fx_5	fx_6	fx_7	fx_8	fx_9	fx_{10}
Частота	1,230	3,664	6,004	8,200	10,190	11,932	13,380	14,498	15,258	15,644
Точка	fy_1	fy_2	fy_3	fy_4	fy_5	fy_6	fy_7	fy_8	fy_9	fy_{10}
Частота	15,642	15,258	14,498	13,380	11,932	10,190	8,200	6,004	3,644	1,232

Шаг 4.
Заполняем таблицу как в Примере 2 и переносим в регистры контроллера.

Примечание.

Если количество импульсов по оси X и Y совпадает, то можно рассчитать шаги только для оси X, а для оси Y просто перенести наоборот, т.е y1=x10, y2=x9 и т.д.

Для построения против часовой стрелки необходимо поменять с начала в конец индексы точек.

2.11.31 Количество присоединенных дополнительных модулей расширения и входов/выходов (D1140, D1142, D1143, D1145)

D1140	Количество присоединенных правосторонних специальных модулей
D1142	Количество входов X всех присоединенных дискретных модулей расширения
D1143	Количество выходов Y всех присоединенных дискретных модулей расширения
D1145	Количество присоединенных левосторонних специальных модулей (только для модели SV)

2.11.32 Управляемый импульсный выход с функцией ускорения/замедления (M1144~M1149, M1154, D1030, D1031, D1144, D1154, D1155)

Данная функция действует только для выхода Y0 контроллеров SA/SX/SC и позволяет организовать выдачу заданного количества импульсов с ускорением/замедлением (разгоном/торможением) непрерывно для нескольких последовательных участков (максимально 10 участков).
Ниже приведена таблица с расшифровкой назначения соответствующих регистров и реле:

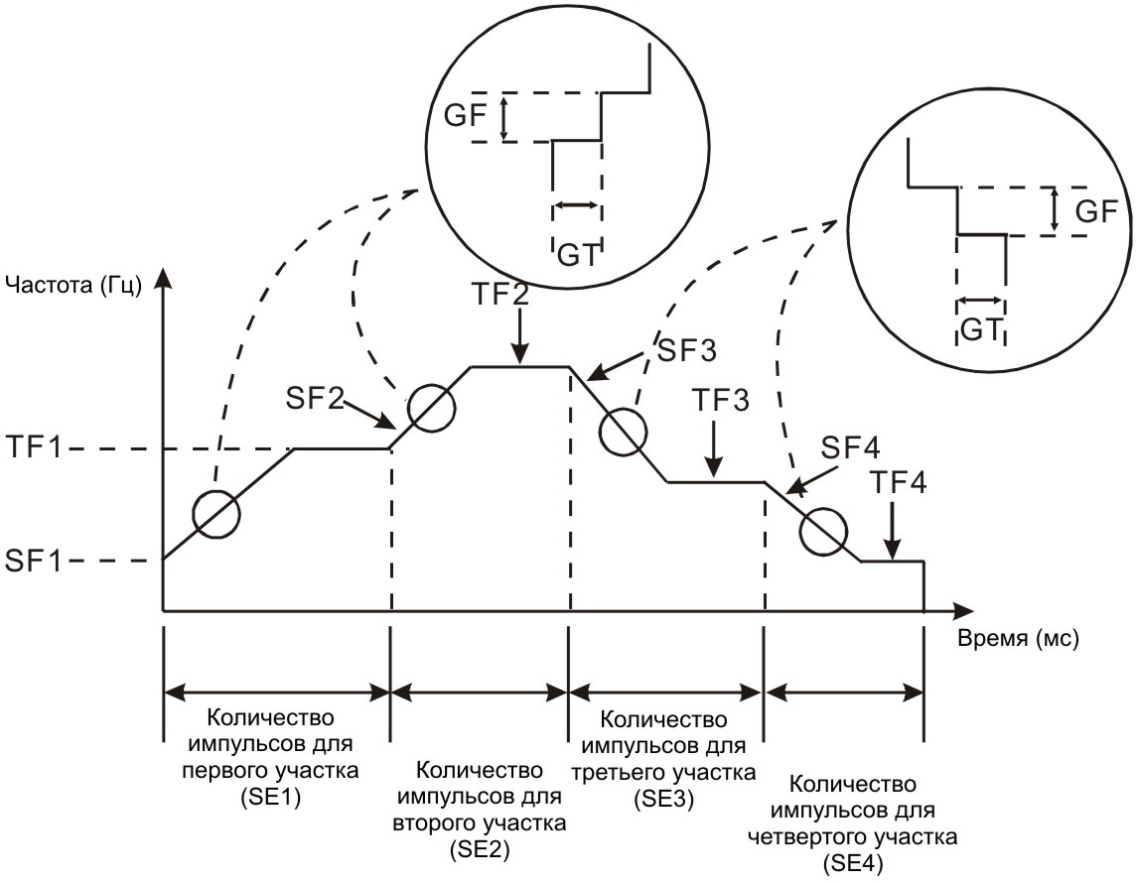
Операнд	Функция
M1144	Запуск функции выдачи заданного количества импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков. ПЛК должен быть в режиме РАБОТА.
M1145	Флаг ускорения для импульсного выхода Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.
M1146	Флаг достижения заданной частоты импульсным выходом Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.
M1147	Флаг замедления для импульсного выхода Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.
M1148	Флаг завершения цикла выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков импульсным выходом Y0.
M1149	Флаг временного прекращения подсчета импульсов на выходе Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.
M1154	Разрешение режима замедления для импульсного выхода Y0 с включенной функцией выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.
D1030	Число выходных импульсов для Y0 накопительным итогом при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков (младшее слово).
D1031	Число выходных импульсов для Y0 накопительным итогом при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков (старшее слово).
D1144	Начальный регистр D для задания параметров частоты/импульсов для каждого участка.
D1154	Рекомендованный интервал шага времени замедления (10 ~ 32767 ms) импульсного выхода Y0 при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков.
D1155	Рекомендованный интервал шага частоты замедления (-1 ~ -32700 ms) импульсного выхода Y0 при включенной функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких участков.

Параметры для каждого участка задаются в шести последовательных регистрах D. Регистры заполняются один за другим. Ниже в таблице показана последовательность заполнения и смысл каждого регистра. Под номером +0 идет начальный регистр, заданный в D1144, далее идет нумерация последующих регистров с использованием условного индекса +1, +2 и т.д. Т.е. к адресу начального регистра нужно прибавлять соответствующее число, чтобы получить

адрес регистра с требуемым параметром.

Регистр	Назначение
+0	Заданное количество участков (N), максимум 10.
+1	Номер участка, отрабатываемого в текущий момент (только чтение).
+2	Начальная частота для первого участка (SF1)
+3	Интервал шага времени при ускорении/замедлении на первом участке (GT1).
+4	Шаг частоты при ускорении/замедлении на первом участке (GF1)
+5	Заданная частота для устоявшего режима (горизонтальная линия) на первом участке (TF1)
+6	Младшие 16 бит заданного количества импульсов для первого участка (SE1)
+7	Старшие 16 бит заданного количества импульсов для первого участка (SE1)
+8	Начальная частота для второго участка (SF2). **Не может быть равной TF1 !!!**
+9	Интервал шага времени при ускорении/замедлении на втором участке (GT2).
+10	Шаг частоты при ускорении/замедлении на втором участке (GF2)
+11	Заданная частота для устоявшего режима (горизонтальная линия) на втором участке (TF2)
+12	Младшие 16 бит заданного количества импульсов для второго участка (SE2)
+13	Старшие 16 бит заданного количества импульсов для второго участка (SE2)
:	: (14 - 19 для третьего участка и т.д. по аналогии для N-го участка. При вычислении порядкового номера регистра параметра для N-го участка в формуле нужно использовать значение "N-1")
+(N-1)*6+2	Начальная частота для N-го участка (SFN). **Не может быть равной TF(N-1) !!!**
+(N-1)*6+3	Интервал шага времени при ускорении/замедлении на N-м участке (GTN).
+(N-1)*6+4	Шаг частоты при ускорении/замедлении на N-м участке (GFN)
+(N-1)*6+5	Заданная частота для устоявшего режима (горизонтальная линия) на N-м участке (TFN)
+(N-1)*6+6	Младшие 16 бит заданного количества импульсов для N-го участка (SEN)
+(N-1)*6+7	Старшие 16 бит заданного количества импульсов для N-го участка (SEN)

Смысл параметров объясняется на рисунке ниже и комментариях к нему:



На вертикальной оси отображается частота импульсов на выходе Y0 для каждом участке, по горизонтальной оси откладывается количество импульсов, которое выдаст выход Y0 на каждом участке. По данным параметрам можно получить время прохождения каждого участка (см. Пример 1).

Параметром SF обозначается начальная частота, с которой начинается новый участок кривой (наклонная линия). Может быть как больше заданной частоты предыдущего участка (ускорение), так и меньше (замедление), но не может быть ей равен.

Параметром TF обозначается заданная частота в устоявшемся режиме (горизонтальная линия на кривой).

Параметром GF обозначается шаг частоты в режиме ускорения/замедления. Это постоянная величина, с которой выход Y0 будет увеличивать/уменьшать частоту выдаваемых импульсов.

Параметром GT обозначается постоянный интервал времени, на который фиксируется частота в режиме ускорения/замедления после очередного шага своего изменения. Затем частота снова изменяется на один шаг и снова фиксируется на постоянный интервал времени, затем снова изменяется на один шаг и фиксируется, и т.д. до выхода на заданную частоту в устоявшемся режиме.
Использования параметров GF и GT позволяет ступенчато изменять частоту равными долями.

Условия, необходимые для корректной работы функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков:

- Начальная и заданная частоты должны быть не ниже 200 Гц. В противном случае функция отключится.
- Начальная и заданная частоты должны быть не выше 32700 Гц. Если установить большую частоту, то контроллер отработает на частоте 32700 Гц.
- Диапазон постоянного интервала времени (GT) при ускорении/замедлении 1 ~ 32767 мс (ед. при задании - мс).
- Диапазон шага частоты (GF) при ускорении 1 ~ 32700 Гц, при замедлении – 1 ~ – 32700. Если шаг частоты установить равным нулю, то на текущем участке заданная частота не будет достигнута. По истечении заданного количества импульсов текущего участка, программа перейдет к следующему и будет достигнута заданная частота этого участка (при условии, что у него шаг частоты не равен нулю).
- Заданное количество импульсов для участка должно быть больше значения, полученного по следующей формуле: (GF × GT/1000) × [(TF – SF)/GF]. В противном случае заданная частота может не быть достигнута ввиду недостатка числа импульсов. Если при расчете получается, что количество импульсов недостаточно, можно увеличить постоянный интервал времени (GT), или увеличить заданное количество импульсов.
- Если в программе помимо данной функции содержаться еще высокоскоростные инструкции, от они будут иметь приоритет в очередности исполнения.

Реакция системы на различные ситуации:

- Если после запуска функции включением реле M1144 участок не был пройден до конца (до включения реле M1148), а реле M1144 уже отключилось и реле M1154 было выключено, то включится режим замедления с параметрами "200 Гц каждые 200 мс" и включится реле M1147. Выход Y0 отключится, когда частота выходных импульсов станет ниже 200 Гц. Если реле M1154 было включено, то замедление произойдет с параметрами, установленными в D1154 и D1155 (аварийное замедление). Постоянный интервал шага времени при замедлении, установленный в D1154, не должен быть меньше либо равен нулю. В противном случае замедление будет происходить с интервалом, стоящим по умолчанию: 200 мс. Шаг частоты замедления, установленный в D1155, не должен быть выше или равен нулю. В противном случае замедление произойдет со следующими параметрами: при f=0 со значением по умолчанию – 1 кГц, при f>0 перед значением автоматически будет установлен знак минус.
- Если M1148 включено (цикл завершен), но M1144 выключено (новый цикл не запущен), то режим замедления нельзя будет включить, а реле M1148 будет сброшено.
- При отключении реле M1144 будет сброшено и реле M1149 (временная остановка подсчета импульсов на выходе Y0).
- Режим ускорения или замедления выбирается исходя из соотношения начальной и заданной частот текущего и следующего участка. Если начальная частота следующего участка больше, чем заданная частота текущего участка, то ПЛК осуществит ускорение и заданная частота следующего участка в данном случае должна быть больше, чем начальная. Если начальная частота следующего участка меньше, чем заданная частота текущего участка, то ПЛК осуществит замедление и заданная частота следующего участка в данном случае должно быть меньше, чем начальная.
- При переводе контроллера из режима СТОП в режим РАБОТА реле M1144 ~ M1149 сбрасятся. При переводе контроллера из режима РАБОТА в режим СТОП будет сброшено только реле M1144, а M1145 ~ M1149 сохранят свое состояние.
- Для задания параметров можно использовать регистры памяти следующих диапазонов: D0 ~ D999 и D2000 ~ D4999. Если задать регистр, выходящий за допустимый диапазон или не существующий, то функция не будет запущена и реле M1144 сбросится.

Далее приводятся несколько примеров применения функции выдачи импульсов с ускорением/замедлением непрерывно для нескольких последовательных участков.

Пример 1
Рассчитать количество импульсов, приходящееся на каждый этап исполнения функции выдачи импульсов для одного участка.

Предположим заданы следующие параметры:
Начальная частота – 200 Гц
Заданная частота – 500 Гц
Заданное количество импульсов на участке – 1000
Шаг частоты при ускорении – 100 Гц
Постоянный интервал времени между шагами ускорения – 100 мс, тогда получим:

Интервал времени для перехода от 200 Гц до 300 Гц – 100 мс, количество импульсов для перехода от 200 Гц до 300 Гц: 200 x 100 / 1000 = 20

Интервал времени для перехода от 300 Гц до 400 Гц – 100 мс, количество импульсов для перехода от 300 Гц до 400 Гц: 300 x 100 / 1000 = 30

Интервал времени для перехода от 400 Гц до 500 Гц – 100 мс, количество импульсов для перехода от 400 Гц до 500 Гц: 400 x 100 / 1000 = 40

Количество импульсов при поддержании заданной частоты: 1000 – (20 + 30 + 40) = 910

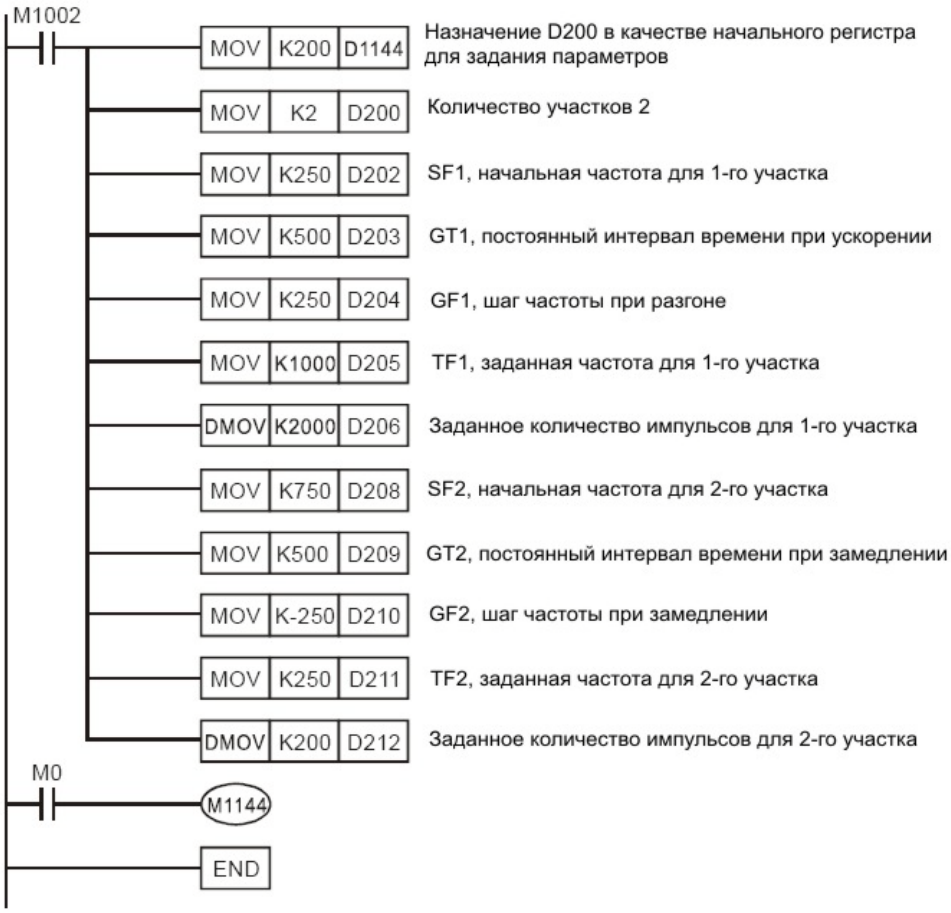
Время исполнения разгона: 3 x 100мс = 300 мс

Время исполнения устоявшегося режима: 1/500 Гц x 910 = 1820 мс

Общее время прохождения участка: 300 мс + 1820 мс = 2120 мс

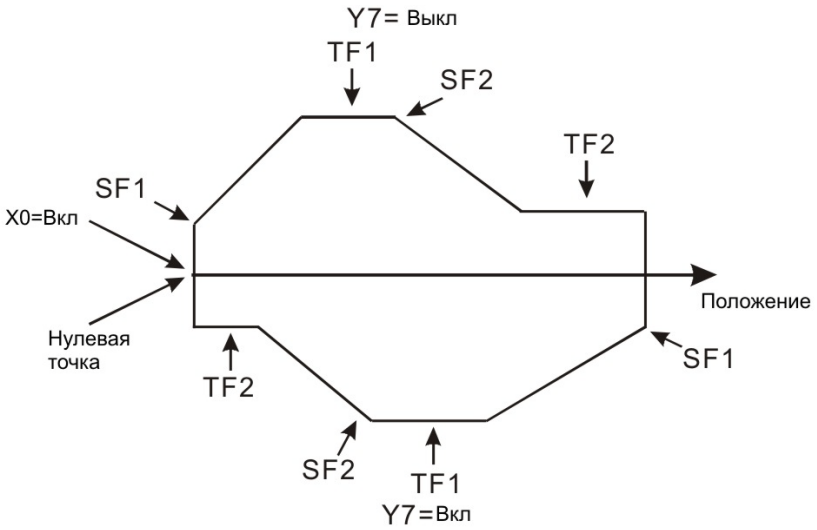
Примечание.
В общем случае количество импульсов для поддержания заданной частоты должно быть не менее 10.

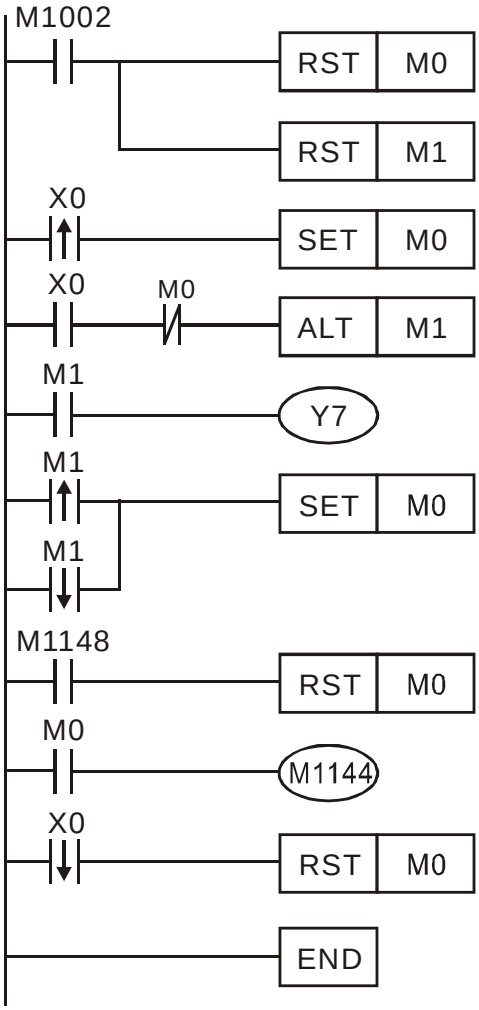
Пример 2
Фрагмент программы, реализующий 1 участок с ускорением и 1 участок с замедлением.



Пример 3
Фрагмент программы, реализующий 1 участок с ускорением и 1 участок с замедлением, с прямым и реверсивным ходом по замкнутой кривой.

Параметры берутся из Примера 2. Ниже приводится рисунок с графическим отображением перемещения и фрагмент программы, реализующий прямое и реверсивное движение.





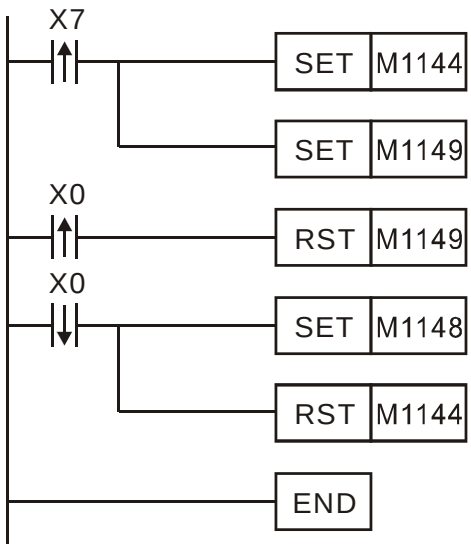
Комментарии:
При замыкании контакта X0 начнется цикличное исполнение данного фрагмента программы и начнется движение вперед-назад как показано на рисунке выше.

Выход Y7 управляет направлением движения.

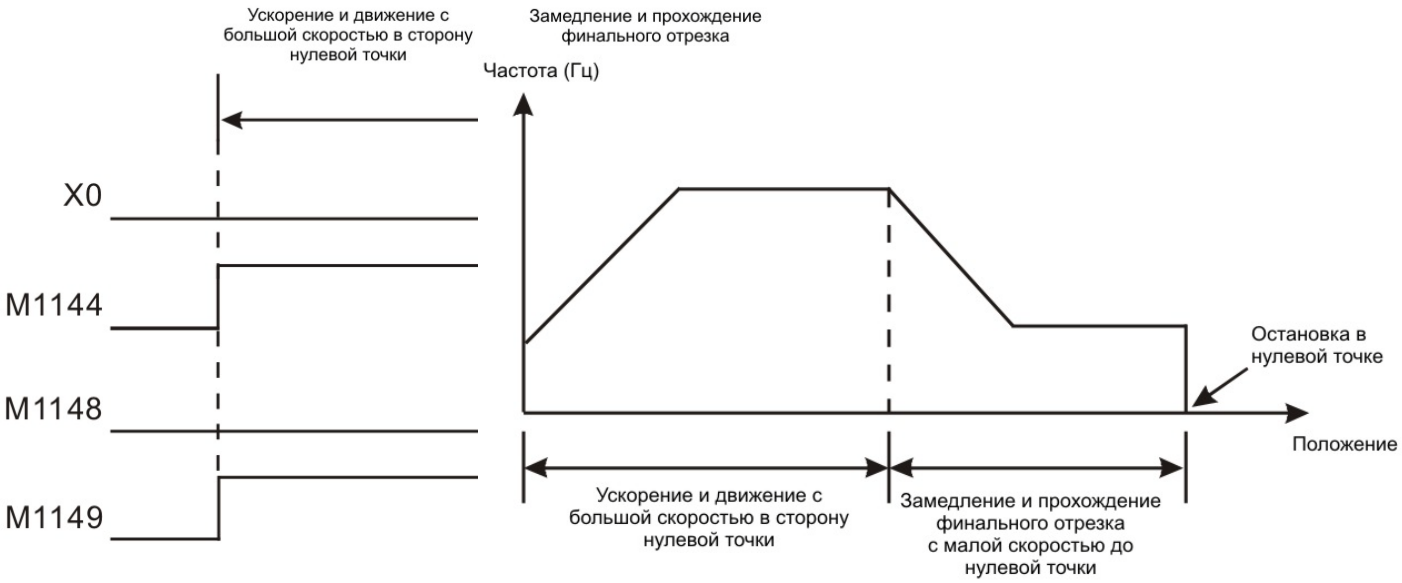
Пример 4.
Фрагмент программы, реализующий возвращение в ноль с 1 участком ускорения и 1 участком замедления.

Программа работает следующим образом:
При выходе в нулевую точку сначала осуществляется разгон на большую скорость, далее движение на большой скорости в направлении нулевой точки, при приближении к точке осуществляется замедление до малой скорости для прохождения финального отрезка перед остановкой в нулевой точке, скачкообразная остановка в нулевой точке.

Далее приводится диаграмма работы, фрагмент программы с комментариями, график перемещения и таблица с параметрами.



Комментарии:
Выход в нулевую точку начинается с замыкания контакта X7, который активирует M1144 (старт импульсного выхода Y0) и M1149 (отключение подсчета выходных импульсов).
После выполнения разгона до большой скорости, начинается движение в сторону нулевой точки без подсчета импульсов, что обеспечивает выход в ноль из любого положения.
Когда появляется передний фронт X0 (от датчика) сбрасывается M1149, отсчитываются заданные 10 импульсов (см. таблицу) и начинается замедление.
После замедления проходится финальный отрезок до нулевой точки с малой скоростью и, после появления заднего фронта X0 (от датчика), выключается M1144 и выход Y0 останавливается.
Соответственно останавливается и движение.



Начальный регистр D + индекс	Параметры
+0	2
+2	250 (Гц)
+3	100 (мс)
+4	500 (Гц)
+5	10000 (Гц)
+6, +7	10 (импульсов)
+8	9750 (Гц)
+9	50 (мс)
+10	-500 (Гц)
+11	250 (Гц)
+12, +13	30000 (импульсов)

2.11.33 Выполнение одного шага программы (M1170, M1171, D1170)

Данная функция доступна только для контроллеров типов EH/EH2/SV. Контроллер должен быть в режиме РАБОТА.
Функция активируется включением реле M1170 и контроллер переходит в режим пошагового выполнения программы. С каждым замыканием реле M1171 контроллер будет выполнять один шаг программы и снова останавливаться, сбрасывая реле M1171. При повторном включении реле M1171 будет выполнен следующий шаг программы, а реле M1171 сброшено и т.д. Номер текущего исполняемого шага программы хранится в регистре D1171.
Если в текущем шаге есть выход Y (катушка), то он будет включен немедленно в текущем шаге, не дожидаясь выполнения команды END.

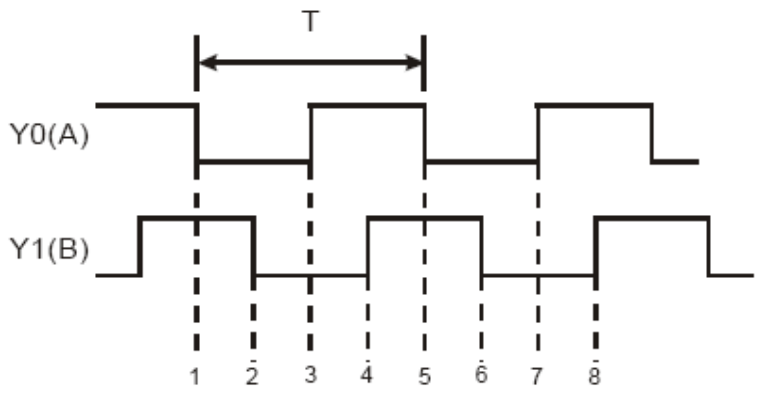
Важные замечания:

Поскольку программа выполняется не в режиме циклического сканирования ряд инструкций не будет работать, например HKY, так как для считывания состояния всех кнопок нужно 8 сканов, следовательно в режиме пошагового исполнения программы данные о состоянии кнопок будут неправильными.

И напротив, инструкции, выполняемые аппаратной частью, такие как высокоскоростной счет, высокоскоростные импульсные входы/выходы, инструкции высокоскоростного сравнения, будут исполняться корректно.

2.11.34 Двухфазный импульсный выход (M1172 ~ M1174, D1172 ~ D1177)

Данная функция действует только для выходов Y0 и Y1 контроллеров типов SA/SX/SC. Двухфазный выход может работать в двух режимах: К1 – фаза A (Y0) опережает фазу B (Y1), и К2 – фаза B опережает фазу A. По достижении заданного количества импульсов включается реле M1174. Для сброса текущего значения нужно отключить M1172.



Частота = 1/T
T – период одного импульса
Внутренний счетчик добавляет импульс к текущему значению каждый раз, когда фиксируется сдвиг фаз (см. рисунок далее).

Функции специальных реле и регистров:

Операнд	Функция
M1172	Запуск двухфазного импульсного выхода
M1173	Разрешение непрерывной выдачи импульсов
M1174	Флаг достижения заданного количества импульсов
D1172	Заданная частота импульсов (12 Гц ~ 20 Кгц)
D1173	Режим работы выхода (К1 или К2)
D1174	Младшие 16 бит заданного количества импульсов
D1175	Старшие 16 бит заданного количества импульсов
D1176	Младшие 16 бит текущего количества импульсов
D1177	Старшие 16 бит текущего количества импульсов

Заданную частоту, количество импульсов и режим работы можно менять, когда M1172=1, а M1174=0. Изменение параметров не повлияет на цикл выполнения с текущими параметрами, а вступят в действие со следующего цикла. Однако изменение режима работы сбросит на ноль текущее значение выданных импульсов.
Текущее значение импульсов обновляется в каждом скане. При включении M1133 текущее значение сбросится на ноль. При переводе контроллера из РАБОТЫ в СТОП сохраняется текущее значение выданных импульсов.
При переводе контроллера из СТОП в РАБОТУ реле M1172 сбрасывается.

Данная функция может быть одновременно находится в программе с инструкций PLSY, но одновременно выполняться они не могут. Приоритет будет у запущенной первой.

2.11.35 Текущее значение встроенных потенциометров (M1178, M1179, D1178, D1179)

Данная функция доступна только для контроллеров типов EH/EH2/SV/SA/SC.

Операнд	Функция
M1178	Запуск потенциометра VR0
M1179	Запуск потенциометра VR1
D1178	Текущее значение потенциометра VR0
D1179	Текущее значение потенциометра VR1

На лицевой панели контроллеров располагаются движки потенциометров VR0 и VR1. Данная функция преобразует физическое вращение движка потенциометра в числовое значение в регистре контроллера. Диапазон от 0 до 255.
При включении M1178 в регистре D1178 будет отображаться значение потенциометра VR0.
При включении M1179 в регистре D1179 будет отображаться значение потенциометра VR1.

2.11.36 Прерывание для считывания текущего значения скоростного счетчика (D1180 ~ D1181, D1198 ~ D1199)

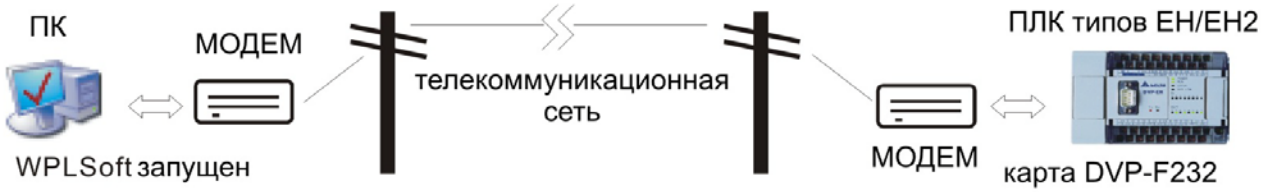
Данная функция действует только для контроллеров типов SA/SX/SC. Позволяет по сигналу от физического входа X осуществить перехват текущего значения скоростного 32-х разрядного счетчика и записать в D1180 ~ D1181, D1198 ~ D1199.

У контроллеров SA/SX вход X0 (входные импульсы) жестко связан с X4 (внешнее прерывание), номер прерывания I401, счетчики C235/C251/C253. Значение будет хранится в 32-х разрядном виде в регистрах D1180 ~ D1181. Вход X1 связан с X5 (внешнее прерывание), номер прерывания I501, счетчик C236. Значение будет хранится в 32-х разрядном виде в регистрах D1198 ~ D1199.

У контроллеров SC вход X10 (входные импульсы) жестко связан с X4 (внешнее прерывание), номер прерывания I401, счетчики C243/C255. Значение будет хранится в 32-х разрядном виде в регистрах D1180 ~ D1181. Вход X11 (входные импульсы) жестко связан с X5 (внешнее прерывание), номер прерывания I501, счетчик C245. Значение будет хранится в 32-х разрядном виде в регистрах D1198 ~ D1199.

2.11.37 Программирование через модемное соединение (M1184 ~ M1188)

Данная функция доступна только для контроллеров типов EH/EH2 и позволяет связаться с удаленным контроллером по модемному соединению из программной среды WPLSoft для закачки программы, мониторинга исполнения программы, внесения корректур. Данная функция доступна и в режиме РАБОТА и в режиме СТОП контроллера.



Порядок соединения:

1. Включить реле M1184 (разрешение модемного соединения)
2. Включить реле M1185 (разрешение инициализации модема)
3. Проверить успешность инициализации модема. Если M1187=1 – успешно, если M1186=1 – не успешно
4. Дождаться соединения

Сводная Таблица значений специальных реле для модемного соединения

Реле	Функция	Примечание
M1184	Разрешение модемного соединения	КогдаM1184=1 все нижеследующие действия разрешены
M1185	Запуск инициализации модема	После окончания инициализации реле сбросится (в ходе инициализации реле включено)
M1186	Инициализация не удалась	Когда M1185=1, M1186=0
M1187	Инициализация прошла успешно	Когда M1185=1, M1187=0.
M1188	Сигнализирует статус подключения модема	Когда M1188=1, модем подключен

Комментарии:

- Для соединение ПЛК с модемом необходимо, чтобы в контроллер была вставлена функциональная карта RS232. В противном случае все вышеупомянутые реле будут недоступны.
- После разрешения функции модема (M1184=1) сначала необходимо инициализировать модем путем включения реле M1185. Если ПЛК не сможет инициализировать модем, то функция автоответчика у модема не включится.
- После успешной инициализации модем автоматически переходит в режим автоответчика.
- Если удаленный ПК (с программной средой WPLSoft) отключается, то модем автоматически переходит в режим ожидания. При выключении модема в данном состоянии, при последующей подаче питания потребуется его повторная инициализация.
- Скорость передачи данных фиксировано установлена в 9600 бод. Другие скорости не допускаются и модем должен поддерживать скорость не ниже 9600 бод.
- Формат инициализации модема контроллером – ATZ или ATS0=1.
- Если контроллер не сможет инициализировать модем, можно использовать программный модуль "Супер терминал" в ПК, используя формат ATZ или ATS0=1.

2.11.38 Установка энергонезависимой области (D1200 ~ D1219)

Пользователь может регулировать объем энергонезависимой памяти по своему усмотрению. Установка параметров подробно описана в параграфе 2.1.

2.11.39 Принудительное программное включение физических входов X (M1304)

Если M1304 включено, то при помощи программатора или программного пакета WPLSoft можно программно включать физические входы (без подачи внешних сигналов на клеммы). Таким способом можно включать входы X0 ~ X1 7 У контроллеров SA/SX/SC соответствующие им светодиоды загораться не будут. У контроллеров EH/EH2/SV при программном включении входов будут загораться соответствующие им светодиоды на корпусе контроллера.

2.11.40 Режимы останова для высокоскоростных импульсных выходов (M1310, M1311, M1334, M1335, D1166, D1167, D1343 ~ D1353)

Данная функция доступна в контроллерах типа SC и применяется совместно с инструкциями DDRVI, DDRVA и PLSY.

Специальные регистры и реле для выбора режима останова импульсных выходов Y10 и Y11:

Операнд	Функция
M1334	Режим останова с плавным замедлением для выхода Y10
M1335	Режим останова с плавным замедлением для выхода Y11

M1310	Режим немедленного останова для выхода Y10
M1311	Режим немедленного останова для выхода Y11
D1166	Выбор режима счета по переднему или заднему фронту от контакта X10
D1167	Выбор режима счета по переднему или заднему фронту от контакта X11
D1343	Время ускорения/замедления для выхода Y10
D1353	Время ускорения/замедления для выхода Y11

Режим 1
Останов с плавным замедлением. Применим для инструкций DDRVI и DDRVA.

В используемой прикладной инструкции необходимо задать параметры ускорения (начальная и заданная частоты и т.д.), выбрать контакт для внешней активации останова.
Реле M1334 для выхода Y10 и M1335 для выхода Y11 должны быть выключены.
Время ускорения/замедления устанавливается в D1343 для Y10 и в D1353 для Y11.
В данном режиме характеристика будет иметь следующий вид:

Сплошные линии показывают штатную схему разгона/горизонтального участка/торможения, а пунктирной линией показан досрочный останов по внешнему сигналу с плавной характеристикой замедления.

Режим 2
Останов без плавного замедления. Применим для инструкций DDRVI, DDRVA и PLSY.

В используемой прикладной инструкции необходимо задать параметры ускорения (начальная и заданная частоты и т.д.), выбрать контакт для внешней активации останова.
Реле M1334 для выхода Y10 и M1335 для выхода Y11 должны быть включены. Если используется инструкция PLSY, то включать реле M1334 и M1335 нет необходимости, так как данная инструкция не использует ускорение/замедление в принципе.
В данном режиме максимальное время реакции от появления внешнего сигнала останова до выдачи последнего импульса составляет не более 1 скана.
В данном режиме характеристика будет иметь следующий вид:

Сплошные линии показывают штатную схему разгона/горизонтального участка/торможения, а пунктирной линией показан досрочный останов по внешнему сигналу без замедления.

Примечание
Реле M1334 и M1335 должны быть включены до активации останова внешним сигналом.

Режим 3
Немедленный останов. Применим для инструкций DDRVI, DDRVA и PLSY.

В данном режиме останов происходит за кратчайший период времени – не более одного импульса на выходе с момента появления внешнего сигнала останова. Выход Y10 работает только со входом X10, на который должен подаваться внешний сигнал останова, а выход Y11 работает только со входом X11.
Для активации Режима 3 необходимо включить реле M1310 для входа X10 и M1311 для входа X11. В регистрах D1166 и D1167 задается режим по переднему фронту (K0) или по заднему фронту (K1) входного импульса на входах X10 и X11 соответственно. Данные параметры должны быть установлены до активации останова. Входы X10 и X11 в данном случае категорически запрещается использовать как входы высокоскоростных счетчиков.

Функция работает следующим образом: при появлении сигнала на X10 на выходе Y10 мгновенно прекращается выдача импульсов. Максимальная задержка не более одного импульса с момента появления сигнала останова. Также работает вход X11 и выход Y11.
В данном режиме характеристика будет иметь вид как на рисунке слева.

Сплошные линии показывают штатную схему разгона/горизонтального участка/торможения, а пунктирной линией показан досрочный мгновенный останов по внешнему сигналу на X10 (X11).

2.11.41 Идентификационный номер правосторонних модулей расширения (D1320~D1327)

Данные специальные регистры доступны только для контроллеров типов EH/EH2/SV. Всего может быть подсоединено до 8 специальных модулей расширения. Их идентификационные номера будут храниться последовательно в восьми регистрах D1320 ~ D1327.

Идентификационные номера (ID) правосторонних модулей для контроллеров EH:

Название модуля	ID модуля (hex)	Название модуля	ID модуля (hex)
DVP04AD-H	H’0400	DVP01PU-H	H’0110
DVP04DA-H	H’0401	DVP01HC-H	H’0120
DVP04PT-H	H’0402	DVP02HC-H	H’0220
DVP04TC-H	H’0403	DVP01DT-H	H’0130
DVP06XA-H	H’0604	DVP02DT-H	H’0230

Идентификационные номера (ID) правосторонних модулей для контроллеров EH2:

Название модуля	ID модуля (hex)	Название модуля	ID модуля (hex)
DVP04AD-H2	H’6400	DVP01PU-H2	H’6110
DVP04DA-H2	H’6401	DVP01HC-H2	H’6120
DVP04PT-H2	H’6402	DVP02HC-H2	H’6220
DVP04TC-H2	H’6403	DVP01DT-H2	H’6130
DVP06XA-H2	H’6604	DVP02DT-H2	H’6230

Идентификационные номера (ID) правосторонних модулей для контроллеров SV:

Название модуля	ID модуля (hex)	Название модуля	ID модуля (hex)
DVP04AD-S	H’0088	DVP04PT-S	H’008A
DVP06AD-S	H’00C8	DVP04TC-S	H’008B
DVP02DA-S	H’0049	DVP06XA-S	H’00CC
DVP04DA-S	H’0089	DVP01PU-S	H'0110

2.11.42 Идентификационный номер левосторонних модулей расширения (D1386~D1393)

Данные специальные регистры доступны только для контроллеров типа SV. Всего может быть подсоединено до 8 специальных модулей расширения. Их идентификационные номера будут храниться последовательно в восьми регистрах D1386 ~ D1393.

Идентификационные номера (ID) левосторонних модулей для контроллеров SV:

Название модуля	ID модуля (hex)	Название модуля	ID модуля (hex)
DVP04AD-SL	H’4400	DVP01HC-SL	H’4120
DVP04DA-SL	H’4401	DVP02HC-SL	H’4220
DVP04PT-SL	H’4402	DVPDNET-SL	H’4130
DVP04TC-SL	H’4403	DVPEN01-SL	H'4050
DVP06XA-SL	H’6404	DVPMDM-SL	H’4040
DVP01PU-SL	H’4110		

2.11.43 Организация коммуникаций между устройствами DELTA с помощью технологии EASY PLC LINK

Коммуникационная технология EASY PLC LINK базируется на протоколе Modbus и позволяет достаточно простым способом организовать последовательный цикличный обмен данными между устройствами DELTA – контроллерами, частотными преобразователями, термоконтроллерами и сервоприводами.
Основными преимуществами технологии EASY PLC LINK являются:

- Возможность передавать простым способом крупные массивы данных между большим количеством разнородных устройств. Контроллеры серий SA/SX/SC, когда они являются Мастером сети, могут поддерживать связь с 16 Ведомыми устройствами, считывая/записывая в одном цикле до 16 регистров (по 16 бит каждый) в каждом устройстве. Контроллеры серий EH/EH2/SV, когда они являются Мастером сети, могут поддерживать связь с 32 Ведомыми устройствами, считывая/записывая в одном цикле до 100 регистров (по 16 бит каждый) в каждом устройстве. Для сравнения инструкция MODRD может в одном цикле считать максимум 6 регистров, а инструкция MODWR записать всего 1 слово (регистр) в одном устройстве.
- При использовании технологии EASY PLC LINK пользователь освобождается от необходимости организовывать разделение во времени обработки каждого коммуникационного запроса, все это осуществляется автоматически. Данный факт является большим преимуществом перед инструкциями MODRD, MODWR и RS, которые могут выполняться только по одной в каждом скане и пользователь вынужден самостоятельно принимать специальные меры в программе для разделения их по времени (например использовать шаговые реле).
- Пропадание связи с одним из Ведомых не влияет на связь с другими Ведомыми.
- Пользователь может контролировать процесс считывания/записи данных в каждом Ведомом путем отслеживания специальных флагов (например через панель оператора или индикацию на пульте).
- При использовании технологии EASY PLC LINK пользователь освобождается от составления длинных и сложных программ, пользуясь удобной и понятной процедурой организации связи по технологии EASY PLC LINK.

Процедура организации связи по технологии EASY PLC LINK осуществляется по следующим шагам:

1. В сети назначается Мастер ПЛК путем внедрения в его программу технологии EASY PLC LINK. В программах Ведомых устройств не должно содержаться никаких специальных регистров и реле, связанных с EASY PLC LINK, а также не должно содержаться никаких коммуникационных инструкций (MODRD, MODWR, RS и др.).
2. Всем устройствам присваиваются уникальные сетевые адреса Modbus (т.е. которые не должны повторятся). Ведомые устройства, которые предполагается объединить по технологии EASY PLC LINK, должны иметь последовательно возрастающую адресацию (2, 3, 4, 5 и т.д.). У контроллеров адрес записывается в регистр D1121 в десятичной форме, для других устройств определяется в соответствующих разделах системного меню.
3. Для всех устройств сети обязательно устанавливается одинаковый протокол связи. В качестве Мастера контроллеры поддерживают оба режима ASCII и RTU. В режиме Ведомого только ASCII и скорость до 38400 бит/сек. Для контроллеров протокол

записывается в регистр D1120 и фиксируется реле M1120 (порт COM2 RS485), для других устройств устанавливается в соответствующих разделах системного меню. Если в каком-либо из контроллеров используется порт COM1 RS232, то для него параметры связи устанавливаются в регистре D1036 и фиксируются M1138. Для COM3 RS485/RS232 параметры связи устанавливаются в регистре D1109 и фиксируются M1136. Для порта COM2 режим RTU включается реле M1143, для COM1 реле M1139. Наиболее распространенные протоколы связи: H86 (9600, 7, E, 1); H87 (9600, 8, E, 1); H96 (19200, 7, E, 1); H97 (19200, 8, E, 1); HA6 (38400, 7, E, 1); HA7 (38400, 8, E, 1). Для получения более подробной информации по процедуре настройки протокола см. описание инструкции API 80 RS.

4. Установить время ожидания ответа в D1129. Диапазон – не менее K200 и не более K3000 (3 сек). Если считывается более 16 регистров, то время должно быть не менее K500. Минимальная скорость передачи – 1200 бод. Если скорость передачи ниже 9600 бод, то время ожидания должно быть не ниже K1000 (1 сек).
5. Записать в регистр D1399 Мастера сетевой адрес первого Ведомого устройства (в десятичном формате), которому в рамках режима EASY PLC LINK присваивается идентификационный номер "Ведомый-1". Допустимый диапазон сетевых адресов от 1 до 230 в десятичном формате. Если будет определен Ведомый с номером "0", работа EASY PLC LINK будет остановлена и реле M1350 сброшено.
6. Записать в регистр D1433 Мастера количество Ведомых устройств. Адрес первого устройства берется из регистра D1399 и далее последовательно по возрастанию порядкового номера сетевого адреса. Например, если в регистр D1399 записано K20, а в регистр D1433 записать K4, то Мастер определит 4 подчиненных устройства, начиная с сетевого адреса Modbus "20", и присвоит им в рамках режима EASY PLC LINK следующие идентификационные номера: K20 – "Ведомый-1", K21 – "Ведомый-2", K22 – "Ведомый-3" и K23 – "Ведомый-4".
7. Далее в программе Мастера для каждого Ведомого определяется адресное поле под считанные из Ведомого данные и под данные, записываемые в Ведомого. Определяется длинна данных – отдельно для записи и отдельно для чтения. Если длина данных = 0, то EASY PLC LINK работать не будет.
8. Задать в программе Мастера отдельно адрес начального регистра каждого Ведомого, начиная с которого будут считываться данные, и отдельно задать адрес начального регистра каждого Ведомого, куда будут записываться данные (см. таблицы ниже).
9. В каждом Ведомом подготовить данные к пересылке, поместив их в указанные в программе Мастера регистры. Полученные данные от Мастера считать из указанных регистров и использовать далее в программе Ведомого (см. таблицы ниже).
10. Выбрать автоматический режим работы EASY PLC LINK путем включения реле M1351, или ручной режим (с заданным количеством циклов опроса) путем включения реле M1352. Одновременное включение данных реле категорически не допускается или работа EASY PLC LINK будет остановлена! В автоматическом режиме EASY PLC LINK будет выполняться до тех пор, пока активно входное условие его активации. Если выбран ручной режим, то в регистре D1431 Мастера необходимо определить количество циклов опроса, по достижению которого EASY PLC LINK отключится. Текущее значение отработанных циклов можно посмотреть в D1432. Когда содержимое D1431=D1432, работа EASY PLC LINK прекращается и реле M1352 сбрасывается. Чтобы возобновить работу EASY PLC LINK в ручном режиме снова включите реле M1352 и отсчет начнется заново в соответствии с заданным числом циклов в D1431.

11. *Обязательно сбросить реле M1354 командой RST.*
12. Выбрать режим "16 Ведомых и до 16 регистров", отключив реле M1353, или режим "32 Ведомых и до 100 регистров", включив реле M1353. Данная опция доступна только в контроллерах EH/EH2/SV и будет рассмотрена ниже.
13. Активировать работу EASY PLC LINK путем включения реле M1350. Данное реле должно включаться только после установки всех параметров работы EASY PLC LINK. В противном случае они не будут приниматься во внимание программой.
14. Для отключения работы EASY PLC LINK необходимо командой RST одновременно сбросить реле M1350 и M1351 (M1352).

Мастер определяет Ведомых только один раз при включении реле M1350, и осуществляет обмен данными только с Ведомыми, с которыми удалось установить связь в этот момент.
Мастер сначала осуществляет чтение затем запись, последовательно Ведомого за Ведомым, т.е. чтение/запись следующего Ведомого начнется только после окончания чтения/записи текущего Ведомого.
Технология EASY PLC LINK не поддерживает работу с 32-х разрядными счетчиками (C200 ~ C255).

Важное замечание:
Все специальные регистры, связанные с технологией EASY PLC LINK, являются фиксировано энергонезависимыми, т.е. их содержимое не сбрасывается ни при снятии питания с ПЛК, ни при включении специального реле общего сброса энергонезависимых регистров (не являющихся специальными). Таким образом, для обнуления специальных регистров необходимо принимать специальные меры в программе.

Описание режима "16 Ведомых и до 16 регистров"

Когда реле M1353 выключено Мастер находится в режиме "16 Ведомых и до 16 регистров". Это означает, что он может установить связь одновременно максимум с 16 Ведомыми и считывать/записывать в одном цикле максимум 16 регистров в каждом Ведомом. Сводная таблица параметров данного режима представлена ниже.

Рассмотрим смысл и установку параметров на примере Ведомого-1 (см. таблицу ниже). Для остальных Ведомых параметры настраиваются аналогично, используя соответствующие специальные регистры и реле в Мастере.

В регистрах Мастера D1480 – D1495 будут хранится данные, считанные из Ведомого-1. В регистры Мастера D1496 – D1511 записываются данные, которые нужно записать в Ведомый-1. В регистр Мастера D1434 записывается в десятичном формате количество регистров Ведомого-1, которые нужно читать (максимум 16). В регистр Мастера D1450 записывается в десятичном формате количество регистров Ведомого-1, в которые нужно записать данные (максимум 16).

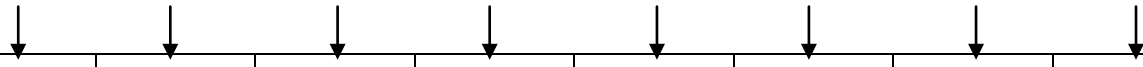
В регистр Мастера D1355 записывается адрес начального регистра Ведомого-1, откуда будут считываться данные. Мастер будет читать указанное в D1434 количество регистров Ведомого-1, начиная с адреса регистра Ведомого-1, указанного в D1355. По умолчанию стоит регистр D100 (H1064) Ведомого. Однако, лучше данный параметр указать во избежание накладок в программе. Считанные данные будут помещены в D1480 – D1495 Мастера.

В регистр Мастера D1415 записывается адрес начального регистра Ведомого-1, куда будут записываться данные. Мастер будет записывать указанное в D1450 количество регистров Ведомого-1, начиная с адреса регистра Ведомого-1, указанного в D1415. По умолчанию стоит регистр D200 (H10C8) Ведомого. Однако, лучше данный параметр указать во избежание накладок в программе. Записываемые данные нужно поместить в D1496 – D1511 Мастера.

Контролировать процесс передачи данных в Ведомый-1 можно по специальным флагам, относящимся только к нему:
M1360 – если реле включено, то связь с Ведомым-1 установлена
M1376 – идет передача данных в Ведомый-1
M1392 – ошибка чтения/записи в Ведомый-1
M1408 – чтение данных в Ведомом-1 завершено. Флаг сбрасывается, когда цикл чтение/запись Ведомого-1 закончится.
M1424 – запись данных в Ведомый-1 завершена. Флаг сбрасывается, когда цикл чтение/запись Ведомого-1 закончится.

Ведущий ПЛК (Master PLC): M1353=0															
Ведомый 1		Ведомый 2		Ведомый 3		Ведомый 4		Ведомый 5		Ведомый 6		Ведомый 7		Ведомый 8	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D1480 … D1495	D1496 … D1511	D1512 … D1527	D1528 … D1543	D1544 … D1559	D1560 … D1575	D1576 … D1591	D1592 … D1607	D1608 … D1623	D1624 … D1639	D1640 … D1655	D1656 … D1671	D1672 … D1687	D1688 … D1703	D1704 … D1719	D1720 … D1735
Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в
D1434	D1450	D1435	D1451	D1436	D1452	D1437	D1453	D1438	D1454	D1439	D1455	D1440	D1456	D1441	D1457
Коммуникационный адрес для чтения данных от ПЛК															
D1355	D1415	D1356	D1416	D1357	D1417	D1358	D1418	D1359	D1419	D1360	D1420	D1361	D1421	D1362	D1422
Обнаружение наличия ведомого ПЛК в сети															
M1360		M1361		M1362		M1363		M1364		M1365		M1366		M1367	
Флаг - идет передача данных															
M1376		M1377		M1378		M1379		M1380		M1381		M1382		M1383	
Флаг ошибки чтения / записи данных															
M1392		M1393		M1394		M1395		M1396		M1397		M1398		M1399	

Флаг – чтение завершено							
M1408	M1409	M1410	M1411	M1412	M1413	M1414	M1415
Флаг – запись завершена							
M1424	M1425	M1426	M1427	M1428	M1429	M1430	M1431



Ведомый 1		Ведомый 2		Ведомый 3		Ведомый 4		Ведомый 5		Ведомый 6		Ведомый 7		Ведомый 8	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215

Заводская уставка адреса начального регистра для чтения данных – H1064 (D100).
Заводская уставка адреса начального регистра для записи данных – H10C8 (D200).

Ведущий ПЛК (Master PLC) : M1353=0															
Ведомый 9		Ведомый 10		Ведомый 11		Ведомый 12		Ведомый 13		Ведомый 14		Ведомый 15		Ведомый 16	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D1736 … D1751	D1752 … D1767	D1768 … D1783	D1784 … D1799	D1800 … D1815	D1816 … D1831	D1832 … D1847	D1848 … D1863	D1864 … D1879	D1880 … D1895	D1896 … D1911	D1912 … D1927	D1928 … D1943	D1944 … D1959	D1960 … D1975	D1976 … D1991
Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в
D1442	D1458	D1443	D1459	D1444	D1460	D1445	D1461	D1446	D1462	D1447	D1463	D1448	D1464	D1449	D1465
Коммуникационный адрес для чтения данных от ПЛК															
D1363	D1423	D1364	D1424	D1365	D1425	D1366	D1426	D1367	D1427	D1368	D1428	D1369	D1429	D1370	D1430
Обнаружение наличия ведомого ПЛК в сети															
M1368		M1369		M1370		M1371		M1372		M1373		M1374		M1375	
Флаг - идет передача данных															

M1384	M1385	M1386	M1387	M1388	M1389	M1390	M1391
Флаг ошибки чтения / записи данных							
M1400	M1401	M1402	M1403	M1404	M1405	M1406	M1407
Флаг – чтение завершено							
M1416	M1417	M1418	M1419	M1420	M1421	M1422	M1423
Флаг – запись завершена							
M1432	M1433	M1434	M1435	M1436	M1437	M1438	M1439

Ведомый 9		Ведомый 10		Ведомый 11		Ведомый 12		Ведомый 13		Ведомый 14		Ведомый 15		Ведомый 16	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D100	D200	D100	D200	D100	D200	D100	D200	D100	D200	D100	D200	D100	D200	D100	D200
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
D115	D215	D115	D215	D115	D215	D115	D215	D115	D215	D115	D215	D115	D215	D115	D215

Заводская уставка адреса начального регистра для чтения данных – H1064 (D100).
Заводская уставка адреса начального регистра для записи данных – H10C8 (D200).

Описание режима "32 Ведомых и до 100 регистров"

Данный режим возможен только для контроллеров типов EH/EH2/SV.
Когда реле M1353 включено Мастер находится в режиме "32 Ведомых и до 100 регистров". Это означает, что он может установить связь одновременно максимум с 32 Ведомыми и считывать/записывать в одном цикле максимум 100 регистров в каждом Ведомом. Сводная таблица параметров данного режима представлена ниже. Установка параметров для каждого Ведомого несколько отличается от режима "16 Ведомых и до 16 регистров" и регистры имеют другое назначение:

- В ячейки D1480 ~ D1495 записываются адреса начальных регистров Мастера, куда будут записываться данные, прочитанные с Ведомых 1 ~ 16 в текущем цикле.
- В ячейки D1496 ~ D1511 записываются адреса начальных регистров Мастера, данные откуда будут записаны в Ведомые 1 ~ 16 в текущем цикле.
- В ячейки D1512 ~ D1527 записываются адреса начальных регистров Ведомых 17 ~ 32, откуда Мастер будет читать данные.
- В ячейки D1528 ~ D1543 записываются адреса начальных регистров Ведомых 17 ~ 32, куда Мастер будет записывать данные.
- В ячейки D1544 ~ D1559 записывается количество регистров в Ведомых 17 ~ 32, которое будет считывать Мастер (максимум 100).
- В ячейки D1560 ~ D1575 записывается количество регистров в Ведомых 17 ~ 32, в которые Мастер будет записывать данные (максимум 100).
- В ячейки D1576 ~ D1591 записываются адреса начальных регистров Мастера для хранения данных, принятых от Ведомых 17 ~ 32.

- В ячейки D1592 ~ D1607 записываются адреса адреса начальных регистров Мастера, откуда данные будут записаны в Ведомые 17 ~ 32.

Например, для Ведомого-1 в данном случае в регистр D1480 Мастера записывается адрес начального регистра Мастера, куда будут сохранены данные, считанные из Ведомого-1.
В регистр D1496 Мастера записывается адрес начального регистра Мастера, откуда будут браться данные для записи в Ведомый-1. Диапазон адресов регистров 1 ~ 9900.
В регистр D1434 записывается количество регистров в Ведомом-1, которое будет считано Мастером (максимум 100), а в регистр D1450 записывается количество регистров, которое будет записано Мастером в Ведомый-1 (максимум 100).
В регистр D1355 записывается адрес начального регистра Ведомого-1, начиная с которого Мастер будет считывать данные в Ведомом-1, а в регистр D1415 записывается адрес начального регистра Ведомого-1, начиная с которого Мастер будет будет записывать данные в Ведомый-1.
Количество регистров чтения/записи можно менять не отключая EASY PLC LINK, но новые уставки в ступят в действие со следующего цикла опроса.

Ведущий ПЛК (Master PLC): M1353=1															
Ведомый 1		Ведомый 2		Ведомый 3		Ведомый 4		Ведомый 5		Ведомый 6		Ведомый 7		Ведомый 8	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D1480	D1496	D1481	D1497	D1482	D1498	D1483	D1499	D1484	D1500	D1485	D1501	D1486	D1502	D1487	D1503
Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в
D1434	D1450	D1435	D1451	D1436	D1452	D1437	D1453	D1438	D1454	D1439	D1455	D1440	D1456	D1441	D1457
Коммуникационный адрес для чтения данных от ПЛК															
D1355	D1415	D1356	D1416	D1357	D1417	D1358	D1418	D1359	D1419	D1360	D1420	D1361	D1421	D1362	D1422
Обнаружение наличия ведомого ПЛК в сети															
M1360		M1361		M1362		M1363		M1364		M1365		M1366		M1367	
Флаг - идет передача данных															
M1376		M1377		M1378		M1379		M1380		M1381		M1382		M1383	
Флаг ошибки чтения / записи данных															
M1392		M1393		M1394		M1395		M1396		M1397		M1398		M1399	
Флаг – чтение завершено															
M1408		M1409		M1410		M1411		M1412		M1413		M1414		M1415	

Флаг – запись завершена							
M1424	M1425	M1426	M1427	M1428	M1429	M1430	M1431

Ведомый 1		Ведомый 2		Ведомый 3		Ведомый 4		Ведомый 5		Ведомый 6		Ведомый 7		Ведомый 8	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215

Заводская уставка адреса начального регистра для чтения данных – H1064 (D100).
Заводская уставка адреса начального регистра для записи данных – H10C8 (D200).

Ведущий ПЛК (Master PLC) : M1353=1															
Ведомый 9		Ведомый 10		Ведомый 11		Ведомый 12		Ведомый 13		Ведомый 14		Ведомый 15		Ведомый 16	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D1488	D1504	D1489	D1505	D1490	D1506	D1491	D1507	D1492	D1508	D1493	D1509	D1494	D1510	D1495	D1511
Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в
D1442	D1458	D1443	D1459	D1444	D1460	D1445	D1461	D1446	D1462	D1447	D1463	D1448	D1464	D1449	D1465
Коммуникационный адрес для чтения данных от ПЛК															
D1363	D1423	D1364	D1424	D1365	D1425	D1366	D1426	D1367	D1427	D1368	D1428	D1369	D1429	D1370	D1430
Обнаружение наличия ведомого ПЛК в сети															
M1368		M1369		M1370		M1371		M1372		M1373		M1374		M1375	
Флаг - идет передача данных															
M1384		M1385		M1386		M1387		M1388		M1389		M1390		M1391	
Флаг ошибки чтения / записи данных															
M1400		M1401		M1402		M1403		M1404		M1405		M1406		M1407	
Флаг – чтение завершено															

M1416	M1417	M1418	M1419	M1420	M1421	M1422	M1423
Флаг – запись завершена							
M1432	M1433	M1434	M1435	M1436	M1437	M1438	M1439

Ведомый 9		Ведомый 10		Ведомый 11		Ведомый 12		Ведомый 13		Ведомый 14		Ведомый 15		Ведомый 16	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215

Заводская уставка адреса начального регистра для чтения данных – H1064 (D100).
Заводская уставка адреса начального регистра для записи данных – H10C8 (D200).

Ведущий ПЛК (Master PLC) : M1353=1															
Ведомый 17		Ведомый 18		Ведомый 19		Ведомый 20		Ведомый 21		Ведомый 22		Ведомый 23		Ведомый 24	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D1576	D1592	D1577	D1593	D1578	D1594	D1579	D1595	D1580	D1596	D1581	D1597	D1582	D1598	D1583	D1599
Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в
D1544	D1560	D1545	D1561	D1546	D1562	D1547	D1563	D1548	D1564	D1549	D1565	D1550	D1566	D1551	D1567
Коммуникационный адрес для чтения данных от ПЛК															
D1512	D1528	D1513	D1529	D1514	D1530	D1515	D1531	D1516	D1532	D1517	D1533	D1518	D1534	D1519	D1535
Обнаружение наличия ведомого ПЛК в сети															
M1440		M1441		M1442		M1443		M1444		M1445		M1446		M1447	
Флаг - идет передача данных															
M1456		M1457		M1458		M1459		M1460		M1461		M1462		M1463	

Флаг ошибки чтения / записи данных							
M1472	M1473	M1474	M1475	M1476	M1477	M1478	M1479
Флаг – чтение завершено							
M1488	M1489	M1490	M1491	M1492	M1493	M1494	M1495
Флаг – запись завершена							
M1504	M1505	M1506	M1507	M1508	M1509	M1510	M1511

Ведомый 17		Ведомый 18		Ведомый 19		Ведомый 20		Ведомый 21		Ведомый 22		Ведомый 23		Ведомый 24	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215	D100 … D115	D200 … D215

Заводская уставка адреса начального регистра для чтения данных – H1064 (D100).
Заводская уставка адреса начального регистра для записи данных – H10C8 (D200).

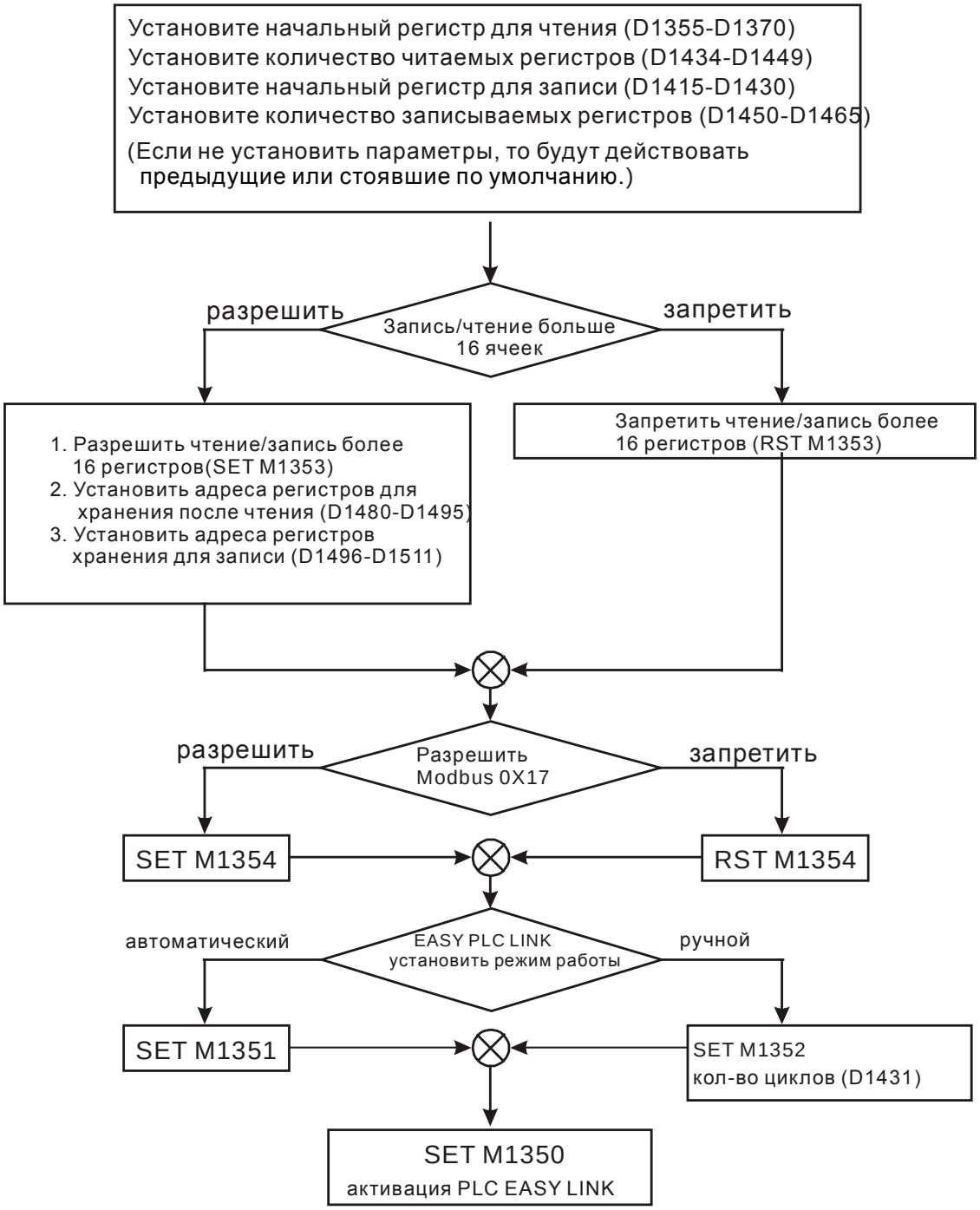
Ведущий ПЛК (Master PLC) : M1353=1															
Ведомый 25		Ведомый 26		Ведомый 27		Ведомый 28		Ведомый 29		Ведомый 30		Ведомый 31		Ведомый 32	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D1584	D1600	D1585	D1601	D1586	D1602	D1587	D1603	D1588	D1604	D1589	D1605	D1590	D1606	D1591	D1607
Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в	Число рег-в
D1552	D1568	D1553	D1569	D1554	D1570	D1555	D1571	D1556	D1572	D1557	D1573	D1558	D1574	D1559	D1575
Коммуникационный адрес для чтения данных от ПЛК															
D1520	D1536	D1521	D1537	D1522	D1538	D1523	D1539	D1524	D1540	D1525	D1541	D1526	D1542	D1527	D1543
Обнаружение наличия ведомого ПЛК в сети															
M1448		M1449		M1450		M1451		M1452		M1453		M1454		M1455	

Флаг - идет передача данных							
M1464	M1465	M1466	M1467	M1468	M1469	M1470	M1471
Флаг ошибки чтения / записи данных							
M1480	M1481	M1482	M1483	M1484	M1485	M1486	M1487
Флаг – чтение завершено							
M1496	M1497	M1498	M1499	M1500	M1501	M1502	M1503
Флаг – запись завершена							
M1512	M1513	M1514	M1515	M1516	M1517	M1518	M1519

Ведомый 25		Ведомый 26		Ведомый 27		Ведомый 28		Ведомый 29		Ведомый 30		Ведомый 31		Ведомый 32	
Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в	Чтение из	Запись в
D100	D200	D100	D200	D100	D200	D100	D200	D100	D200	D100	D200	D100	D200	D100	D200
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
D115	D215	D115	D215	D115	D215	D115	D215	D115	D215	D115	D215	D115	D215	D115	D215

Заводская уставка адреса начального регистра для чтения данных – H1064 (D100).
Заводская уставка адреса начального регистра для записи данных – H10C8 (D200).

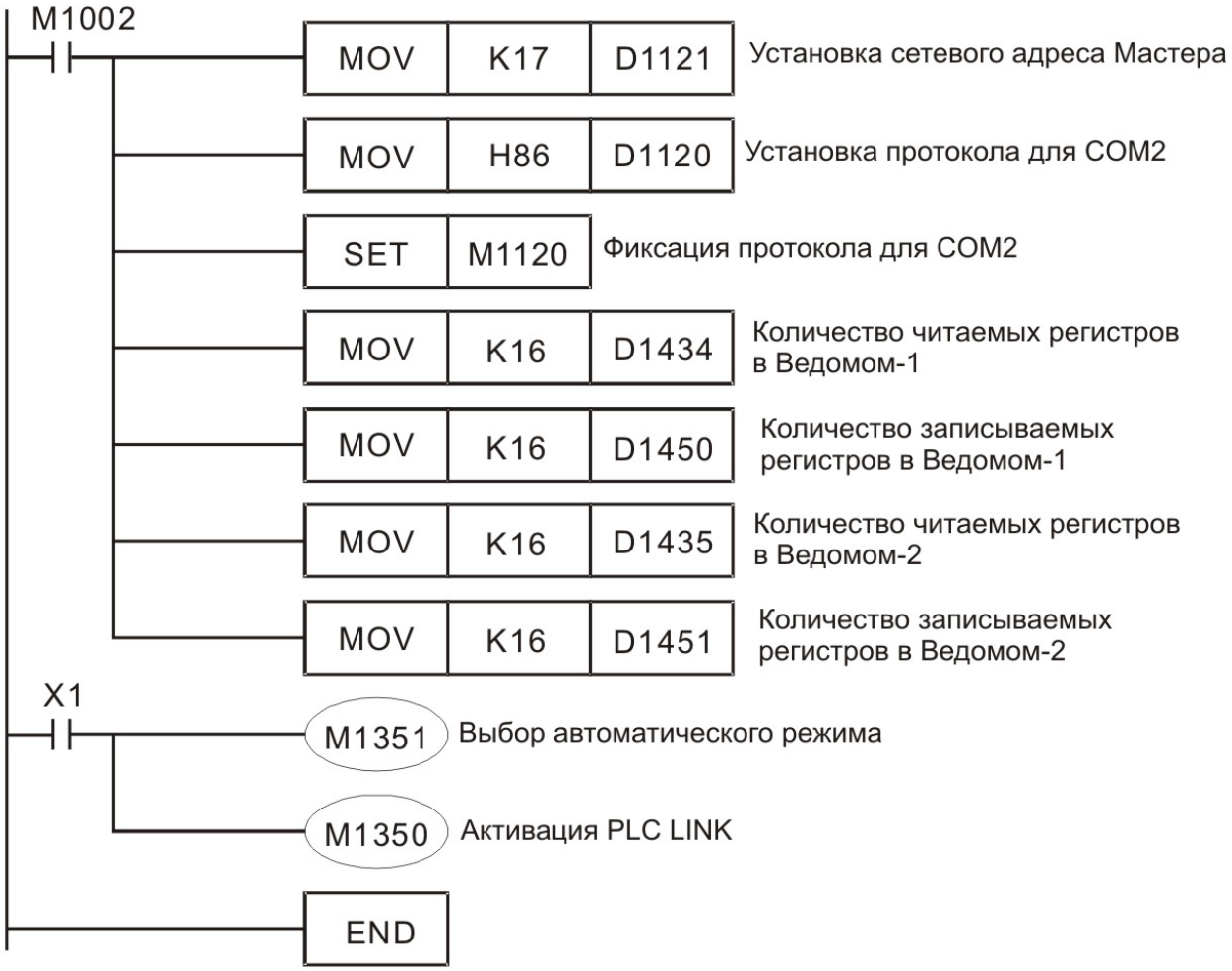
Ниже приведен краткий алгоритм основных шагов процесса настройки работы EASY PLC LINK:



Примеры реализации коммуникационной технологии EASY PLC LINK

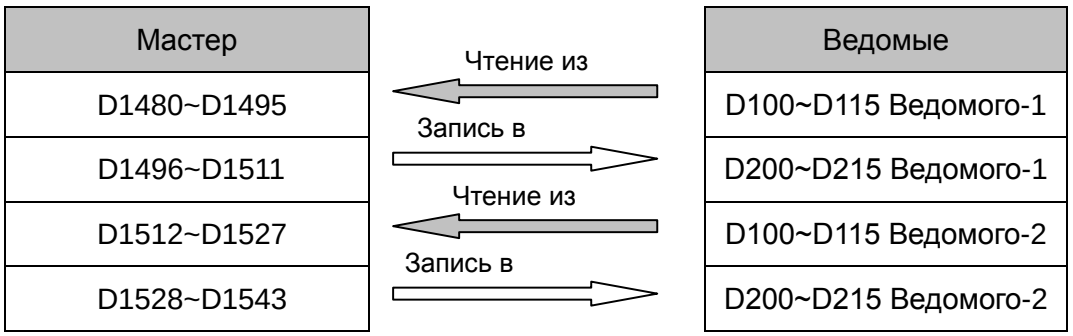
Пример 1.
Фрагмент программы обмена данными Мастера ПЛК с двумя Ведомыми ПЛК через порт COM2 RS485 по технологии EASY PLC LINK.

Значение регистра D1433 нужно записать K2 (два Ведомых), в регистр D1399 записывается сетевой адрес Ведомого-1, реле M1353 выключено (режим 16 и 16).
Далее см. комментарии на фрагменте программы.



Когда замыкается X1 устанавливается связь между Мастером и двумя Ведомыми по технологии EASY PLC LINK. Данные из регистров D100 ~ D115 обоих Ведомых будут считаны и сохранены в регистры Мастера D1480 ~ D1495 (из Ведомого-1) и D1512 ~ D1527 (из Ведомого-2). Данные из регистров Мастера D1496 ~ D1511 будут записаны в регистры D200 ~ D215 Ведомого-1, а из D1528 ~ D1543 в регистры D200 ~ D215 Ведомого-2.

Данный процесс проиллюстрирован ниже.



Мастер сначала читает/записывает Ведомого-1 и только после полного окончания опроса переходит к Ведомому-2.

Для наглядности ниже приведена таблица с конкретными данными в регистрах до активации EASY PLC LINK и после начала обмена данными.

M1350=0 (выключено)

Мастер	Значение регистров	Ведомые	Значение регистров
D1480~D1495	K0 для всех	D100~D115 Ведомый-1	K5000 для всех
D1496~D1511	K1000 для всех	D200~D215 Ведомый-1	K0 для всех
D1512~D1527	K0 для всех	D100~D115 Ведомый-2	K6000 для всех
D1528~D1543	K2000 для всех	D200~D215 Ведомый-2	K0 для всех

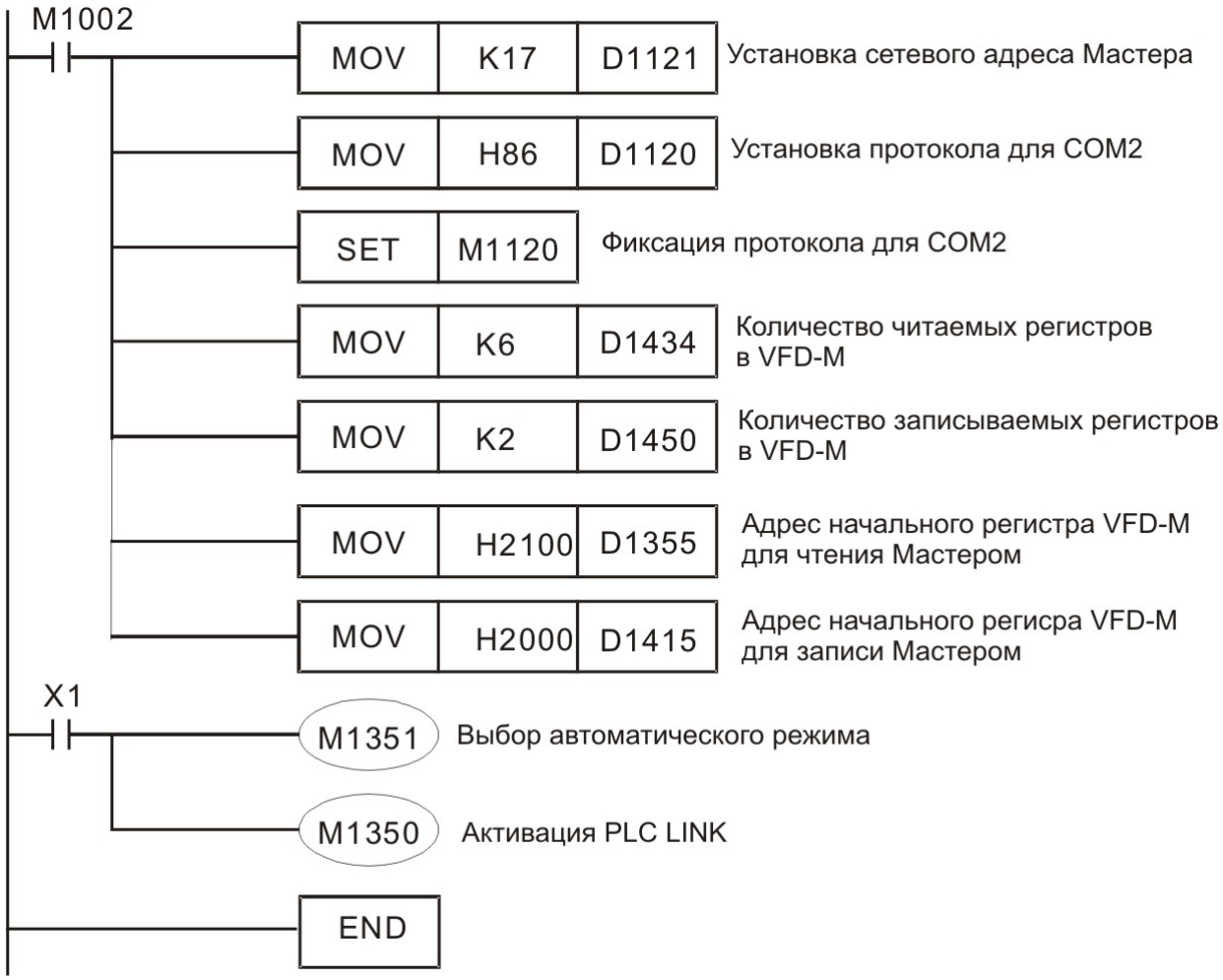
M1350=1 (включено)

Мастер	Значение регистров	Ведомые	Значение регистров
D1480~D1495	K5000 для всех	D100~D115 Ведомый-1	K5000 для всех
D1496~D1511	K1000 для всех	D200~D215 Ведомый-1	K1000 для всех
D1512~D1527	K6000 для всех	D100~D115 Ведомый-2	K6000 для всех
D1528~D1543	K2000 для всех	D200~D215 Ведомый-2	K2000 для всех

Пример 2.

Фрагмент программы обмена данными Мастер ПЛК с одним Ведомым частотным преобразователем VFD-M через порт COM2 RS485 по технологии EASY PLC LINK.

Значение регистра D1433 нужно записать K1 (один Ведомый), в регистр D1399 записывается сетевой адрес Ведомого-1 (VFD-M), реле M1353 выключено (режим 16 и 16).
Далее см. комментарии на фрагменте программы.



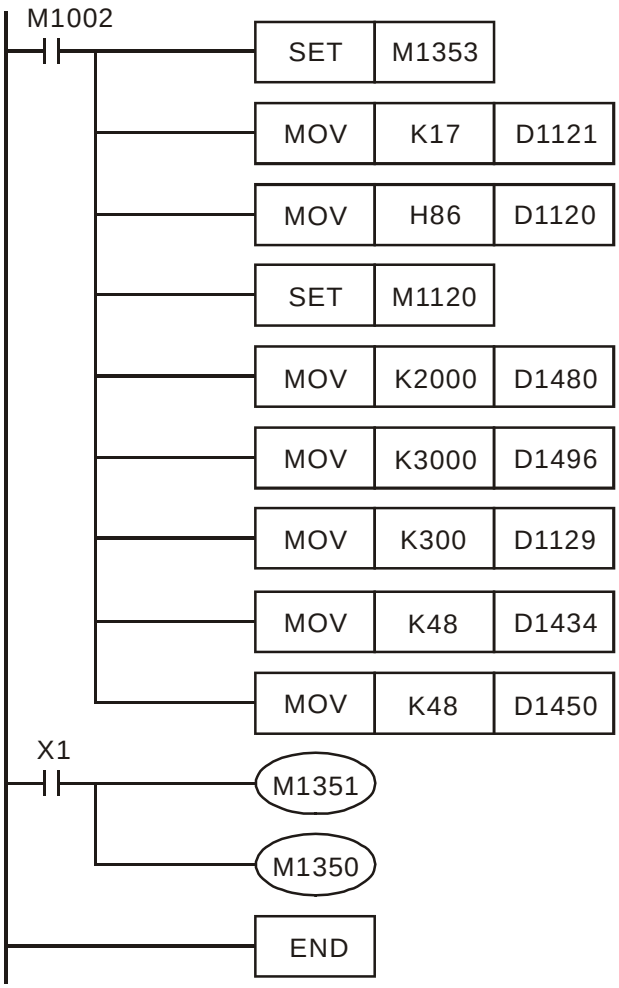
Когда замыкается X1 устанавливается связь между Мастером ПЛК и Ведомым VFD-M по технологии EASY PLC LINK. Данные из регистров H2100 ~ H2105 VFD-M будут считаны и сохранены в регистры Мастера D1480 ~ D1485. Данные из регистров Мастера D1496 ~ D1497 будут записаны в регистры H2000 ~ H2001 VFD-M.
Процесс будет идти цикл за циклом пока включены реле M1351 и M1350.

Пример 3.

Фрагмент программы обмена данными Мастер ПЛК с одним Ведомым через порт COM2 RS485 по технологии EASY PLC LINK в режиме "32 Ведомых и 100 регистров". Доступен для контроллеров типов EH/EH2/SV.

Значение регистра D1433 нужно записать K1 (один Ведомый), в регистр D1399 записывается сетевой адрес Ведомого-1, реле M1353 включено (режим 32 и 100).
В ячейку Мастера D1480 записывается адрес начального регистра Мастера куда будут помещаться считанные и Ведомого-1 данные (диапазон 1 ~ 9900).

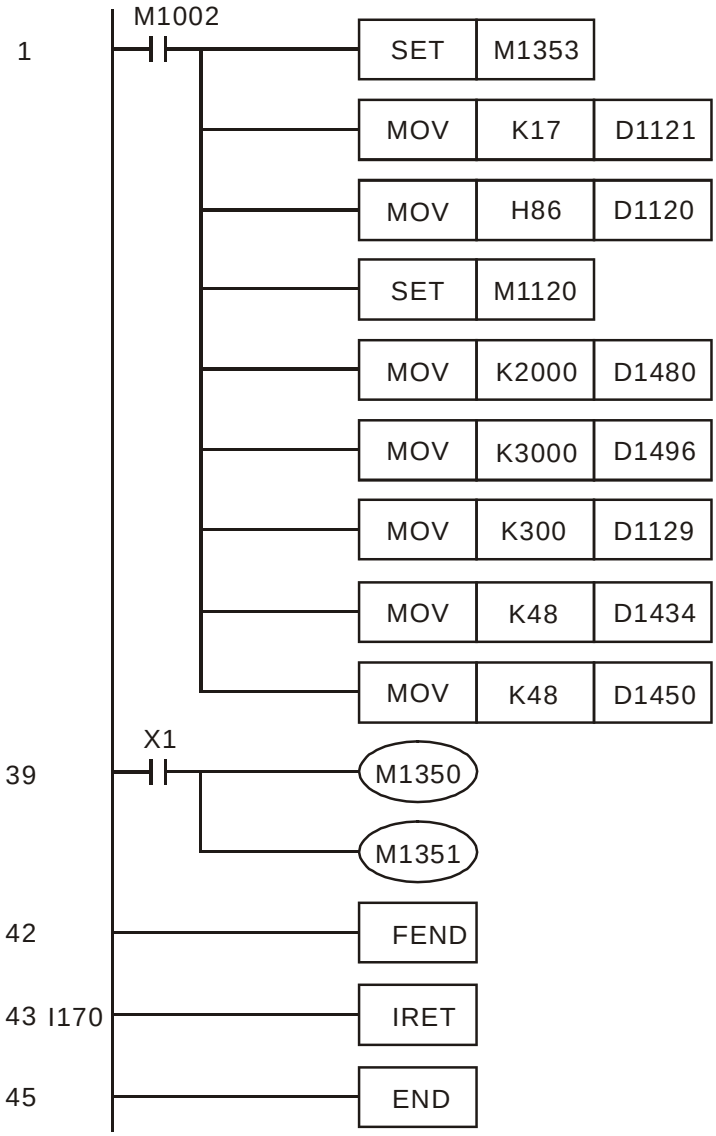
В ячейку Мастера D1496 записывается адрес начального регистра Мастера откуда будут браться данные для записи в Ведомого-1 (диапазон 1 ~ 9900). В ячейку D1434 записывается количество регистров для считывания из Ведомого (в примере 48). В ячейку D1450 записывается количество регистров для записи в Ведомого (в примере 48). Реле M1351 включает автоматический режим опроса, а реле M1350 активирует EASY PLC LINK.



Пример 4.

Фрагмент программы обмена данными Мастер ПЛК с одним Ведомым через порт COM2 RS485 по технологии EASY PLC LINK в режиме "32 Ведомых и 100 регистров" с использованием прерывания I170. Доступен для контроллеров типов EH/EH2/SV.

Данный пример аналогичен предыдущему, отличие заключается в том, что данные будут обрабатываться не по достижению команды END, а сразу после включения указателя I170. Необходимо иметь ввиду, что если порт Ведомого работает медленно (особенно сигнал определения направления), то нет смысла использовать прерывание в программе.



2.12 Адресация операндов контроллеров Delta DVP

Операнд	Диапазон	Тип	Адрес	Тип ПЛК		
				ES/EX/SS	SA/SX/SC	EH/EH2/SV
S	000~255	бит	0000~00FF	0~127	0~1024	0~1024
S	246~511	бит	0100~01FF	-		
S	512~767	бит	0200~02FF			
S	768~1023	бит	0300~03FF			
X	000~377 (Octal)	бит	0400~04FF	0~177	0~177	0~377
Y	000~377 (Octal)	бит	0500~05FF	0~177	0~177	0~377
T	000~255	бит/слово	0600~06FF	0~127	0~255	0~255
M	000~255	бит	0800~08FF	0~1279	0~4095	0~4095
M	256~511	бит	0900~09FF			
M	512~767	бит	0A00~0AFF			
M	768~1023	бит	0B00~0BFF			
M	1024~1279	бит	0C00~0CFF			
M	1280~1535	бит	0D00~0DFF	-		

M	1536~1791		бит	B000~B0FF			
M	1792~2047		бит	B100~B1FF			
M	2048~2303		бит	B200~B2FF			
M	2304~2559		бит	B300~B3FF			
M	2560~2815		бит	B400~B4FF			
M	2816~3071		бит	B500~B5FF			
M	3072~3327		бит	B600~B6FF			
M	3328~3583		бит	B700~B7FF			
M	3584~3839		бит	B800~B8FF			
M	3840~4095		бит	B900~B9FF			
C	0~199	16-бит	бит/слово	0E00~0EC7	0~127	0~199	0~199
	200~255	32-бит	бит/2слова	0EC8~0EFF	232~255	200~255	200~255
D	000~256		слово	1000~10FF	0~1311	0~4999	0~9999
D	256~511		слово	1100~11FF			
D	512~767		слово	1200~12FF			
D	768~1023		слово	1300~13FF			
D	1024~1279		слово	1400~14FF			
D	1280~1535		слово	1500~15FF			
D	1536~1791		слово	1600~16FF	-	0~4999	0~9999
D	1792~2047		слово	1700~17FF		-	
D	2048~2303		слово	1800~18FF			
D	2304~2559		слово	1900~19FF			
D	2560~2815		слово	1A00~1AFF			
D	2816~3071		слово	1B00~1BFF			
D	3072~3327		слово	1C00~1CFF			
D	3328~3583		слово	1D00~1DFF			
D	3584~3839		слово	1E00~1EFF			
D	3840~4095		слово	1F00~1FFF			
D	4096~4351		слово	9000~90FF			
D	4352~4607		слово	9100~91FF			
D	4608~4863		слово	9200~92FF			
D	4864~5119		слово	9300~93FF			
D	5120~5375		слово	9400~94FF			
D	5376~5631		слово	9500~95FF			
D	5632~5887		слово	9600~96FF			
D	5888~6143		слово	9700~97FF			
D	6144~6399		слово	9800~98FF			
D	6400~6655		слово	9900~99FF			
D	6656~6911		слово	9A00~9AFF			
D	6912~7167		слово	9B00~9BFF			
D	7168~7423		слово	9C00~9CFF			
D	7424~7679		слово	9D00~9DFF			
D	7680~7935		слово	9E00~9EFF			
D	7936~8191		слово	9F00~9FFF			
D	8192~8447		слово	A000~A0FF			
D	8448~8703		слово	A100~A1FF			
D	8704~8959		слово	A200~A2FF			
D	8960~9215		слово	A300~A3FF			
D	9216~9471		слово	A400~A4FF			
D	9472~9727		слово	A500~A5FF			
D	9728~9983		слово	A600~A6FF			
D	9984~9999		слово	A700~A70F			

2.13 Коды ошибок

Если после загрузки программы в контроллер начал мигать индикатор ERROR и включилось реле M1004, то это значит, что программа содержит неправильно используемый операнд, синтаксическую ошибку, или какой-либо из операндов вышел за допустимый диапазон.
В данном случае по нижеприведенной таблице можно узнать причину ошибки. ПЛК хранит номер шага в D1137, а код ошибки в D1004. Если причиной является общая ошибка цикла программы, то адрес в D1137 будет недействителен.

Код ошибки	Описание
0001	Операнд битового устройства S выходит за границы диапазона.
0002	Метка P выходит за границы диапазона или дублируется.
0003	Операнд KnSm выходит за границы диапазона.
0102	Флаг прерывания I выходит за границы диапазона или дублируется.
0202	Инструкция MC выходит за границы диапазона.
0302	Инструкция MCR выходит за границы диапазона.
0401	Операнд битового устройства X выходит за границы диапазона.
0403	Операнд битового устройства KnXm выходит за границы диапазона.
0501	Операнд битового устройства Y выходит за границы диапазона.
0503	Операнд битового устройства KnYm выходит за границы диапазона.
0601	Операнд битового устройства T выходит за границы диапазона.
0604	Операнд регистра T выходит за границы диапазона.
0801	Операнд битового устройства M выходит за границы диапазона.
0803	Операнд битового устройства KnMm выходит за границы диапазона.
0D01	Неправильный операнд в инструкции DECO.
0D02	ES/EX/SS/EH: Неправильный операнд в инструкции ENCO. SA/SX/SC: Некоректное использование первого операнда в инструкции ANS.
0D03	Неправильный операнд в инструкции DHSCS.
0D04	Неправильный операнд в инструкции DHSCR.
0D05	Неправильный операнд в инструкции импульсного выхода.
0D06	Неправильный операнд в инструкции PWM.
0D07	Неправильное использование операндов в инструкциях FROM/TO.
0D08	Неправильные операнды в инструкции PID.
0D09	Неправильный операнд в инструкции SPD.
0D0A	Неправильный операнд в инструкции DHSZ.
0D0B	Неправильный операнд в инструкции IST.
0E01	Операнд битового устройства C выходит за границы диапазона.
0E04	Операнд регистра C выходит за границы диапазона.
0E05	Неправильный операнд Cxxx в инструкции DCNT.
0E18	Ошибка преобразования BCD.
0E19	Деление на ноль (делитель = 0).
0E1A	Операнд вышел за допустимый диапазон (включая индексы E и F)
0E1B	Индекс корня является отрицательным числом.
0E1C	Ошибка связи инструкций FROM/TO.
0F04	Регистр D выходит за границы диапазона.
0F05	Неправильный операнд D в инструкции DCNT.
0F06	Неправильный операнд в инструкции SFTR.
0F07	Неправильный операнд в инструкции SFTL.

0F08	Неправильный операнд в инструкции REF.
0F09	Неправильный операнд в инструкциях WSFR, WSFL.
0F0A	Количество раз использования инструкций TTMR, STMR превышает допустимое значение.
0F0B	Количество раз использования инструкции SORT превышает допустимое значение.
0F0C	Количество раз использования инструкции TKY превышает допустимое значение.
0F0D	Количество раз использования инструкции HKY превышает допустимое значение.
1000	Неправильный операнд в инструкции ZRST.
10EF	Неправильное использование индексов E и F, или индексированное значение выходит за допустимый диапазон.
2000	Количество раз использования инструкций TTMR, PR, HOUR превышает допустимое значение. Неправильное использование операндов в инструкциях MRT, ARWS.
C400	В программе содержится не распознанная инструкция.
C401	Общая ошибка цикла программы.
C402	Инструкция LD/LDI непрерывно используется более 9 раз.
C403	Инструкция MPS непрерывно используется более 9 раз.
C404	FOR-NEXT превышает 6 уровней вложения.
C405	Инструкция STL/RET находится в цикле FOR/NEXT. Инструкция SRET/IRET находится в цикле FOR/NEXT. Инструкция MC/MCR находится в цикле FOR/NEXT. Инструкция END/FEND находится в цикле FOR/NEXT.
C407	Инструкция STL непрерывно используется более 9 раз.
C408	Использование MC/MCR в STL или указателей I/P в STL.
C409	Использование STL/RET в подпрограмме или обработке прерывания.
C40A	Использование MC/MCR в подпрограмме или обработке прерывания
C40B	MC/MCR не начинается с N0 или прерывается.
C40C	Инструкциям MC и MCR соответствует разные значения N.
C40D	Некорректное использование указателей P/I.
C40E	IRET не должно стоять после последней команды FEND. SRET не должно стоять после последней команды FEND.
C40F	Программа ПЛК и данные в параметрах не инициализированы.
C41B	Недействительная инструкция RUN/STOP для модуля расширения.
C41C	Количество точек ввода/вывода модулей расширения превышает максимальное число.
C41D	Количество модулей расширения превышает допустимое количество.
C41E	Неправильная установка параметров аппаратной части для модуля расширения.
C41F	Не удалось записать данные в память.
C420	Ошибка чтения/записи функциональной карты.
C430	Ошибка инициализации параллельного интерфейса (внутренняя шина ПЛК - модули).
C440	Ошибка аппаратной части высокоскоростного счетчика.
C441	Ошибка аппаратной части высокоскоростного компаратора.
C442	Ошибка аппаратной части импульсного выхода.
C443	Нет ответа от модуля расширения.
C4EE	В программе нет инструкции END.
C4FF	Инструкция с неизвестным именем (отсутствует в списке инструкций).

ГЛАВА 3

Базовый набор команд и инструкций контроллеров Delta DVP

3.1 Перечень базовых команд контроллеров Delta DVP

В таблицах ниже перечислены базовые команды контроллеров. Колонки с заголовком "ES" включают также контроллеры типов EX/SS, колонки с заголовком "SA" включают также SX/SC, а колонки "EH/SV" включают также EH2.
Для контроллеров EH/EH/SV в скобках указано время исполнения при задействовании реле M1536~M4095. Время исполнения команд для других операндов указано без скобок.

Логические команды

Код команды	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
			ES	SA	EH/SV		
LD	Нормально-открытый контакт	X, Y, M, S, T, C	3.8	3.8	0.24 (0.56)	1~3	
LDI	Нормально-закрытый контакт	X, Y, M, S, T, C	3.88	3.88	0.24 (0.56)	1~3	
AND	Последовательный нормально-открытый контакт (логическое И)	X, Y, M, S, T, C	2.32	2.32	0.24 (0.56)	1~3	
ANI	Последовательный нормально-закрытый контакт (И-НЕ)	X, Y, M, S, T, C	2.4	2.4	0.24 (0.56)	1~3	
OR	Параллельный нормально-открытый контакт (логическое ИЛИ)	X, Y, M, S, T, C	2.32	2.32	0.24 (0.56)	1~3	
ORI	Параллельный нормально-закрытый контакт (ИЛИ-НЕ)	X, Y, M, S, T, C	2.4	2.4	0.24 (0.56)	1~3	
ANB	«И» блок: последовательное включение блоков контактов	нет	1.76	1.76	0.24	1~3	
ORB	«ИЛИ» блок: параллельное включение блоков контактов	нет	1.76	1.76	0.24	1~3	
MPS	Точка начала разветвления с одним входным условием для всего разветвления	нет	1.68	1.68	0.24	1~3	
MRD	Промежуточная точка разветвления с одним входным условием для всего разветвления (ответвление)	нет	1.6	1.6	0.24	1	
MPP	Точка конца разветвления с одним входным условием для всего разветвления	нет	1.6	1.6	0.24	1	

🕮 Команды установки состояния выходов

Код команды	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
			ES	SA	EH/SV		
OUT	Присвоение выходу результата предыдущего логического выражения	Y, M, S	5.04	5.04	0.24 (0.56)	1~3	
SET	Фиксированное включение операнда (установка логической "1")	Y, M, S	3.8	3.8	0.24 (0.56)	1~3	
RST	Сброс контактов в исходное, очистка регистров от содержимого	Y, M, S, T, C, D, E, F	7.8	7.8	0.24 (0.56)	3	

🕮 Таймеры, счетчики

API	Код инструкции	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
				ES	SA	EH/SV		
96	TMR	Таймер (16 бит)	T-K или T-D	10.6	10.6	9.6	4	
97	CNT	Счетчик (16 бит)	C-K или C-D (16 бит)	9.7	9.7	12.8	4	
97	DCNT	Счетчик (32 бит)	C-K или C-D (32 бит)	10.3	10.3	14.3	6	

🕮 Команды исключения участков программы

Код команды	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
			ES	SA	EH/SV		
MC	Начало исключаемого участка программы	N0 ~ N7	5.6	5.6	5.6	3	
MCR	Конец исключаемого участка программы	N0 ~ N7	5.7	5.7	5.7	3	

🕮 Команды опроса входов по переднему или заднему фронту импульса

API	Код инструкции	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
				ES	SA	EH/SV		
90	LDP	Формирование импульса по переднему фронту входного сигнала.	X, Y, M, S, T, C	5.1	5.1	0.56 (0.88)	3	
91	LDF	Формирование импульса по заднему фронту входного сигнала	X, Y, M, S, T, C	5.1	5.1	0.56 (0.88)	3	
92	ANDP	Последовательное соединение контакта, формирующего импульс по переднему фронту сигнала	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	
93	ANDF	Последовательное соединение контакта, формирующего импульс по заднему фронту сигнала	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	

94	ORP	Параллельное соединение контакта, формирующего импульс по переднему фронту сигнала	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	
95	ORF	Параллельное соединение контакта, формирующего импульс по заднему фронту сигнала	X, Y, M, S, T, C	4.9	4.9	0.56 (0.88)	3	

🕮 Выдача импульса выходом по переднему/заднему фронту входного сигнала

API	Код инструкции	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
				ES	SA	EH/SV		
89	PLS	Выдача импульса выходом при появлении переднего фронта сигнала на входе	Y, M	7.8	7.8	9.92	3	
99	PLF	Выдача импульса выходом при появлении заднего фронта сигнала на входе	Y, M	7.8	7.8	10.16	3	

🕮 Обозначение конца программы

Код команды	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
			ES	SA	EH/SV		
END	Конец программы	нет	5	5	0.24	1	

🕮 Прочие инструкции

API	Код инструкции	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
				ES	SA	EH/SV		
	NOP	Пустая строка	нет	0.88	0.88	0.16	1	
98	INV	Инверсия: замена результата логических связей на противоположный	нет	1.6	1.6	0.24	1	
	P	Указатель	P0 ~ P255	0.88	0.88	-	1	
	I	Указатель прерывания	I□□□	0.88	0.88	-	1	

🕮 Команды пошагового управления

Код команды	Функция	Операнды	Скорость выполнения (мкс)			К-во шагов	Стр.
			ES	SA	EH/SV		
STL	Начало пошагового управления	S	11.6	10.6	0.56	1	
RET	Конец пошагового управления	нет	7.04	6.04	0.24	1	

3.2 Описание базовых команд контроллеров Delta DVP

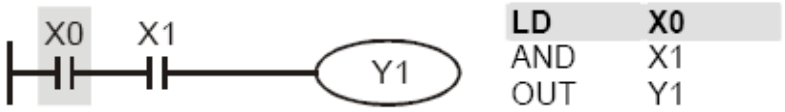
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
LD	Нормально-открытый контакт	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда LD используется в качестве нормально-открытого контакта для программирования начала логических цепочек. В контактных схемах команда всегда расположена слева и соединяется непосредственно с шиной питания.

Применение:



Команда "нормально-открытый контакт X0" открывает последовательную логическую связь. Если на входах X0 и X1 одновременно будет сигнал "1", тогда и выход Y1 установится в состояние "1".

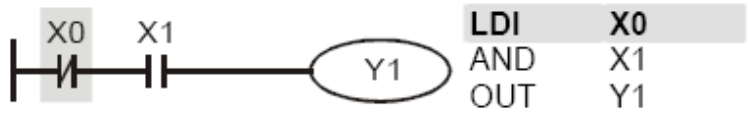
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
LDI	Нормально-закрытый контакт	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда LD используется в качестве нормально-закрытого контакта для программирования начала логических цепочек. В контактных схемах команда всегда расположена слева и соединяется непосредственно с шиной питания.

Применение:



Команда "нормально-закрытый контакт X0" открывает последовательную логическую связь. Если на входе X0 будет "0", а на X1 будет сигнал "1", тогда выход Y1 установится в состояние "1".

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
AND	Последовательный нормально-открытый контакт (логическое И)	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда AND используется в качестве последовательного нормально-открытого контакта для программирования операции логического умножения (И). Команда представляет логическую операцию и поэтому не может программироваться в начале цепи. В начале логического выражения программируются инструкции LD или LDI.

Применение:

Команда "последовательный нормально-открытый контакт X0" создает последовательную логическую связь с контактом X1 и служит для выполнения операции логического умножения. Если на входе X1 будет "0" и на X0 будет сигнал "1", тогда выход Y1 установится в состояние "1".

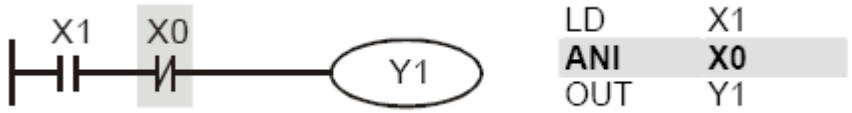
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ANI	Последовательный нормально-закрытый контакт (И-НЕ)	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда ANI используется в качестве последовательного нормально-закрытого контакта для программирования операции И-НЕ. Команда представляет логическую операцию и поэтому не может программироваться в начале цепи. В начале логического выражения программируются инструкции LD или LDI.

Применение:



Команда "последовательный нормально-закрытый контакт X0" создает последовательную логическую связь с контактом X1 и служит для выполнения логической операции И-НЕ. Если на входе X1 будет "1" и на X0 не будет сигнала "1", тогда выход Y1 установится в состояние "1".

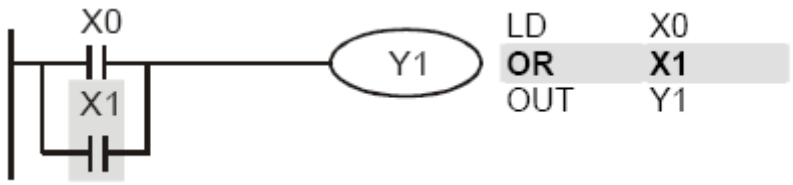
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
OR	Параллельный нормально-открытый контакт (логическое ИЛИ)	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда OR используется в качестве параллельного нормально-открытого контакта для программирования операции логического сложения (ИЛИ). Команда представляет логическую операцию и поэтому не может программироваться в начале цепи. В начале логического выражения программируются инструкции LD или LDI.

Применение:



Команда "параллельный нормально-открытый контакт X1" создает параллельную логическую связь с контактом X0 и служит для выполнения операции логического сложения. Если хотя бы на одном из входов X0 или X1 будет "1", тогда и на выходе Y1 будет состояние "1".

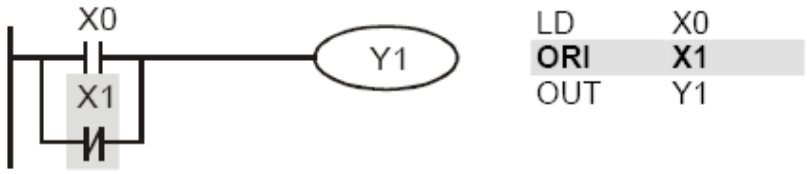
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ORI	Параллельный нормально-закрытый контакт (ИЛИ-НЕ)	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда ORI используется в качестве параллельного нормально-закрытого контакта для программирования логической операции ИЛИ-НЕ. Команда представляет логическую операцию и поэтому не может программироваться в начале цепи. В начале логического выражения программируются инструкции LD или LDI.

Применение:



Команда "параллельный нормально-закрытый контакт X1" создает параллельную логическую связь с контактом X0 и служит для выполнения логической операции ИЛИ-НЕ. Если на входе X0 будет "1" или на входе X1 будет "0" (одно или оба условия одновременно), тогда на выходе Y1 будет состояние "1".

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ANB	«И» блок: последовательное включение блоков контактов	+	+	+

Операнд	нет

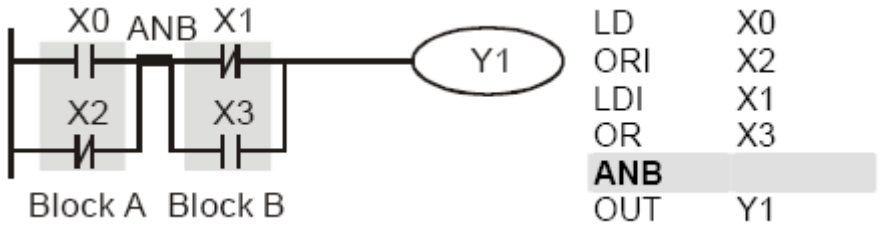
Описание:

- Команда ANB используется для последовательного соединения цепочек из двух групп контактов. Отдельные блоки, параллельно включенных элементов, заносятся в программу раздельно. Чтобы эти блоки соединить последовательно, после каждого блока программируется ANB инструкция.

- Начало разветвления программируется с помощью инструкций LD или LDI.

- ANB-инструкция является независимой и не требует ввода никаких операндов.
- ANB-инструкция внутри всей программы может программироваться многократно.
- В контактной схеме ANB-инструкция изображается как последовательное соединение. ANB-инструкция, имеющаяся на языке списка инструкций (IL), при конвертировании в контактную схему появляется автоматически и изображается как перемычка.
- Если программируется несколько отдельных блоков непосредственно один за другим, то нужно ограничить число LD и LDI инструкций и, соответственно, также число ANB-инструкций до 8.

Применение:



Команда ANB создает последовательную логическую связь между двумя логическими блоками (Block A и Block B).

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ORB	«ИЛИ» блок: параллельное включение блоков контактов	+	+	+

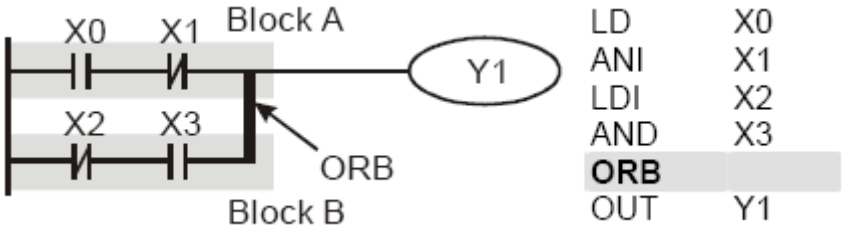
Операнд	нет

Описание:

- Команда ORB используется для параллельного соединения групп контактов. Если несколько последовательных блоков включаются параллельно, то нужно после программирования каждого отдельного блока вводить ORB-инструкцию.
- Начало разветвления программируется с помощью инструкций LD ил и LDI.
- ORB-инструкция является независимой и не требует ввода никаких операндов.
- ORB-инструкция внутри всей программы может программироваться многократно.
- В контактной схеме ORB-инструкция изображается как параллельное соединение. ORB-инструкция, имеющаяся на языке списка инструкций (IL), при конвертировании в контактную схему появляется автоматически и изображается как перемычка.
- Если программируется несколько отдельных блоков непосредственно один за другим, то

нужно ограничить число LD и LDI инструкций и, соответственно, также число ORB-инструкций до 8.

Применение:



Команда ORB создает параллельную логическую связь между двумя логическими блоками (Block A и Block B).

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
MPS	Точка начала разветвления с одним входным условием для всего разветвления	+	+	+

Операнд	нет

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
MRD	Промежуточная точка разветвления с одним входным условием для всего разветвления (ответвление)	+	+	+

Операнд	нет

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
MPP	Точка конца разветвления с одним входным условием для всего разветвления	+	+	+

Операнд	нет

Описание:

- Инструкции MPS, MRD, MPP служат для того, чтобы создавать уровни логических связей – разветвлений. Например, после одного начального логического выражения создать несколько логических выражений на выходе, т.е. включать несколько выходов-катушек от одного входа. С программной точки зрения данные команды представляют собой точки, обладающие памятью результата предыдущих логических операций.

- С помощью команды MPS запоминается предыдущий результат логических связей (обработки логического выражения) и обозначается начало разветвления.

- С помощью инструкции MRD возможно прочтение нескольких частных ответвлений между началом (MPS) и концом (MPP) разветвления, учитывающих на каждом разветвлении результат обработки логического выражения до MPS.

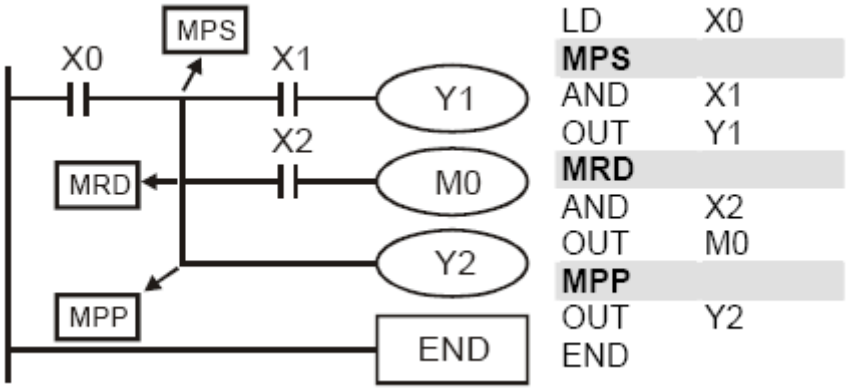
- Последнее частное разветвление создается MPP инструкцией.

- Открывшееся с помощью MPS инструкции разветвление всегда должно быть закрыто MPP инструкцией.

- Все три инструкции не требуют никаких операндов.

- В контактной схеме эти инструкции не изображаются. Если программирование выполняется в контактной схеме, разветвления используются как обычно. MPS-, MRD- и MPP-инструкции на языке списка инструкций (IL) появляются автоматически, после того как программа конвертируется в контактную схему.

Применение:



1) MPS

Промежуточный результат (здесь X0) на 1-ом уровне логических связей занесен на 1-ое место в стековую память промежуточных связей. Выполняется логическое умножение X1 с X0 и устанавливается выход Y1.

2) MRD

Перед выполнением следующей инструкции опрашивается промежуточный результат на 1-ом месте памяти логических связей. Выполняется логическое умножение X2 с X0 и устанавливается выход M0.

3) MPP

Перед выполнением следующей инструкции опрашивается промежуточный результат на 1-ом месте памяти логических связей. Устанавливается выход Y2. Операция на 1-ом уровне

промежуточных результатов завершена, и память логических связей стирается.

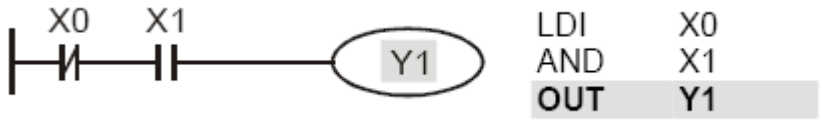
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
OUT	Выход	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	-	+	+	+	-	-	-

Описание:

- Команда OUT служит для включения или отключения выхода в зависимости от результата логических связей (результата обработки центральным процессором логического выражения).
- С помощью инструкции OUT можно завершить программирование строки (логического выражения).
- Программирование нескольких инструкций OUT как результат обработки логического выражения также возможно.
- Результат логических связей, представленный посредством инструкции OUT, может применяться в следующих шагах программы как состояние входного сигнала, т.е. может многократно опрашиваться во многих логических выражениях.
- Результат логических связей, представленный OUT инструкцией, активен (включен) до тех пор, пока действуют условия его включения.
- При программировании двойной записи одинаковых выходов (их адресов) могут возникнуть проблемы при отработке программы. Избегайте двойной записи выходов, так как это приведет к некорректной работе программы.

Применение:



При условии: X0=0 и X1=1 – команда OUT Y1 установит выход контроллера Y1 в состояние "1".

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
SET	Включение выхода с фиксацией	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	-	+	+	+	-	-	-

Описание:

- Когда выполняется входное условие для инструкции SET, то она включает стоящий за ней операнд и фиксирует его состояние независимо от того, действует входное условие или нет.
- С помощью SET могут устанавливаться в "1" (включаться) операнды Y, M или S.

Применение:


```
LD    X0
ANI   Y0
SET   Y1
```

Выход Y1 включится при выполнении условий X0, Y0 и больше от этих условий зависеть не будет. Выключить выход Y1 можно будет только командой RST Y1 или снятием питания с ПЛК.

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
RST	Сброс состояния операнда	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	-	+	+	+	+	+	+

Описание:

Когда выполняется входное условие для инструкции RST, то она выключает стоящий за ней операнд и фиксирует его состояние независимо от того, действует входное условие или нет.

С помощью RST-инструкции могут отключаться:

- Выходы Y, контакты M и операнды состояния шагов S выключаются (состояние сигнала "0").
- Текущее значение таймеров и счетчиков, а также содержание регистров D, E и F сбрасываются на "0".

Применение:


```
LD    X0
RST   Y5
```

Выход Y5 выключится при выполнении условия X0 и останется выключенным даже когда условие X0 выполняться не будет.

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
TMR	Таймер (16 бит)	+	+	+

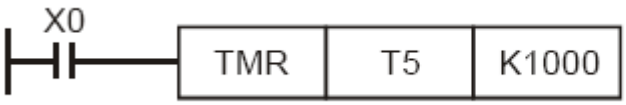
Операнд		
	T – K	T0 – T255, K0 - K32767
	T – D	T0 – T255, D0 – D9999

Описание:

При активации входного условия команда TMR начинает отсчет заданной уставки и по ее достижении замыкает свой контакт.

- С помощью инструкции TMR можно завершить программирование строки (логического выражения).
- Результат логических связей, представленный посредством инструкции TMR, может применяться в следующих шагах программы как состояние входного сигнала, т.е. может многократно опрашиваться во многих логических выражениях.
- Результат логических связей, представленный TMR инструкцией, активен (включен) до тех пор, пока действуют условия его включения.
- Если при отсчете уставки входное условие прекратило действовать до того, как закончился отсчет уставки, то контакт таймера не замкнется, а накопленное значение сбросится на ноль.
- По применению таймера см. Главу 2

Применение:




```
LD      X0
TMR     T5 K1000
```

При условии X0=1 инструкция TMR T5 будет вести отчет времени, пока значение в регистре T5 не достигнет значения K1000 (100 сек), после чего замкнется контакт T5. При X0=0 выполнение инструкции TMR прекратится и T5 сбросится на "0".

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
CNT	Счетчик (16 бит)	+	+	+

Операнд		
	C – K	C0 – C199, K0 - K32767
	C – D	C0 – C199, D0 – D9999

Описание:

- Команда CNT служит для суммирования количества замыканий входного контакта (макс. 32767 импульсов) и присвоения состояния сигнала (включения или отключения выхода) когда текущее значение счетчика достигнет заданного значения.
- С помощью инструкции CNT можно завершить программирование строки (логического выражения).
- Результат логических связей, представленный посредством инструкции CNT, может применяться в следующих шагах программы как состояние входного сигнала, т.е. может многократно опрашиваться во многих логических выражениях.
- Для сброса текущего значения счетчика нужно использовать команду RST.
- См. так же Главу 2 по использованию счетчика.

Применение:


```
LD      X0
CNT     C20 K100
```

При изменении состояния X0 с "0" на "1" значение регистра C20 будет увеличено на 1, и так пока значение в регистре C20 не достигнет значения K100 (100 импульсов). После этого счет прекратится, контакт C20 замкнется и новые импульсы не будут оказывать воздействия ни на состояние контакта, ни на текущее значение счетчика.
Для сброса значения регистра C20 нужно использовать команду RST C20.

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
DCNT	Счетчик (32 бит)	+	+	+

Операнд		
	C – K	C200 – C254, K -2147483648 - K2147483647
	C – D	C200 – C254, D0 – D9999

Описание:

- Инструкция DCNT работает с 32-х разрядными счетчиками C200 ~ C255.
- При работе с 32-х разрядными счетчиками общего назначения C200 ~ C234 инструкция DCNT осуществляет суммирование или вычитание на «1» при появлении одного импульса на входе. Режим счета (вверх или вниз) определяется состоянием реле M1200 ~ M1235.
- Счетчики C235 ~ C255 являются высокоскоростными. Детальное описание их работы дано в Главе 2.
- При отключении инструкции DCNT счет прекращается, но текущее значение сохраняется. Для очистки регистра счетчика и сброса контакта необходимо использовать команду RST.

- Подробное описание счетчиков дано в Главе 2.

Применение:

Ступенчатая диаграмма:

Список инструкций:		Действия:
LD	M0	Нормально-открытый контакт M0
DCNT	**C254 K1000**	Установка заданного значения C254 как K1000

При условии M0 = 1 значение регистра C254 будет увеличено или уменьшено на 1 при каждом изменении состояния соответствующих входах (X0, X1), и так пока значение в регистре C254 не достигнет значения K1000 (1000 импульсов). Для сброса значения регистра C254 нужно использовать команду RST C254

Инструкция	**Функция**	**ПЛК**		
		ES/EX/SS	**SA/SX/SC**	**EH/EH2/SV**
MC/MCR	Начало/конец исключаемого участка программы	+	+	+

Операнд	N0 – N7

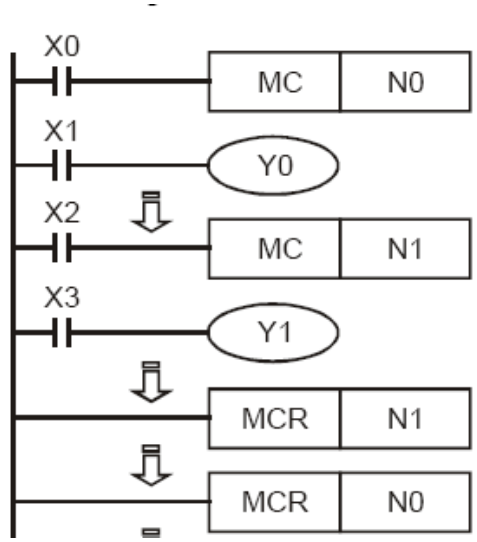
Описание:

- Инструкции MC/MCR позволяют временно отключать определенные участки программы (мастер-контроль).
- Инструкция MC обозначает начало исключаемого участка. По своему назначению данная инструкция является аналогичной главному контакту питающей шины в релейно-контактных схемах, отключающему участок схемы от источника питания.
- Инструкция MCR обозначает конец исключаемого участка программы. Перед инструкцией MCR не должно стоять никаких контактов и условий.
- Инструкции MC/MCR поддерживают до 8 уровней вложенности, но последовательность нумерации должна быть строго по возрастанию: от N0 до N7.
- Если условие включения инструкции MC выполняется, то участок программы между MC и MCR соответствующего номера вложенности будет выполняться без каких-либо ограничений.
- Если условие включения инструкции MC не выполняется, то участок программы до соответствующей инструкции MCR не будет выполняться и операнды примут следующие состояния:

Операнд	**Описание**
Таймеры общего назначения	Текущее значение = 0. Контакты не реагируют, выходы отключены.
Аккумулятивный таймер	Выход отключен, текущее значение и состояние контактов не меняется.
Таймер подпрограмм	Выход отключен, текущее значение и состояние контактов не меняется.
Счетчики	Выходы отключены, текущее значение и состояние контактов не меняется.
Выходы, управляемые инструкцией OUT	Все выключены.
Выходы, управляемые инструкциями SET и RST	Остаются без изменения.
Прикладные инструкции	Все выключены. Циклы FOR-NEXT выполняться заданное количество раз, но операнды внутри них будут вести себя как между MC и MCR.

Примечание: Применение инструкций MC и MCR не сокращает время цикла программы.

Применение:




```
LD    X0
MC    N0
LD    X1
OUT   Y0
LD    X2
MC    N1
LD    X3
OUT   Y1
MCR   N1
MCR   N0
```

Область программы между инструкциями MC N0 и MCR N0 будет выполняться только если X0=1. Область программы между инструкциями MC N1 и MCR N1 будет выполняться только если X0=1 и X2=1.

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
LDP	Начало логического выражения с опросом по переднему фронту входного импульса	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

- Команда LDP используется для программирования импульсного начала логической связи.
- Инструкция LDP должна программироваться в начале цепи.
- LDP-инструкция используется также вместе с инструкциями ANB и ORB для запуска разветвлений.
- LDP-инструкция после положительного фронта сохраняется на время цикла программы (скана).

Применение:


```
LDP   X0
AND   X1
OUT   Y1
```

Команда "LDP X0" открывает последовательную логическую связь.

Если вход X0 изменит свое состояние с "0" на "1" (при этом X1=1), тогда выход Y1 будет в состоянии "1" в течение одного скана.

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
LDF	Начало логического выражения с опросом по заднему фронту входного импульса	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

- Команда LDF используется для программирования импульсного начала логической связи.
- Инструкция LDF должна программироваться в начале цепи.
- LDF-инструкция используется также вместе с инструкциями ANB и ORB для запуска разветвлений.
- LDF-инструкция после отрицательного фронта сохраняется на время цикла программы (скана).

Применение:


```
LDF   X0
AND   X1
OUT   Y1
```

Команда "LDF X0" открывает последовательную логическую связь. Если вход X0 изменит свое состояние с "1" на "0" (при этом X1=1), тогда выход Y1 будет в состоянии "1" в течение одного скана.

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ANDP	Логическое «И» с опросом по переднему фронту входного импульса	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда ANDP используется для программирования последовательного соединения импульсного контакта с опросом по переднему фронту.

Применение:


```
LD    X0
ANDP  X1
OUT   Y1
```

Команда "ANDP X1" создает последовательную логическую связь. Если вход X1 изменит свое состояние с "0" на "1" (при этом X0=1), тогда выход Y1 будет в состоянии "1" в течение одного скана.

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ANDF	Логическое «И» с опросом по заднему фронту входного импульса	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда ANDF используется для программирования последовательного соединения импульсного контакта с опросом по заднему фронту.

Применение:


```
LD    X0
ANDF  X1
OUT   Y1
```

Команда "ANDF X1" создает последовательную логическую связь. Если вход X1 изменит свое состояние с "1" на "0" (при этом X0=1), тогда выход Y1 будет в состоянии "1" в течение одного скана.

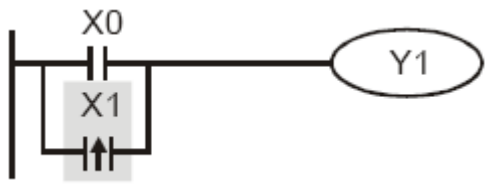
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ORP	Логическое «ИЛИ» с опросом по переднему фронту входного импульса	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда ORP используется для программирования параллельного соединения импульсного контакта с опросом по переднему фронту.

Применение:




```
LD    X0
ORP   X1
OUT   Y1
```

Команда "ORP X1" создает параллельную логическую связь. Выход Y1 будет в состоянии "1" в течение одного скана если вход X1 изменит свое состояние с "0" на "1" или X0=1.

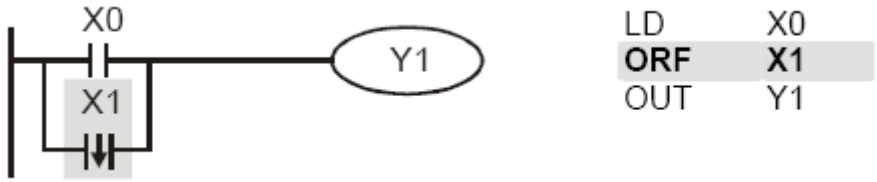
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
ORF	Логическое «ИЛИ» с опросом по заднему фронту входного импульса	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	+	+	+	+	+	+	-

Описание:

Команда ORF используется для программирования параллельного соединения импульсного контакта с опросом по заднему фронту.

Применение:



Команда " ORF X1" создает параллельную логическую связь. Выход Y1 будет в состоянии "1" в течение одного скана если вход X1 изменит свое состояние с "1" на "0" или X0=1.

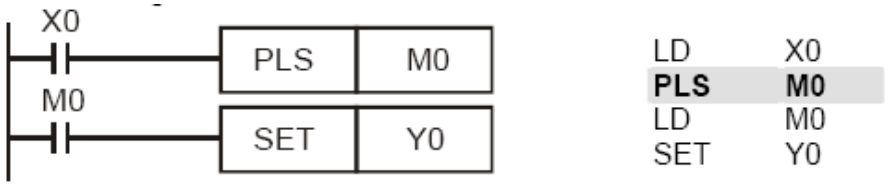
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
PLS	Создание импульса на выходе по переднему фронту входного сигнала	+	+	+

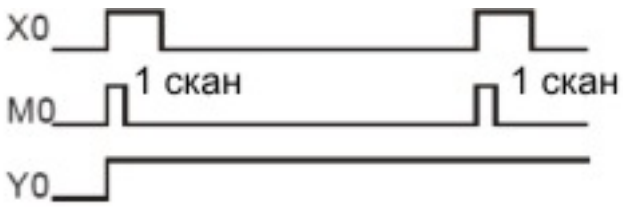
Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	-	+	+	-	-	-	-

Описание:

Команда PLS формирует на выходе один импульс длиною в 1 скан по переднему фронту входного сигнала, независимо от его продолжительности.

Применение:





При изменении входного сигнала на входе X0 с "0" на "1" (возрастающий фронт) реле M0 благодаря PLS-инструкции включается на время одного скана. Контакт M0 запускает команду SET, которая включает выход Y0.

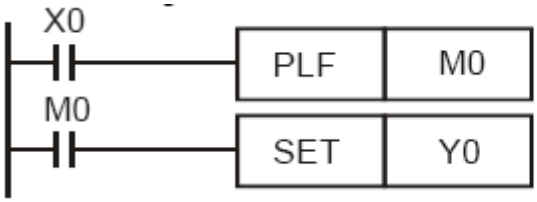
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
PLF	Создание импульса на выходе по заднему фронту входного сигнала	+	+	+

Операнд	X0 – X377	Y0 – Y377	M0-M4095	S0-S1023	T0 – T255	C0 – C255	D0-D9999
	-	+	+	-	-	-	-

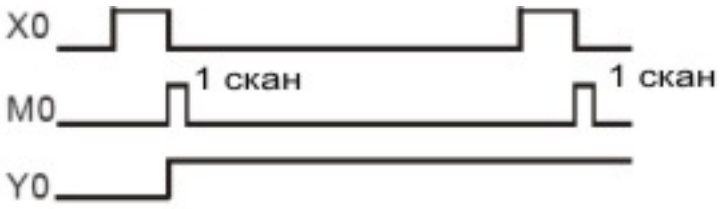
Описание:

Команда PLF формирует на выходе один импульс длиною в 1 скан по заднему фронту входного сигнала, независимо от его продолжительности.

Применение:




```
LD    X0
PLF   M0
LD    M0
SET   Y0
```



При изменении входного сигнала на входе X0 с "1" на "0" (спадающий фронт) реле M0 благодаря PLF-инструкции включается на время одного скана. Контакт M0 запускает команду SET, которая включает выход Y0.

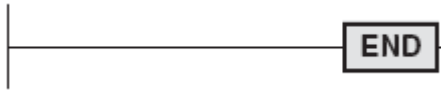
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
END	Конец программы	+	+	+

Операнд	нет

Описание:

Любая программа для контроллера (ступенчатая диаграмма или список инструкций) должна заканчиваться командой END. Контроллер осуществляет сканирование программы с шага «0» до команды END, а затем снова возвращается к шагу «0». После выполнения команды END осуществляется установка выходов, а также сброс текущего времени цикла сторожевого таймера (Watch-Dog-Timer).

Применение:



Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
NOP	Пустая строка в программе	+	+	+

Операнд	нет

Описание:

Инструкция NOP не осуществляет в программе никаких действий. Таким образом, после ее выполнения сохраняются логические состояния всех предыдущих шагов программы. Инструкцию NOP применяют в тех случаях, когда необходимо удалить какую-либо действующую инструкцию, сохранив при этом длину программы, или зарезервировать место под какое-либо действие. Применяется в языке «список инструкций».
Количество NOP инструкций в программе не ограничено. После завершения отладки программы инструкции NOP желательно удалить, так как они бесполезно удлиняют программу, увеличивая время цикла.

Применение:

Ступенчатая диаграмма

Список инструкций

```
LD    X0
NOP
OUT   Y1
```

Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
INV	Инверсия - замена результата логических связей на противоположный	+	+	+

Операнд	нет

Описание:

- INV-инструкция инвертирует состояние сигнала результата стоящей впереди инструкции.
- Полученная согласно обработки "1", после инверсии становится "0".
- Полученный согласно обработки "0", после инверсии становится "1".
- INV-инструкция может применяться, как AND и ANI инструкции.

- INV-инструкция может применяться для реверсирования сигнала результата комплексной схемы.
- INV-инструкция может применяться для реверса сигнала результата импульсных инструкций LDP, LDF, ANP и т. д.

Применение:

Если X0 = 0, выход Y1 = 1. Если X0 = 1, выход Y1 = 0.

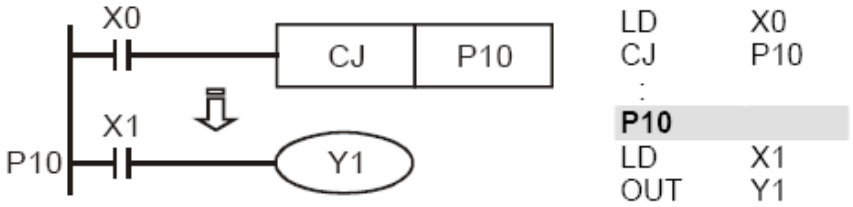
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
P	Указатель точки перехода	+	+	+

Операнд	P0 – P255

Описание:

- P-инструкция служит для указания точки перехода для команд CJ, CALL.
- Номер точки в программе не должен повторяться.

Применение:



Точка P10 указывает адрес перехода программы при выполнении инструкции CJ P10.

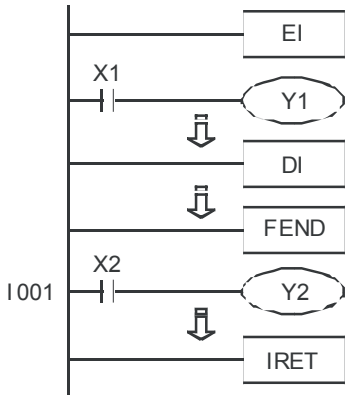
Инструкция	Функция	ПЛК		
		ES/EX/SS	SA/SX/SC	EH/EH2/SV
I	Указатель точки прерывания	+	+	+

Операнд	I00□, I10□, I20□, I30□, I40□, I50□, I6□□, I7□□, I8□□ I010, I020, I030, I040, I050, I060, I110, I120, I130, I140, I150, I160, I170, I180

Описание:

I-инструкция служит для указания точки перехода к подпрограмме обработки прерывания для команд IRET (API 03), EI (API 04), DI (API 05), см. также Параграф 2.9.

Применение:




```
EI          Разрешение прерывания
LD    X1    Норм. откр. контакт  X1
OUT   Y1    Выход  Y1
 :
DI          Запрещение  прерывания
 :
FEND        Конец  основной  программы
I001        Точка подпр. обработки прерывания
LD    X2    Норм.откр.контакт  X2
OUT   Y2    Выход  Y2
 :
IRET        Конец подпрограммы обраб. прерывания
```

ГЛАВА 4

Инструкции пошагового управления контроллеров Delta DVP

4.1 Инструкции STL и RET

Пошаговое управление позволяет удобным способом организовать циклически повторяющиеся технологические процессы. Для инициализации шага используется инструкция STL, при появлении которой программа переходит в режим шаговой ступенчатой диаграммы. Инструкция RET обозначает конец участка с пошаговым управлением, и программа возвращается в режим обычной ступенчатой диаграммы.

Инструкция	Функция	Операнд	ПЛК		
			ES/EX/SS	SA/SX/SC	EH/EH2/SV
STL	Начало участка программы с пошаговым режимом	S0 – S1023	+	+	+

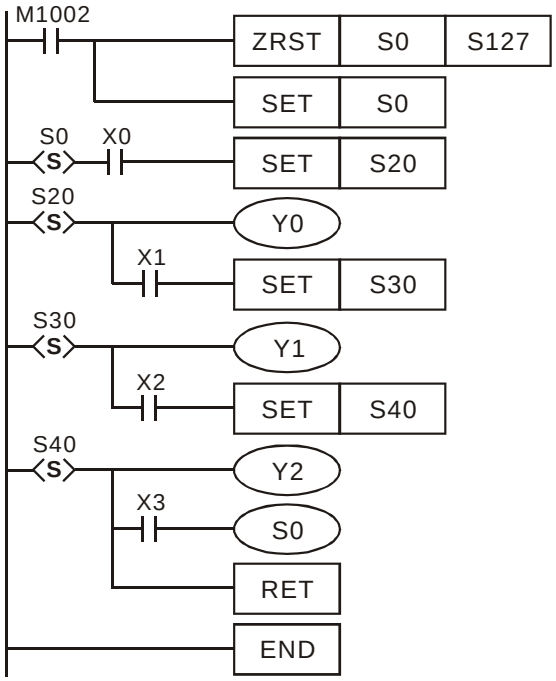
Данная инструкция обозначается в ступенчатой диаграмме операндом S (аббревиатуру STL набирать в программе нигде не надо). Для ввода участка программы с пошаговым режимом необходимо использовать операнды S0 ~ S9, а для обозначения шагов внутри участка с пошаговым исполнением программы используются операнды S10 ~ S1023. Номера шагов (операнды S) в программе не должны повторяться.

Инструкция	Функция	Операнд	ПЛК		
			ES/EX/SS/	SA/SX	EH
RET	Конец участка программы с пошаговым режимом	нет	+	+	+

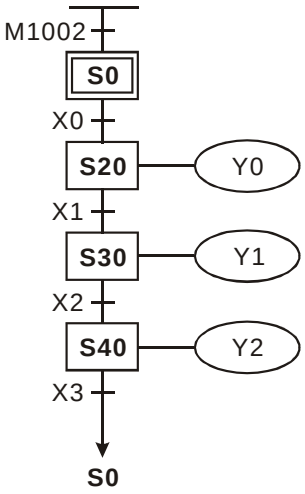
Данная инструкция обозначает конец участка с пошаговым управлением. После данной инструкции программа переходит в обычный режим. Программа контроллера может содержать максимум десять участков с пошаговым управлением (S0 ~ S9) и каждый участок должен заканчиваться инструкцией RET.

Пример:

Ступенчатая диаграмма (язык LD):



Последовательная функциональная диаграмма (язык SFC):



4.2 Последовательные функциональные диаграммы (язык SFC, Sequential Function Chart)

В промышленной автоматизации любой циклический процесс можно разбить на последовательность шагов. В каждом шаге реализуется определенное действие в зависимости от входных сигналов. Переход к следующему шагу осуществляется только после полной отработки предыдущего шага и срабатывании условия перехода к следующему шагу. При переходе к следующему шагу все результаты предыдущего шага сбрасываются. Данная логика и была положена в основу Последовательных Функциональных Диаграмм (SFC).
Если шаговый режим используется внутри традиционных Ступенчатых Диаграмм (LD), то данный участок программы называют Шаговые Ступенчатые Диаграммы. Программа может содержать до 10 независимых участков с пошаговым управлением.

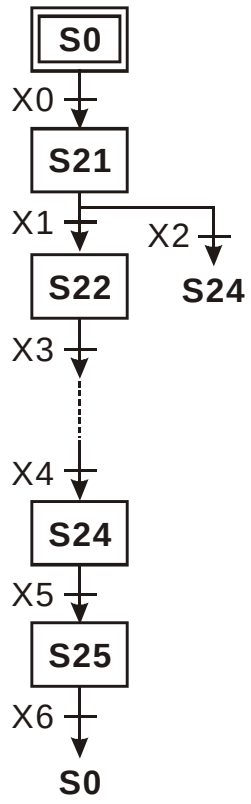
Режим пошагового управления в виде последовательной функциональной диаграммы очень удобен для машин-автоматов, работающих циклично по относительно несложной логике, и имеет два основных преимущества:

1. Позволяет использовать несколько раз один и тот же выход (Y).
2. Действия на последовательной функциональной диаграмме легко понять, изменить в ходе наладки, найти ошибку и осуществлять дальнейшую эксплуатацию.

Ниже приводится пример и комментарии:

1. Язык SFC является одной из разновидностей инструментов редактирования диаграмм. Структура диаграммы SFC очень похожа на технологическую блок-схему процесса. Каждый номер реле S соответствует конкретному шагу внутри ПЛК, олицетворяющему собой определенную процедуру на технологической блок-схеме. Когда текущая процедура окончена, программа переходит к следующему шагу согласно заданному условию. Таким образом, процесс может повторяться циклично неограниченное количество раз.
2. На рисунке справа приведен пример последовательной диаграммы, которая работает следующим образом: шаговый режим инициализируются реле S0, которое передает шаг к реле S21 при замыкании контакта X0. Реле S21 передает шаг реле S22 при замыкании контакта X1 или перескакивает на реле S24 при замыкании контакта X2. Подобным образом происходит движение шаг за шагом вниз по диаграмме до тех пор пока реле S25 при замыкании контакта X6 не передаст шаг снова к S0 и программа вернется на начало и повторит цикл сверху вниз.

SFC:



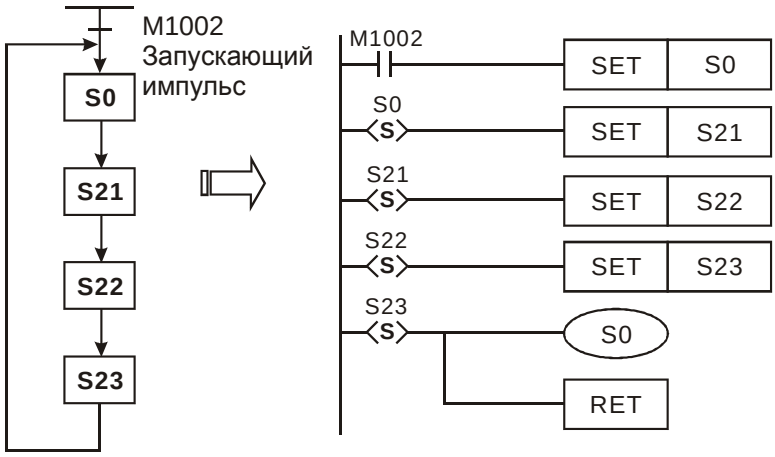
Для редактирования последовательных функциональных диаграмм используются следующие условные обозначения:

LAD	Символ используется для перехода в режим релейно-контактных схем (LD) и обозначает, что программа находится в режиме обычных ступенчатых диаграмм, а не в режиме шаговых ступенчатых диаграмм.
	Символ используется для программирования инициализирующих шаговых операндов S0...S9.
	Символ используется для программирования шаговых операндов общего назначения S10...S1023.
	Символ используется для программирования команды перехода в заданную точку диаграммы: переход вверх или вниз к не прилегающим точкам, возвращение на начальный шаг или переход к другой диаграмме.
	Символ используется для программирования условных переходов между отдельными шагами в программе.
	Символ используется для программирования селективного разветвления в программе. Текущий шаг переходит к какому-либо из следующих шагов в зависимости от того, какое из входных условий активировалось.

	Символ используется для программирования сборки (окончания) селективного разветвления.
	Символ используется для программирования параллельного разветвления в программе, при котором текущий шаг разветвляется на два или несколько процессов, обрабатывающихся одновременно.
	Символ используется для программирования сборки (окончания) параллельного разветвления.

4.3 Как работает инструкция пошагового управления

Инструкция STL применяется для организации синтаксиса последовательной функциональной диаграммы (SFC), делая процесс написания программы аналогичным созданию технологической блок-схемы процесса. Это делает процесс более простым и наглядным, а программу более «читаемой».
На рисунке слева внизу приведен пример последовательной диаграммы, который демонстрирует четкую логику и наглядность построения процесса. На рисунке справа показан тот же фрагмент, но конвертированный в ступенчатые диаграммы.



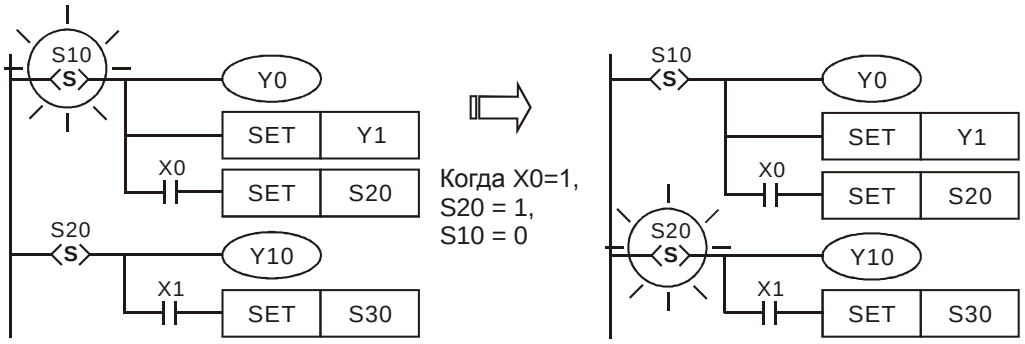
В конце каждой последовательной диаграммы должна стоять инструкция RET, обозначающая конец данной диаграммы и возвращающая программу в обычный режим ступенчатых диаграмм. Всего в программе может быть до 10 последовательных диаграмм (участков с пошаговым управлением) и каждый из них должен заканчиваться инструкцией RET.

Каждый шаг представляет собой набор определенных управляющих процедур. В каждый момент выполняются процедуры только одного шага. Для должной работы шаг должен строиться по следующей схеме:

- Установка состояния выхода
- Проверка входного условия
- Определение номера следующего шага, к которому переходит управление

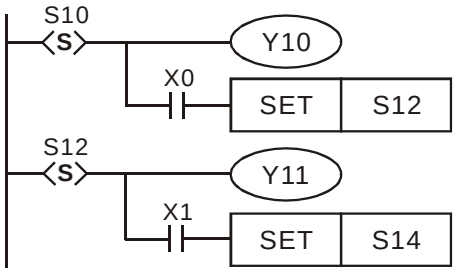
Далее приведены примеры реализации пошагового управления.

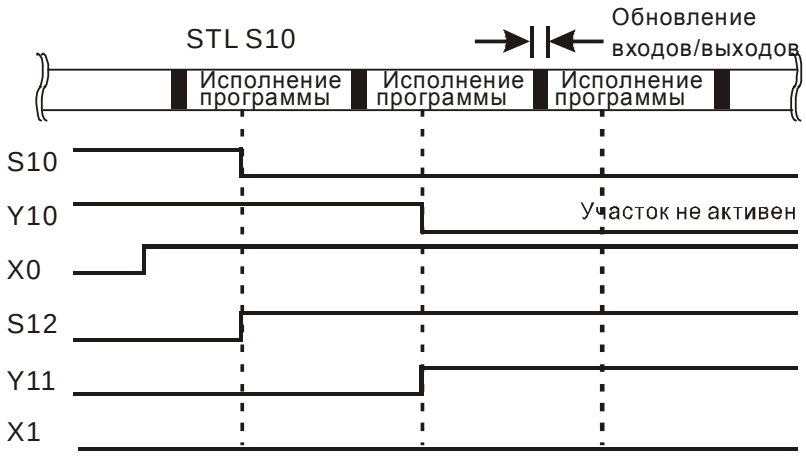
Пример 1



Комментарии:
Когда S10 включится, то Y0 и Y1 тоже включатся. Когда замкнется X0, управление перейдет к шагу S20 и реле S20 включится, а реле S10 выключится. Соответственно выход Y0 отключится, выход Y10 включится, а выход Y1 останется включенным, так как он был зафиксирован инструкцией SET.

Примечание:
Когда замыкается контакт Sn, то активируется относящийся к нему участок программы. После того, как контакт Sn разомкнется, его участок программы дезактивируется. Время задержки дезактивации составит 1 скан программы.



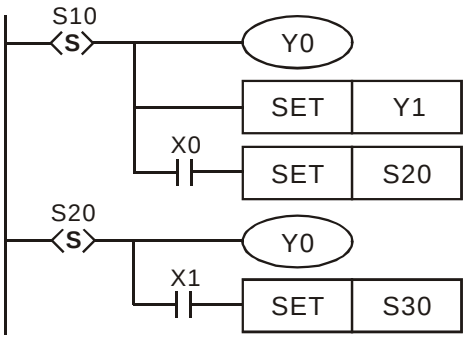


При передаче управления от S10 к S12 (происходит одновременно), после задержки в 1 скан выход Y10 выключится, а выход Y11 включится. Наложения времени работы выходов друг на друга не будет.

Пример 2
Повторное использование в программе одного и того же выхода.

В обычных ступенчатых диаграммах включение одного и того выхода от разных входных условий в разных частях программы является затруднительным, так как может привести к непредсказуемым действиям, и поэтому не применяется.
При использовании пошагового режима данная проблема снимается, так как каждый шаг активирует только свой участок программы, а остальная программа не активна.

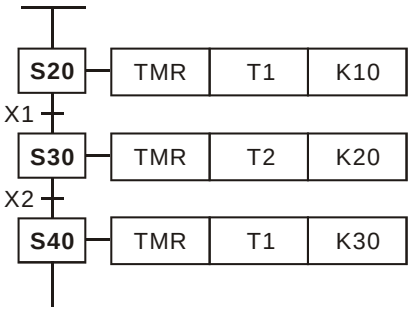
Вследствие этого, если выход и повторяется где-то в программе, то это не окажет влияния на текущий активный участок программы. Единственным ограничением является недопустимость включения выхода командой SET, так как она фиксирует включенное состояние выхода даже при переходе программы на следующий шаг (см. Пример 1).



Комментарии:
Выход Y0 будет включен и при включении S10 и при включении S20. Однако надо иметь ввиду, что при передаче управления от S10 к S20 выход Y0 будет выключен (а это может быть через какие-либо промежуточные участки программы, выполнение которых займет время).

Важное замечание:
Нельзя использовать те же выходы, которые задействованы в участках программы с шаговым управлением, в участках программы с обычными ступенчатыми диаграммами.

Пример 3
Повторное использование таймера.



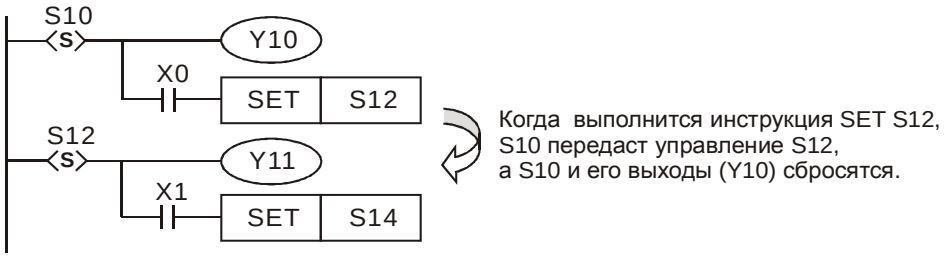
Контакт таймера может использоваться в качестве выхода, и в пошаговом режиме возможно его неоднократное использование. В контроллерах типов ES/EX/SS/SA/SX/SC нельзя повторно использовать таймер в прилегающих друг к другу шагах.

Важное замечание:
Нельзя использовать те же таймеры, которые задействованы в участках программы с шаговым управлением, в участках программы с обычными ступенчатыми диаграммами.

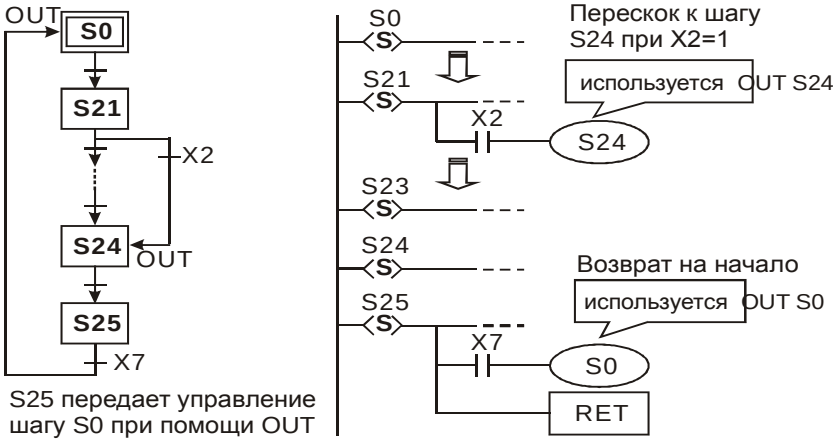
Пример 4
Особенности применения инструкций SET и OUT для перехода к следующему шагу.

В шаговом режиме для организации перехода от одного шага к другому используются две инструкции SET и OUT. В программе может быть несколько шаговых последовательностей и переход может осуществляться как к следующему шагу одной последовательности, так и к шагу другой последовательности. Ввиду этого, существуют некоторые отличия в применении инструкций SET и OUT в качестве включателя следующего шага, которые рассмотрены ниже:

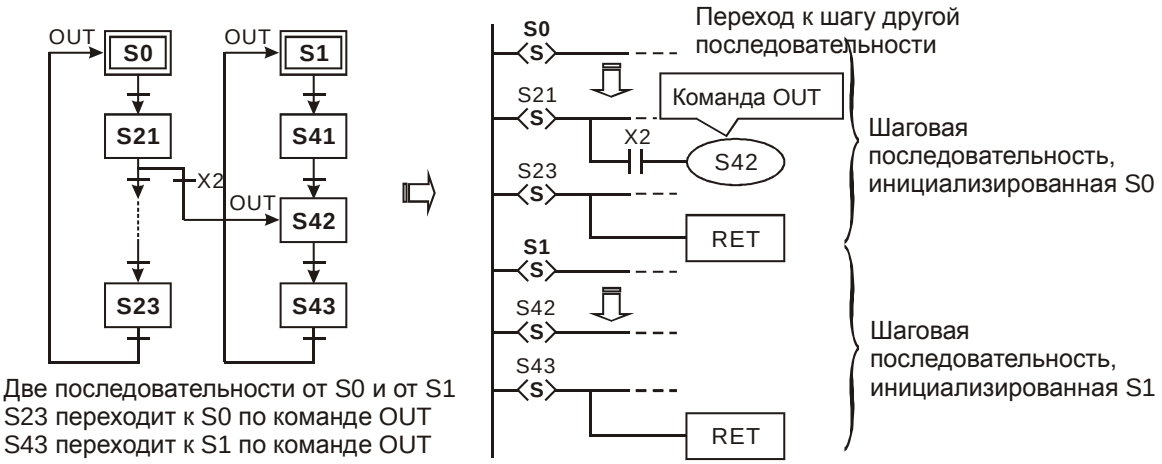
Инструкция SET применяется для перехода к следующему шагу только внутри одной последовательности. После перехода все выходы предыдущего шага сбрасываются.



Инструкция OUT применяется для перехода на начало последовательности (к инициализирующему реле), для перехода к неприлегающим шагам в рамках одной последовательности, а также для перехода к шагу другой последовательности (инициализированной другим стартовым реле). После перехода к следующему шагу, выходы предыдущего шага сбрасываются.



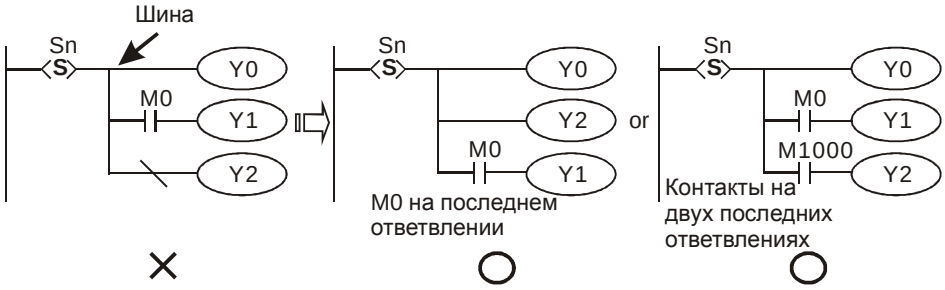
На примере слева показан возврат на начало последовательности, а на рисунке справа показан переход от шага S21 к неприлегающему шагу S24, а затем возвращение на начало к инициализирующему реле S0. На нижнем рисунке демонстрируется переход к шагу другой последовательности



Правила применения шаговых реле

1. Правильное положение условия для включения выходов

Если после шагового реле идет разветвление от основной командной линии (шины), то нельзя располагать нормально открытый (LD) или закрытый контакты (LDI) на промежуточном ответвлении, за которым идут еще ответвления. Для корректной работы контакт необходимо поместить на последнем ответвлении. В противном случае будет выдана ошибка при компиляции. Данная особенность проиллюстрирована на рисунке ниже:



неправильно правильно правильно

2. Ограничения на использование шаговых реле с некоторыми командами

Инструкции, которые можно применять в шаговых последовательностях:

Инструкция / Шаговые реле		LD/LDI/LDP/LDF AND/ANI/ANDP/ANDF OR/ORI/ORP/ORF INV/OUT/SET/RST	ANB/ORB MPS/MRD/MPP	MC/MCR
Инициализирующий/обычный шаг		Да	Да	Нет
Разветвление/ схождение	Обычный выход	Да	Да	Нет
	Передача шага	Да	Нет	Нет

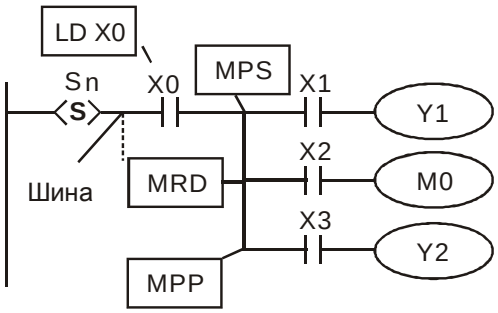
- Инструкции мастер-контроля MC/MCR не могут быть использованы в режиме пошагового управления.

- STL-инструкции нельзя использовать в общих подпрограммах и подпрограммах обработки прерывания.
- Команда CJ может быть использована в режиме пошагового управления, однако это затруднит работу и лучше ее не применять.

3. Положение инструкций MPS/MRD/MPP при использования языка IL

Инструкции разветвления MPS/MRD/MPP нельзя применять сразу после шагового реле. Сначала необходимо выполнить инструкцию LD или LDI.

Ступенчатые диаграммы

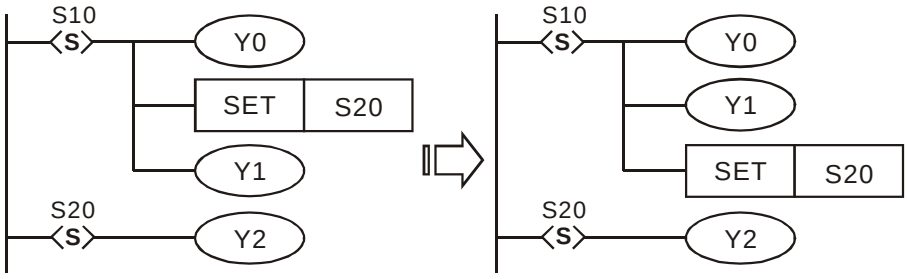


Список инструкций

```
STL     Sn
LD      X0
MPS
AND     X1
OUT     Y1
MRD
AND     X2
OUT     M0
MPP
AND     X3
OUT     Y2
```

4. Правильное положение команд передачи управления следующему шагу и окончания последовательности

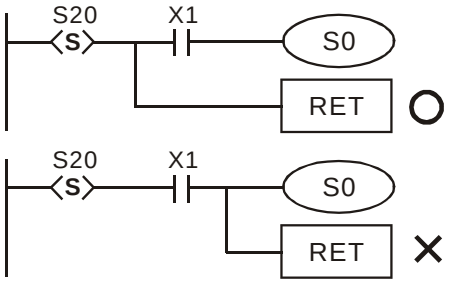
Переход к следующему шагу осуществляется только после выполнения всех действий в текущем шаге. Однако для более правильной работы программы команда передачи управление следующему шагу должна располагаться после всех остальных команд, см. рисунок ниже:



нежелательно правильно

Инструкция RET обязательно должна стоять сразу после последней инструкции STL шаговой последовательности.

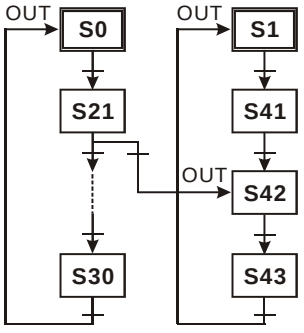
На рисунке ниже правильным является верхний вариант расположения инструкции RET.



4.4 Правила построения шаговых последовательностей

1. Шаговая последовательность должна начинаться с инициализирующего реле S0~S9, а заканчиваться инструкцией RET.
2. Если инструкция STL не используется (не используются S0~S9), то шаговые реле S10~S1023 можно использовать как обычные промежуточные реле.
3. Если инструкция STL используется, то шаговые реле не должны повторятся (реле с определенным номером должно использоваться только один раз).
4. Существуют следующие типы шаговых последовательностей:
   - Одиночная последовательность. В программе существует только одна последовательность без селективных и параллельных разветвлений.
   - Сложная одиночная последовательность. В программе существует только одна последовательность с селективным или с параллельным разветвлением с последующей сборкой.
   - Множественность последовательностей. В программе существуют несколько последовательностей.
5. В множественности последовательностей можно осуществлять перенос управляющего шага из одной последовательности в другую. Данный момент проиллюстрирован ниже:

   В программе существует две последовательности: S0 и S1. Программа сначала записывает состояния шагов реле S0~S30, а затем S1~S43. Однако можно после любого шага перейти в другую последовательность, используя команду OUT (выходная катушка). В данном примере при выполнении условия после шагового реле S21 последовательности S0 произойдет переход к реле S42 последовательности S1.



6. Для организации разветвлений в рамках одной последовательности можно использовать не более 8 шагов с ответвлениями. В программе может содержаться максимум 16 последовательностей. Это могут быть несколько независимых последовательностей + параллельные ветви одной последовательности, или только параллельные ветви одной последовательности, которые при одновременном исполнении рассматриваются как отдельные процессы и их должно быть также не более 16.

7. Для сброса шаговых реле необходимо использовать команду ZRST, а чтобы запретить включение выхода Y можно включить специальное реле M1034=1.
8. Шаговые реле подразделяются на общие и энергонезависимые. При написании программы необходимо учитывать, что энергонезависимые реле сохраняют свое состояние при отключении питания. Изменить диапазон адресов энергонезависимых реле можно записав требуемые значения в соответствующие регистры (см. Главу 2).
9. Для мониторинга шагового режима существуют следующие специальные реле и регистры (по операндам инструкции IST см. пункт 4.6):

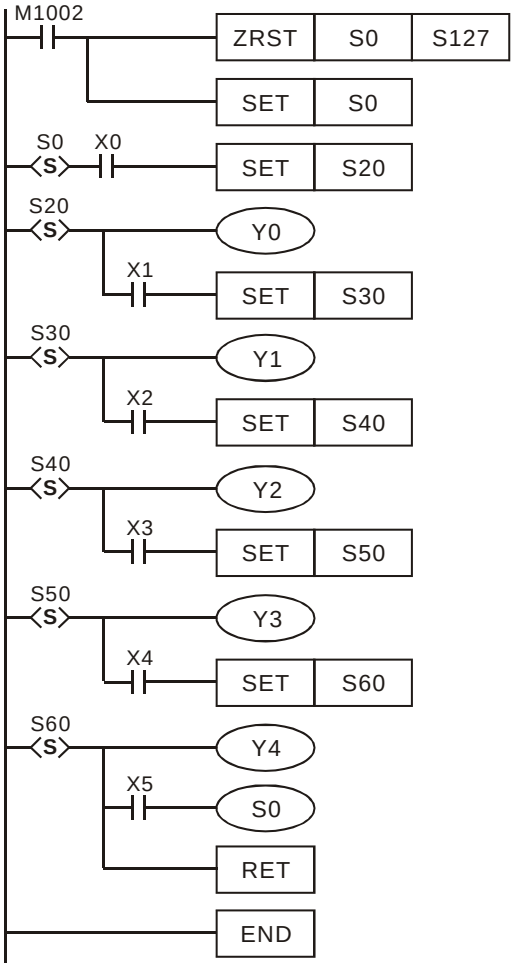
Операнд	Функция
M1040	Флаг запрета передачи управляющего шага. Когда M1040=1 останавливается выполнение шаговой последовательности.
M1041	Инициация следующего шага. Флаг инструкции IST.
M1042	Разрешение импульсов. Флаг инструкции IST.
M1043	Возвращение в нулевую точку завершено. Флаг инструкции IST.
M1044	Разрешение непрерывного режима работы. Флаг инструкции IST.
M1045	Запрещение сброса всех выходов. Флаг инструкции IST.
M1046	M1046=1, если какой-либо из шагов включен (режим пошагового управления).
M1047	Разрешение мониторинга выполнения режима пошагового управления (STL).
D1040	Номер реле Sn предыдущего шага 1 (режим STL).
D1041	Номер реле Sn предыдущего шага 2 (режим STL).
D1042	Номер реле Sn предыдущего шага 3 (режим STL).
D1043	Номер реле Sn предыдущего шага 4 (режим STL).
D1044	Номер реле Sn предыдущего шага 5 (режим STL).
D1045	Номер реле Sn предыдущего шага 6 (режим STL).
D1046	Номер реле Sn предыдущего шага 7 (режим STL).
D1047	Номер реле Sn предыдущего шага 8 (режим STL).

4.5 Типы шаговых последовательностей

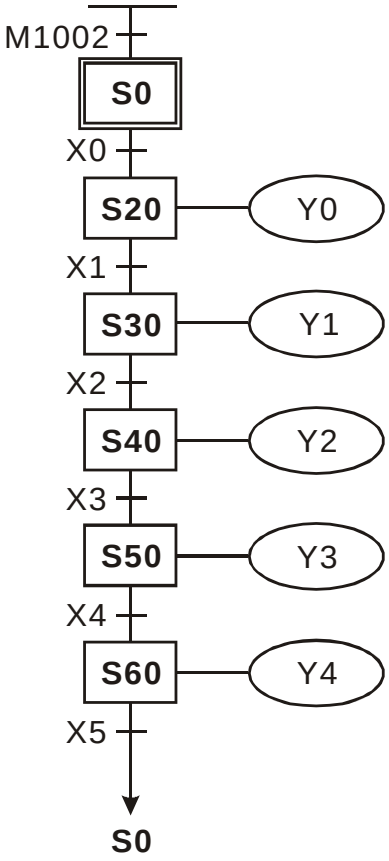
Одиночная последовательность. Является базовым и наиболее распространенным типом шаговой последовательности. Не содержит каких-либо разветвлений.

Пример 1
Одиночная последовательность без разветвлений.

Шаговая ступенчатая диаграмма



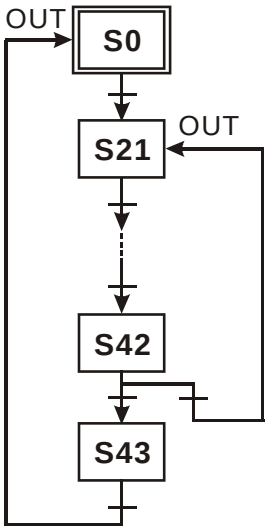
Последовательная функциональная диаграмма (SFC)



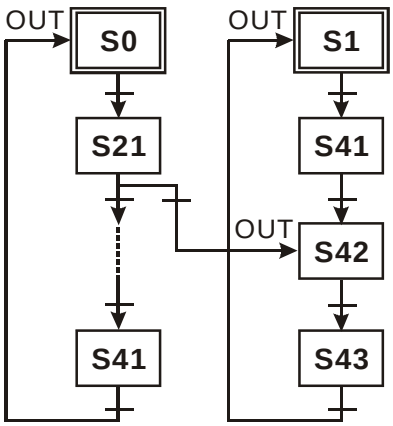
Пример 2

Передача управления к неприлегающему шагу (скачок).

Управление передается в рамках одной последовательности к неприлегающему шагу выше.

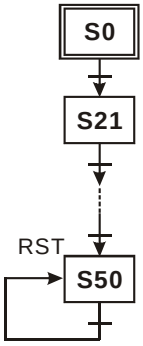


Управление передается шагу в другой последовательности. Обе последовательности зациклены.



Пример 3
Последовательность с автосбросом.

После прохождения одного цикла последовательность сама себя сбросит и процесс остановится. Для повторного запуска потребуется срабатывание входного условия для инициализирующего реле S0.



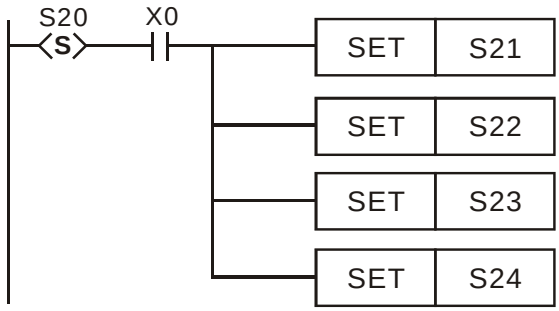
Сложная одиночная последовательность. Содержит селективные, параллельные или комбинированные разветвления и сборки.

1. Структура параллельного разветвления

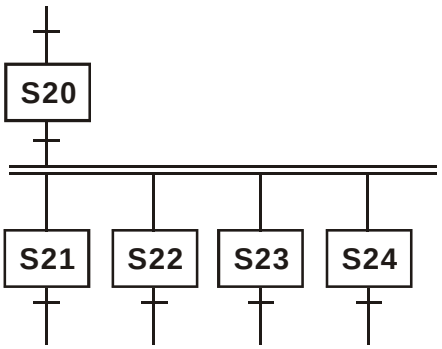
Смысл параллельного разветвления заключается в том, что управление от текущего шага переходит сразу к нескольким шагам, запуская тем самым несколько параллельных процессов, имеющих общее начало.

На примере ниже при замыкании S20 и X20 управление от S20 перейдет одновременно к шаговым реле S21, S22, S23 и S24.

Ступенчатая диаграмма:



Последовательная диаграмма (SFC):

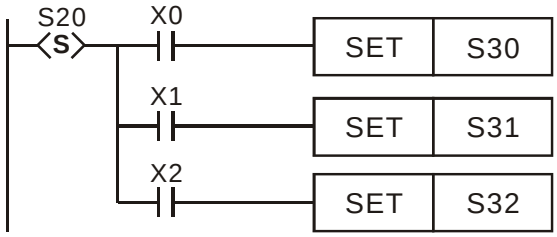


2. Структура селективного разветвления

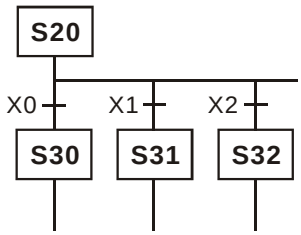
Смысл селективного разветвления заключается в том, что управление от текущего шага переходит только к одному из нескольких возможных шагов. Тем самым процесс как бы переходит из одного русла в другое.

На примере ниже управление от реле S20 перейдет к S30 при замыкании X0, к S31 при замыкании X1 или к S32 при замыкании X2.

Ступенчатая диаграмма:



Последовательная диаграмма (SFC):

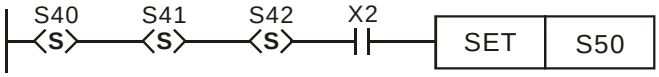


3. Структура параллельной сборки разветвления

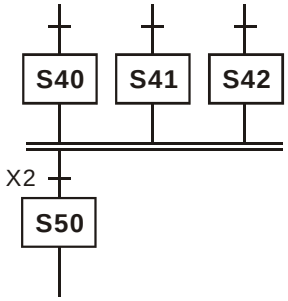
Смысл параллельной сборки в том, что контакты шаговых реле соединяются по схеме «И» (последовательно), и тем самым управление перейдет к следующему шагу, только если все условия будут одновременно выполняться.

На примере ниже управление перейдет к S50, только если одновременно замкнуться S40, S41, S42 и X2.

Ступенчатая диаграмма:



Последовательная диаграмма (SFC):

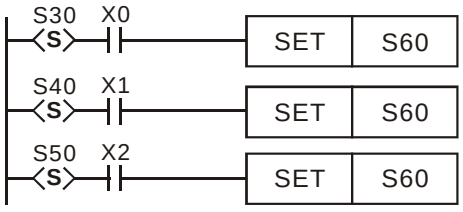


4. Структура селективной сборки разветвления

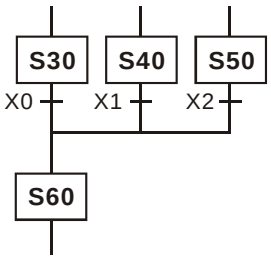
Смысл селективной сборки в том, что контакты шаговых реле соединяются по схеме «ИЛИ» (параллельно), и тем самым обеспечивается переход управления к следующему шагу от одного из нескольких реле.

На примере ниже реле S60 примет управление или от S30, или от S40, или от S50, в зависимости от того, которое из них первым замкнется (одновременно с соответствующим контактом X).

Ступенчатая диаграмма:



Последовательная диаграмма (SFC):



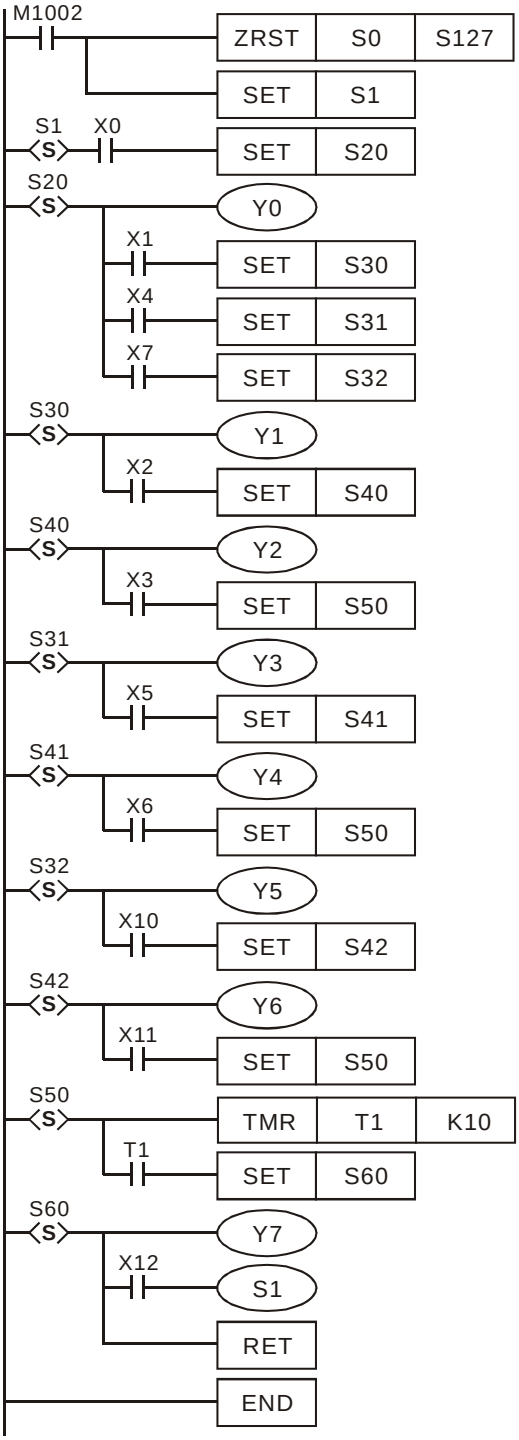
Пример 1
Одиночная последовательность с селективным разветвлением и селективной сборкой.

Ниже проиллюстрирована программа в виде ступенчатой и последовательной диаграмм, которая работает следующим образом:
При переводе контроллера в режим «Работа» замыкается реле M1002 и выдает запускающий импульс. Инструкция ZRST сбрасывает реле S0~S127, команда SET включает инициализирующее реле S1, которое запускает свою шаговую последовательность.

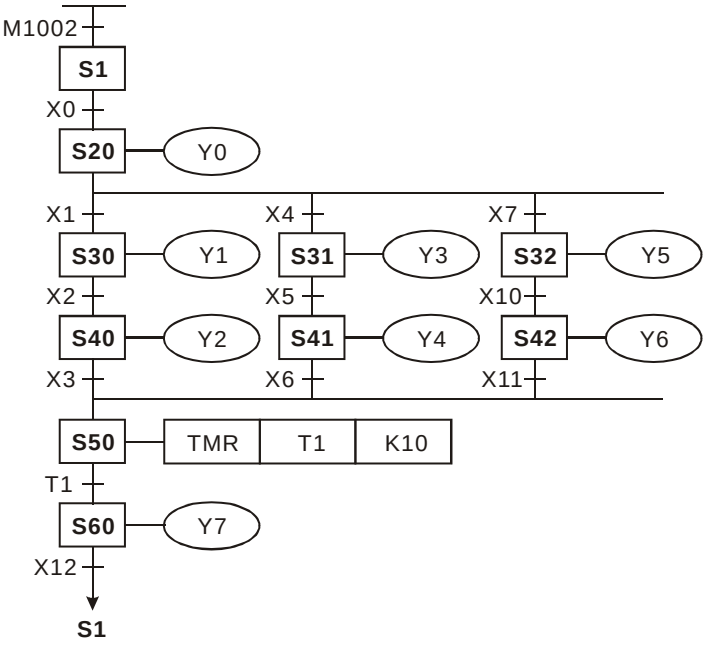
После реле S20 идет селективное разветвление на три возможных варианта: при замыкании X1 управление перейдет к S30, при замыкании X4 к S31, а X7 к S32. Каждое из данных реле (S30~S32) содержит свой процесс, заканчивающийся передачей управления к шагу S50 независимо от того, по какой из ветвей пошел процесс.

Реле S50 в свою очередь отрабатывает таймер T1 и передает управление реле S60, которое возвращает процесс на начало к шагу S1. Таким образом, последовательность будет работать циклично до тех пор, пока контроллер находится в состоянии «Работа».

Ступенчатая диаграмма:



Последовательная диаграмма (SFC):



Пример 2

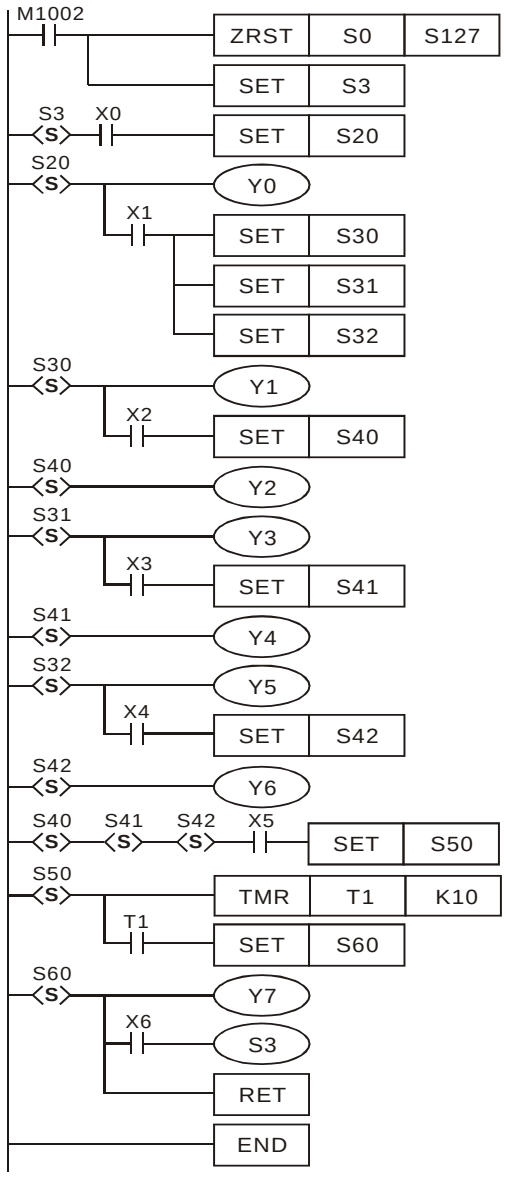
Одиночная последовательность с параллельным разветвлением и параллельной сборкой.

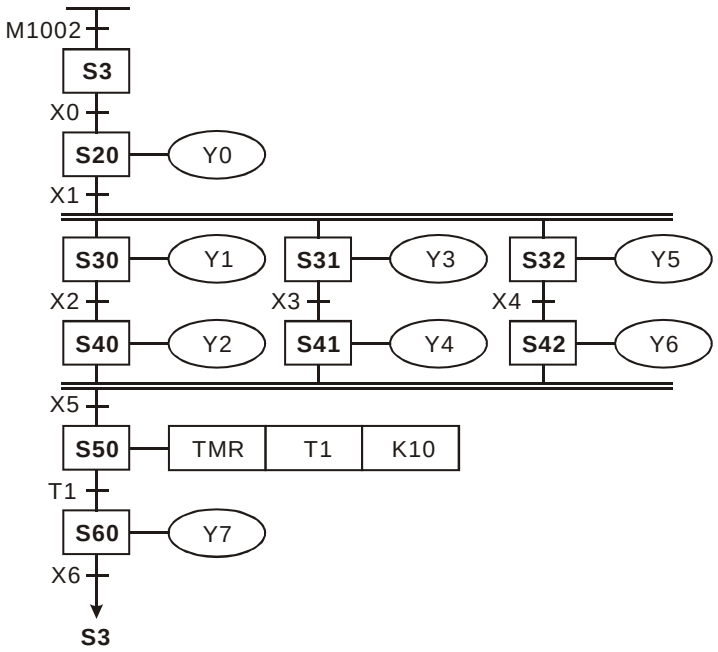
Ниже проиллюстрирована программа в виде ступенчатой и последовательной диаграмм, которая работает следующим образом:

При переводе контроллера в режим «Работа» замыкается реле M1002 и выдает запускающий импульс. Инструкция ZRST сбрасывает реле S0~S127, команда SET включает инициализирующее реле S3, которое запускает свою шаговую последовательность.

После реле S20 идет параллельное разветвление на три процесса: от реле S30, от S31 и от S32, которые запускаются одновременно. Каждое из данных реле (S30~S32) содержит свой процесс, заканчивающийся переходом к командной строке реле S50. Управление к реле S50 перейдет только тогда, когда замкнуться все контакты: S40, S41, S42 и X5.

Реле S50 в свою очередь отрабатывает таймер T1 и передает управление реле S60, которое возвращает процесс на начало к шагу S3. Таким образом, последовательность будет работать циклично до тех пор, пока контроллер находится в состоянии «Работа».





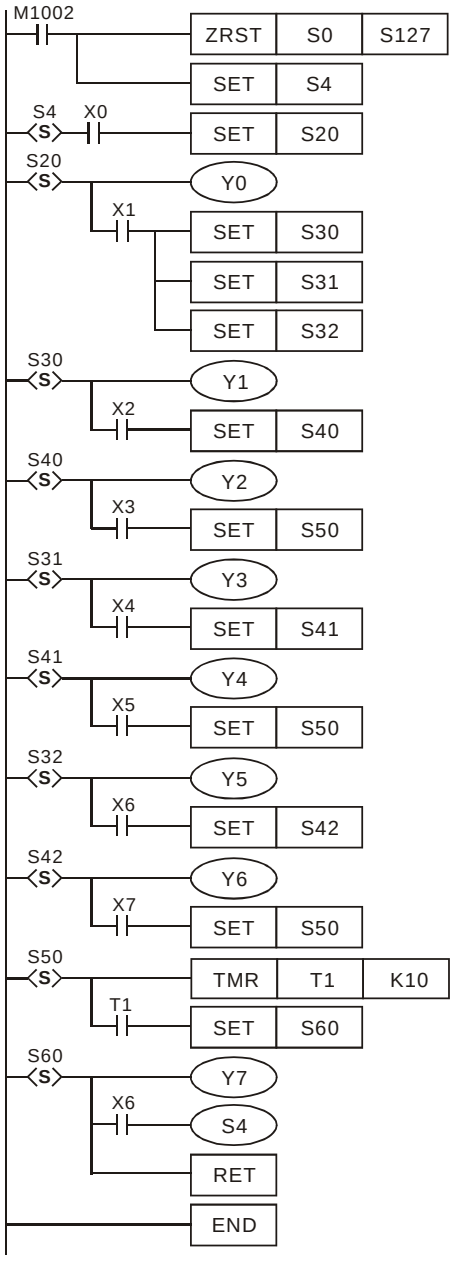
Пример 3
Одиночная последовательность с параллельным разветвлением и селективной сборкой.

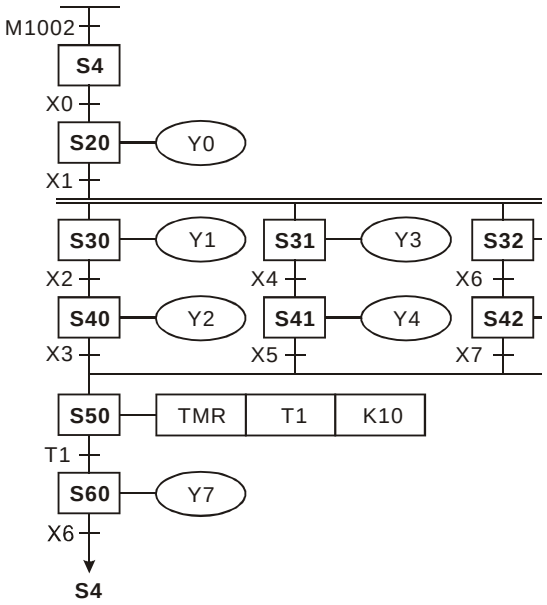
Ниже проиллюстрирована программа в виде ступенчатой и последовательной диаграмм, которая работает следующим образом:

При переводе контроллера в режим «Работа» замыкается реле M1002 и выдает запускающий импульс. Инструкция ZRST сбрасывает реле S0~S127, команда SET включает инициализирующее реле S4, которое запускает свою шаговую последовательность.

После реле S20 идет параллельное разветвление на три процесса: от реле S30, от S31 и от S32, которые запускаются одновременно и заканчиваются передачей управления к шагу S50. Передача управления произойдет от той ветви, в которой процесс закончился быстрее.

Реле S50 в свою очередь отрабатывает таймер T1 и передает управление реле S60, которое возвращает процесс на начало к шагу S3. Таким образом, последовательность будет работать циклично до тех пор, пока контроллер находится в состоянии «Работа».





Пример 4

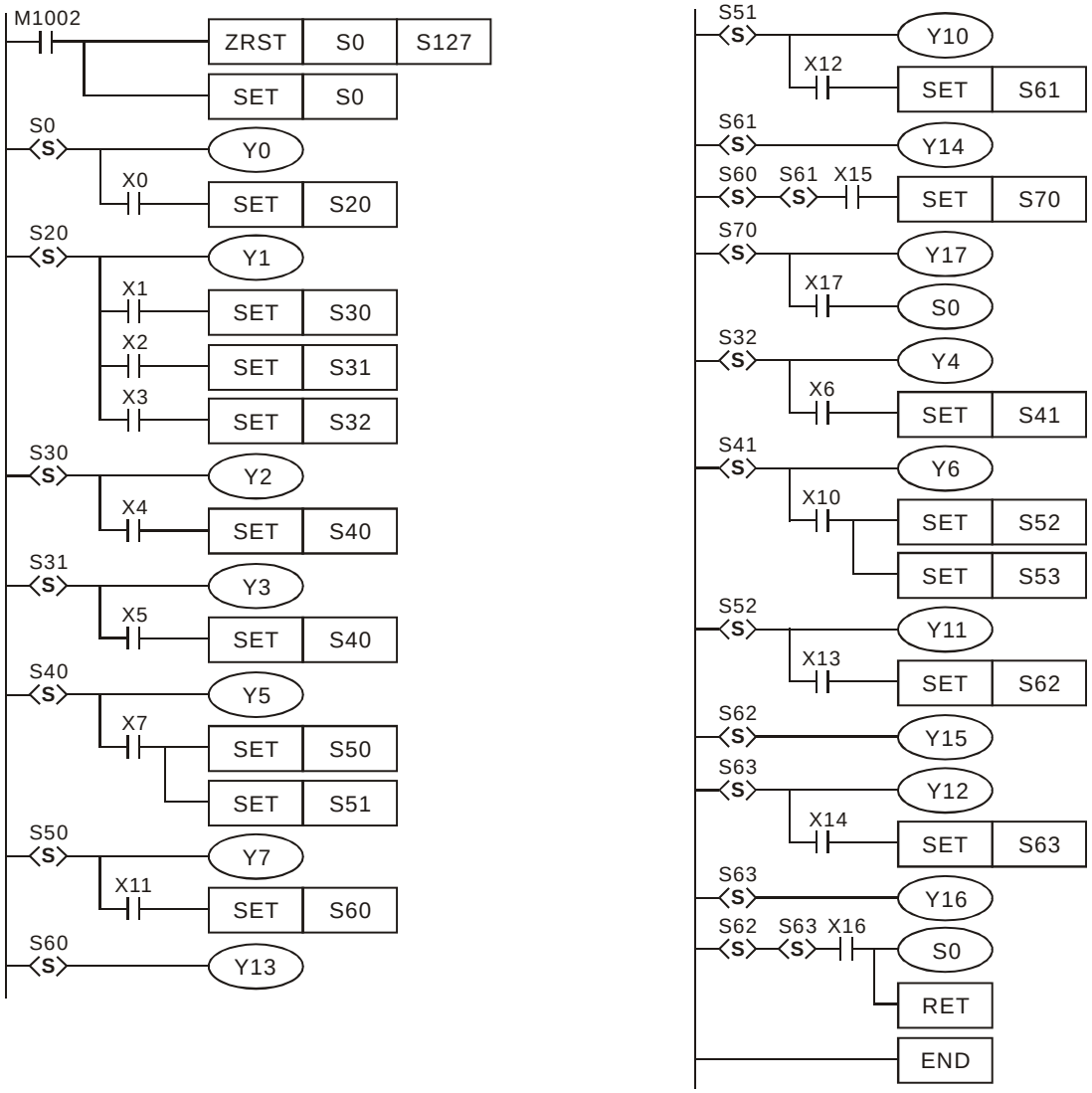
Одиночная комбинированная последовательность с параллельными разветвлениями и сборками, а также селективным разветвлением и сборкой.

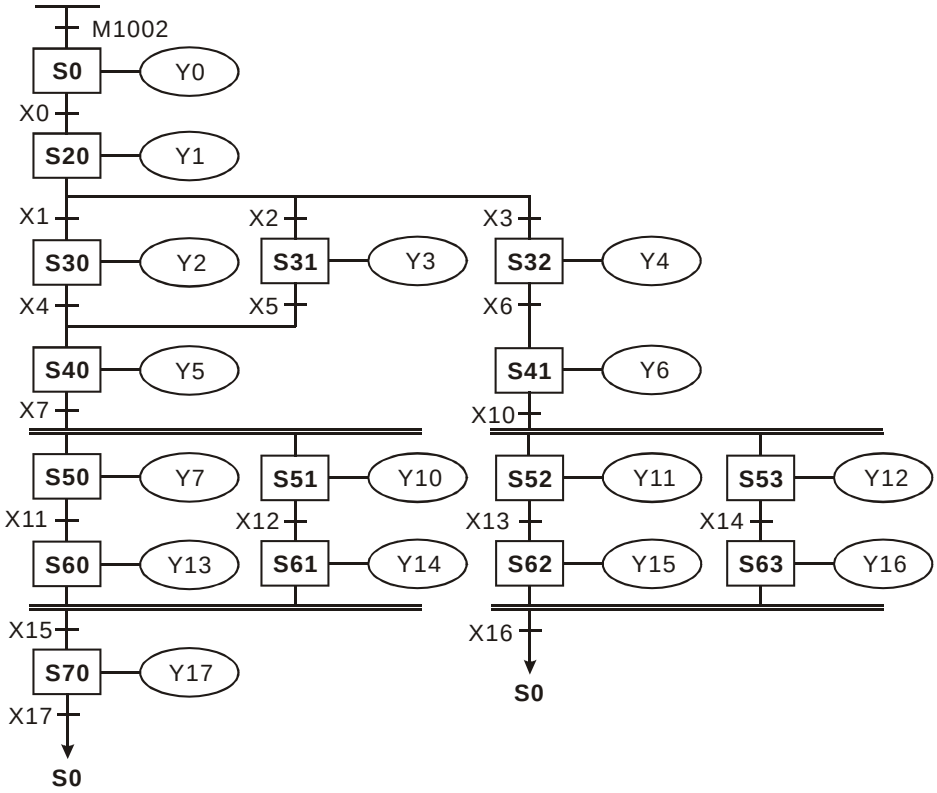
Ниже проиллюстрирована программа в виде ступенчатой и последовательной диаграмм, которая работает следующим образом:

После реле S20 идет селективное разветвление на три возможных варианта: при замыкании X1 управление перейдет к S30, при замыкании X2 к S31, а X7 к S32. Далее идет селективная сборка S30 и S31 на реле S40, а S32 перескакивает на S41.

Реле S40 запускает параллельное разветвление на S50 и S51, каждое из которых содержит свой процесс, заканчивающийся S60 и S61 соответственно и их дальнейшей параллельной сборкой и возвратом на S0.

Реле S32 запускает шаг S41, который осуществляет параллельное разветвление на S52 и S53, каждое из которых содержит свой процесс, заканчивающийся S62 и S63 соответственно и их дальнейшей параллельной сборкой и возвратом на S0.



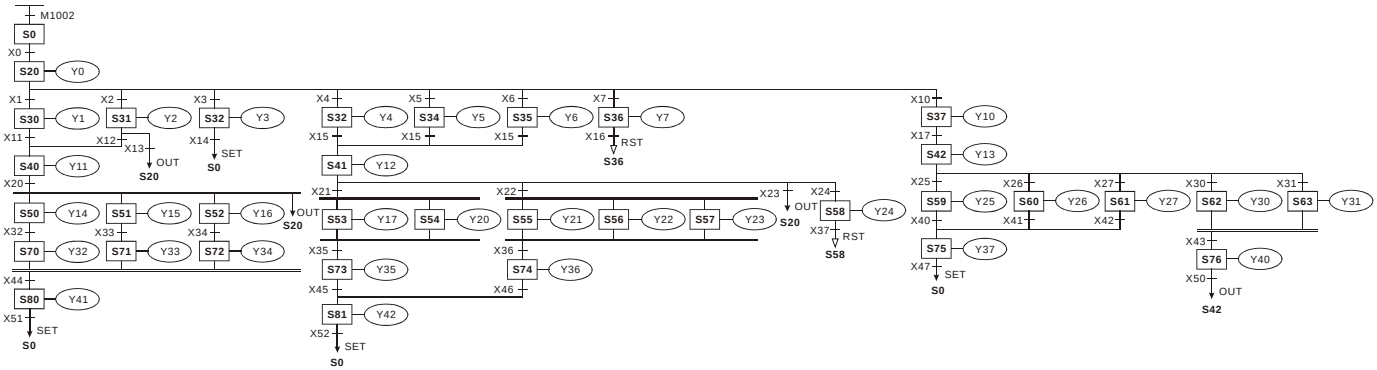


Пример 5

Одиночная сложная последовательность, демонстрирующая количественные ограничения применения шаговых последовательностей.

Максимальное количество шагов с ответвлениями – 8. В данном примере после S20 идет ровно 8 реле с ответвлениями: S30 ~ S37.

Максимальное количество последовательностей – 16. Они могут быть независимыми, а могут быть параллельными ветвями одной последовательности, но в совокупности должно быть не более 16 процессов. В данном примере после S40 четыре ветви, после S41 семь ветвей, и после S42 пять ветвей. Команды OUT, выходящие непосредственно из ветвящегося шага, также считаются процессом и должны учитываться при подсчете общего количества процессов. В данном примере это Y11, Y12 и Y13.



4.6 Инструкция IST

API	Символ	Параметры	Функция	Контроллеры		
60	IST	(S) (D1) (D2)	Установка параметров для объединенных шаговых последовательностей	ES/EX/SS	SA/SX/SC	EH/SV

Тип Пар.	Битовые				Словные операнды (регистры)										Программные шаги
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*													IST: 7 шагов
D_1				*												
D_2				*												

Импульсное выполнение								16-бит								32-бит							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Назначение:
Инструкция IST позволяет увязать в единую схему управления три шаговые последовательности и задать общие для них исходные параметры. Применяется в основном для удобного объединения на одном пульте управления трех отдельных шаговых процессов: автоматический режим работы технологической установки, наладочный (ручной) режим работы, а также возвращение в нулевую точку.

Задаваемые параметры инструкции:
S – адрес первого из восьми последовательных управляющих входов
D_1 – номер первого шагового реле в автоматическом режиме
D_2 – номер последнего шагового реле в автоматическом режиме

Параметр «S» определяет восемь управляющих входов для инструкции IST, каждому из которых жестко присваивается определенное действие. Если в данном параметре задан вход X10, то будет зарезервировано восемь входов X0 ~ X17 и они будет иметь следующие функции:

S = X10: Ручной режим
X11: Возвращение в ноль
X12: Режим исполнения одного шага
X13: Режим исполнения одного цикла
X14: Непрерывная работа
X15: Разрешение возврата в ноль
X16: Пуск
X17: Стоп

В качестве управляющих входов могут использоваться битовые операнды X, Y и M.

Параметр «D_1» определяет номер первого реле шаговой последовательности, которая отрабатывает автоматический режим.

Параметр «D_2» определяет номер последнего реле шаговой последовательности, которая отрабатывает автоматический режим.

В качестве параметров «D_1» и «D_2» могут использоваться только шаговые реле S. Диапазон для контроллеров SA/SX/SC/EH/EH2/SV S20 ~ S899, а для ES/EX/SS S20 ~ S127.
Причем $\mathbf{D_2 > D_1}$.

Ограничения:
Инструкция IST может применяться в программе только один раз.

Специальные регистры и реле:
M1040 ~ M1047, D1040 ~ D1047
(назначение далее по тексту)

Для своей работы инструкция IST задействует следующие операнды:

S0 – инициация ручного (наладочного) режима
S1 – инициация выхода в ноль
S2 – инициация автоматического режима

Указанным выше инициализирующим реле жестко присваиваются обозначенные функции. Следовательно, в программе они не могут использоваться для других целей. При необходимости инициализации шаговых последовательностей вне инструкции IST можно использовать оставшиеся реле S3 ~ S9.

Шаговые реле S10 ~ S19 жестко резервируются под процедуру выхода в ноль и не могут быть использованы для других целей. Если в ходе выполнения возвращения в ноль (S1=1) принудительно включить какое-нибудь из реле S10 ~ S19, то выхода в ноль не произойдет и может создаться аварийная ситуация.

При работе в автоматическом режиме (S2=1) при принудительном включении какого-нибудь из шаговых реле диапазона $\mathbf{D_1 \sim D_2}$ или реле M1043 произойдет остановка автоматического режима и может создаться аварийная ситуация.

Используются следующие специальные реле и регистры:

Операнд	Функция
M1040	Когда M1040=1 все операции запрещены. 1. В ручном режиме M1040 всегда включено 2. В режимах возврат в ноль/один цикл: во временной промежуток между нажатиями кнопок Пуск и Стоп, M1040=1 3. Пошаговый режим: M1040 будет включено до нажатия кнопки Пуск 4. Непрерывный режим: когда ПЛК будет переходить из «Стоп» в «Работа», M1040 останется включенным до нажатия кнопки «Пуск»
M1041	Инициация следующего шага. Используется системой в автоматическом режиме (S2=1) для перехода к следующему шагу. 1. Ручной режим/возврат в ноль: M1041=0 2. Режимы один шаг/цикл: M1041 включается в момент нажатия кнопки Пуск 3. Непрерывный режим: M1041 включается с нажатием кнопки Пуск, выключается с нажатием кнопки Стоп
M1042	Разрешение импульсного выхода. Импульсы начинают посылаться при нажатии кнопки Пуск.
M1043	Включается при завершении выхода в ноль.
M1044	Разрешение непрерывного режима работы. Должно быть включено в автоматическом режиме, чтобы S2 повторно запускалось.
M1045	Запрещение сброса всех выходов. Данный флаг используется при переводе технологической установки, не находящейся в нулевой точке, в следующие состояния: • из ручного режима (S0) в нулевую точку (S1) • из автоматического режима (S2) в ручной режим (S0) • из автоматического режима (S2) в нулевую точку (S1) При M1045=0, при включении шага Sn в диапазоне $D_1 \sim D_2$ будет включен соответствующий выход SET Yn, а затем и реле Sn и катушка Yn будут сброшены при переходе к следующему

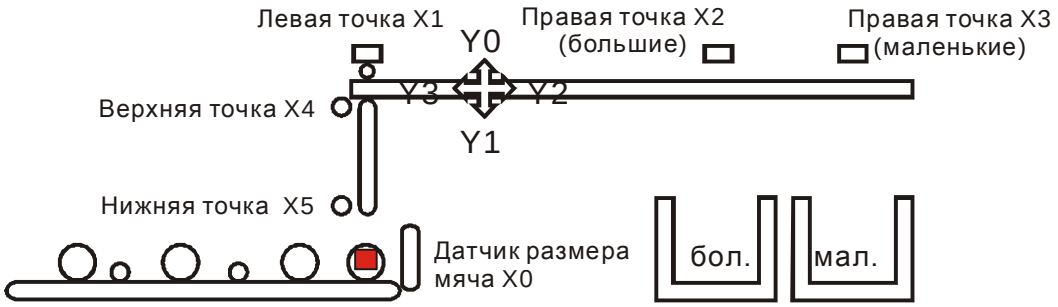
	шагу. При M1045=1, при включении шага в диапазоне D_1 ~ D_2 будет включен соответствующий выход SET Yn, а затем при переходе к следующему шагу катушка Yn останется включенной, а реле Sn сбросится. Если технологическая установка выполняет выход в ноль, то состояние M1045 роли не играет. Выход SET Y останется включенным, а реле S сбросится.
M1046	M1046=1, когда любое из шаговых реле включено. При M1047=1 включение любого шагового реле приведет к включению M1046. В регистры D1040 ~ D1047 будет записываться восемь номеров предыдущих шагов, которые были до включения текущего реле.
M1047	Разрешение мониторинга шаговой последовательности. При включении инструкции IST, реле M1047 включится автоматически и будет активно пока инструкция IST не выключится.
D1040 ~ D1047	Номера восьми предыдущих включавшихся шагов (по отношению к текущему шагу).

При выполнении инструкции IST следующие реле будет включаться/выключаться автоматически: M1040, M1041, M1042, M1047, S0, S1 и S2.

Пример использования инструкции IST

Рассмотрим вариант применения инструкции IST на примере работы руки робота, осуществляющей выбор и перемещение в соответствующую коробку маленьких и больших мячей. Рука робота должна работать с одного пульта управления в трех режимах: ручной, автоматический и осуществлять по команде выход в ноль.

Технологическая схема выглядит следующим образом:



Рука робота осуществляет следующие действия, для осуществления которых замыкаются соответствующие выходы контроллера:
Y0 – перемещение вверх
Y1 – перемещение вниз
Y2 – перемещение вправо
Y3 – перемещение влево
Y4=1 – захват мяча с конвейера
Y4=0 – отпускание мяча в коробке

Границы перемещения руки определяются соответствующими концевыми выключателями:
X1 – левый предел перемещения
X2 – правый предел перемещения, когда рука захватила большой мяч (внизу коробка для больших мячей)

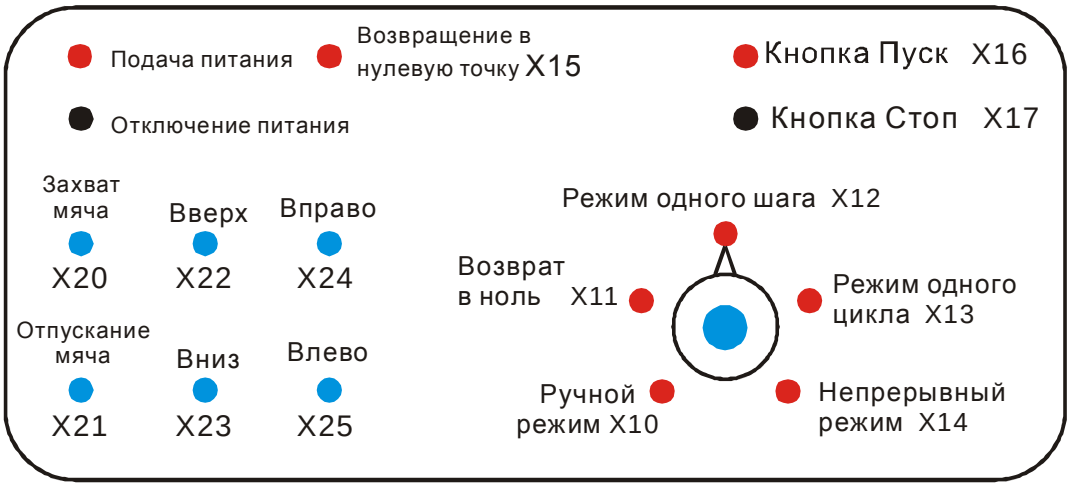
X3 – правый предел перемещения, когда рука захватила маленький мяч (внизу коробка для маленьких мячей)
X4 – верхний предел перемещения
X5 – нижний предел перемещения

Определение размеров мяча осуществляется датчиком с контактом X0: если мяч большой, то контакт замыкается, если маленький, то не замыкается.

Рука робота работает следующим образом:

1. Ручной режим. Каждое действие должно осуществляться с физическим нажатием соответствующей кнопки на пульте управления.
2. Выход в ноль. Должен осуществляться с нажатием соответствующей кнопки на пульте управления.
3. Автоматический режим. Работает в трех вариантах:
   - Один шаг. С каждым нажатием кнопки Пуск рука будет осуществлять одно действие.
   - Один цикл. При нажатии кнопки Пуск в момент нахождения руки робота в нулевой точке, рука выполнит один цикл и снова вернется в нулевую точку. Если нажать кнопку Стоп в момент выполнения цикла, рука остановится, а при повторном нажатии кнопки Пуск продолжит выполнение цикла с текущего места до завершения в нулевой точке.
   - Непрерывная работа. При нажатии кнопки Пуск в момент нахождения руки робота в нулевой точке, рука будет работать без перерыва цикл за циклом. Для остановки нужно нажать кнопку Стоп, рука отработает текущий цикл до конца и остановится в нулевой точке.

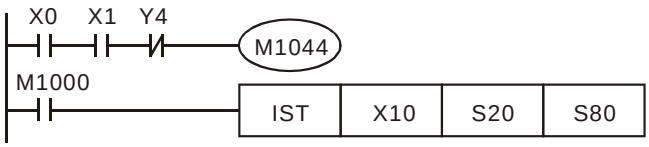
Для управления рукой робота предназначена следующая панель оператора:



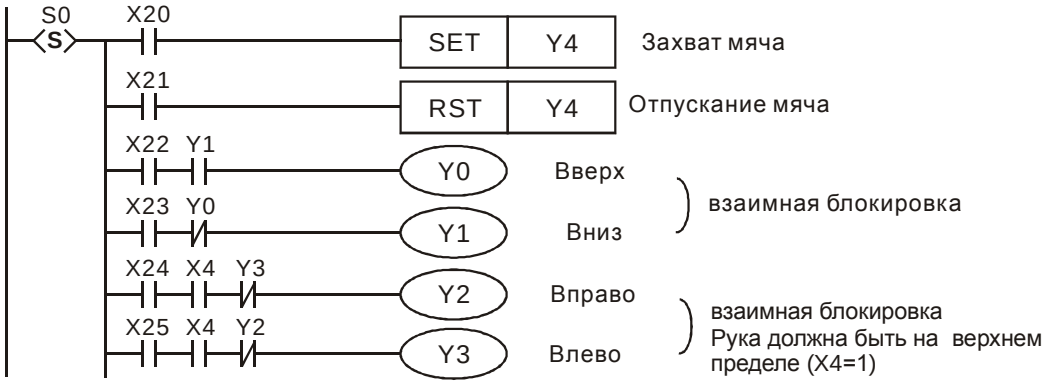
Контакты X20 – X 25 используются в ручном режиме.
Контакты X10 – X14 заводятся на позиционный регулятор и служат для выбора режима
Контакт X16 заводится на кнопку Пуск, а X17 на копку Стоп.

Выбор номера первого управляющего контакта (входа X) выбирается в параметре инструкции IST «S». В данном примере X10, следовательно, выделяются последовательно 8 входов X10 – X17.

Начальный участок программы:



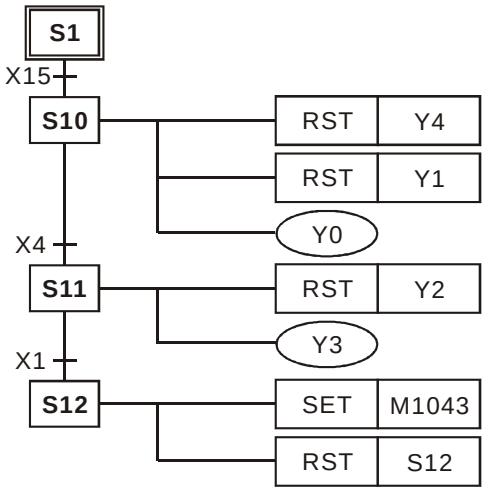
Участок программы, реализующий ручной режим (на пульте X10=1 (S0=1))
Каждое действие руки робота осуществляется путем нажатия соответствующей кнопки на пульте управления.



Участок программы, реализующий возврат в нулевую точку (на пульте регулятор X11=1 (S1=1), кнопка X15=1)

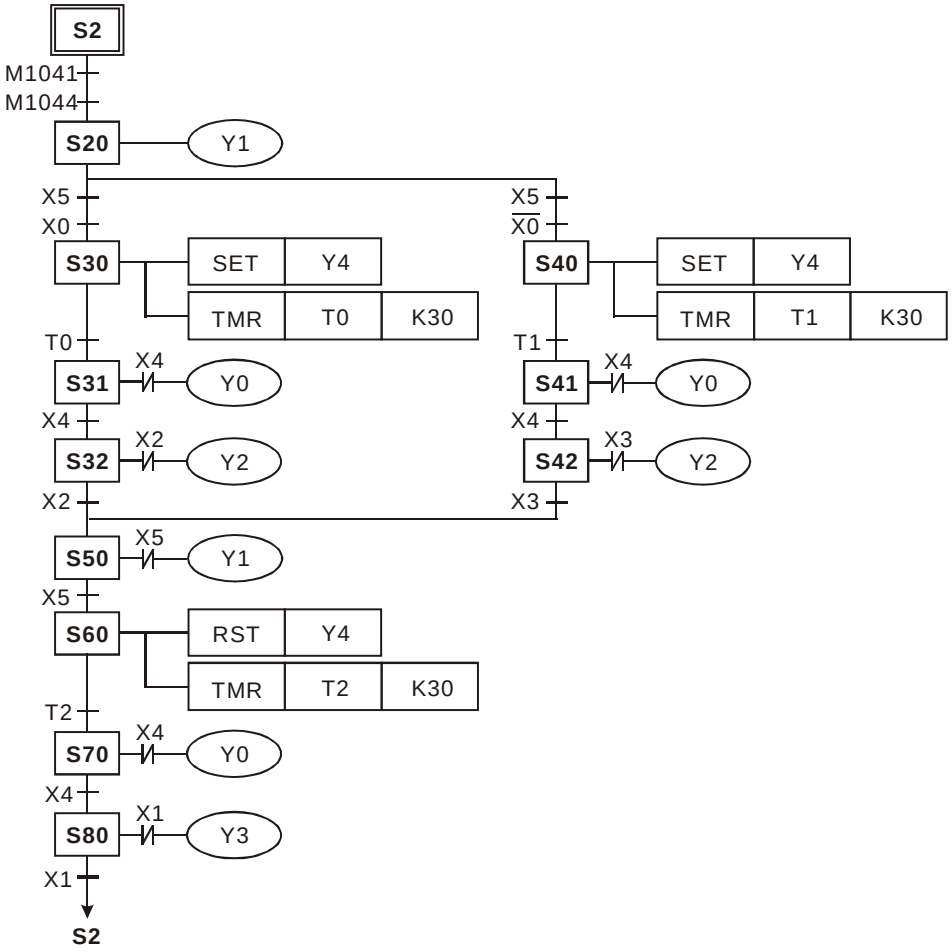
Шаговая ступенчатая диаграмма

Последовательная функциональная диаграмма (SFC)

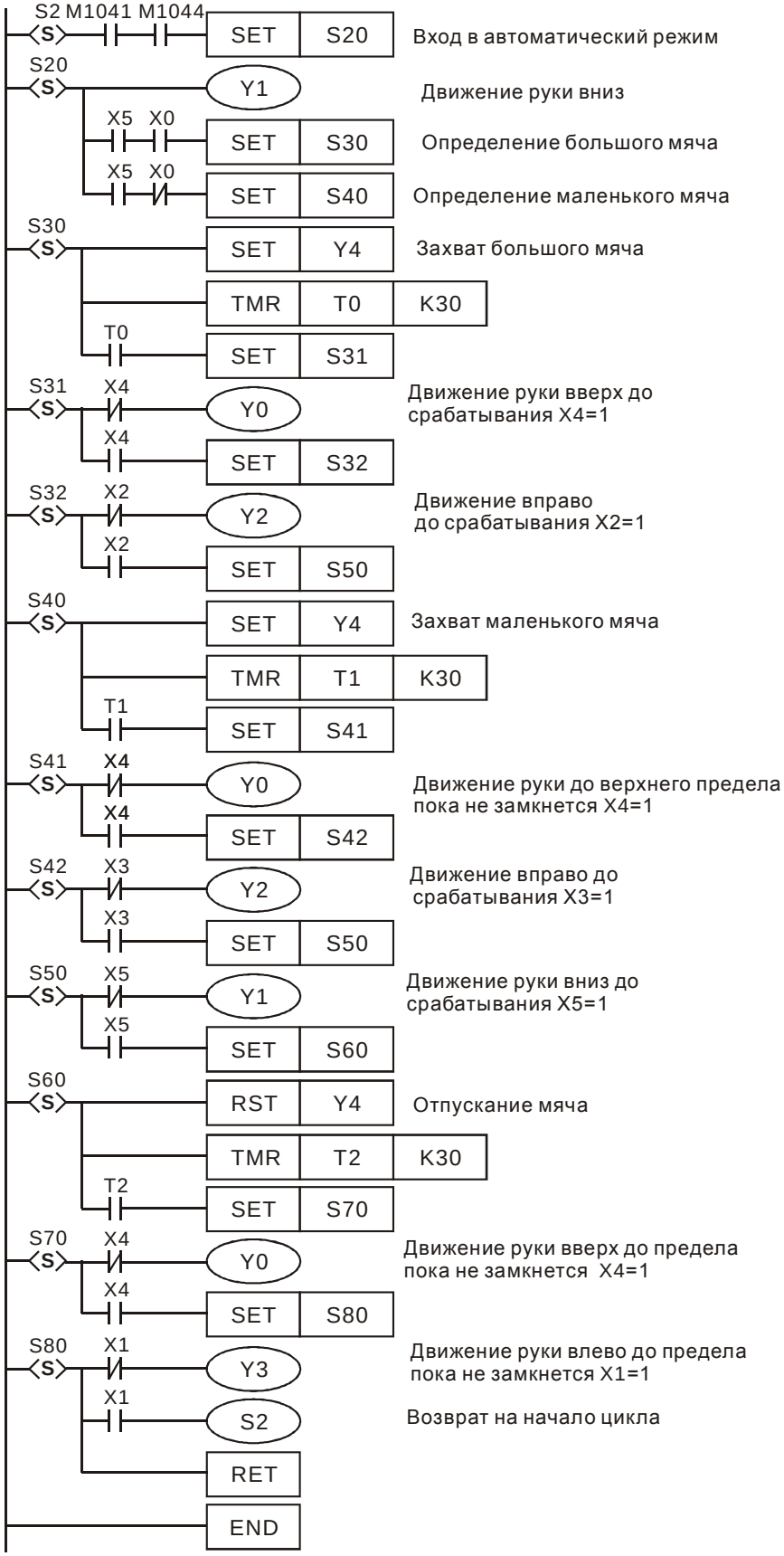


Участок программы, реализующий непрерывный автоматический режим (S2=1)
На пульте X14=1 и M1044=1 – непрерывная работа, X12=1 – режим одного шага, X13 – режим одного цикла.

Последовательная функциональная диаграмма (SFC)



Шаговая ступенчатая диаграмма



ГЛАВА 5

Категории и использование прикладных инструкций контроллеров Delta DVP

5.1 Общая компоновка прикладных инструкций

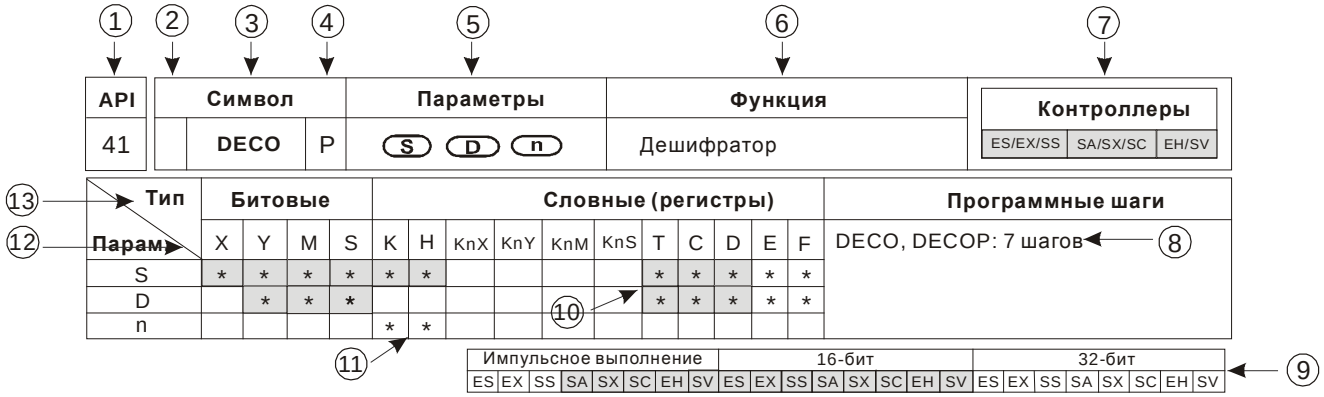
Прикладная инструкция представляет собой готовый программный модуль, предназначенный для решения какой-либо конкретной специальной задачи. Например, инструкция ADD является готовым инструментом для осуществления операции сложения двух чисел.
Прикладные инструкции позволяют просто и наглядно решать наиболее распространенные задачи, встречающиеся при написании рабочей программы контроллера. Таким образом, наличие прикладных инструкций освобождают пользователя от самостоятельного написания соответствующих программных блоков, что весьма существенно экономит время на разработку.

Контроллеры семейства Delta DVP предоставляют пользователю большое количество прикладных инструкций, охватывающих все типовые задачи, встречающиеся в ходе разработки рабочей программы контроллера. В общей сложности существует уже порядка 246 прикладных инструкций. В каждой модели контроллера реализован свой набор инструкций.

Прикладная инструкция обозначается как «API» и имеет свой номер, а также мнемонический символ (имя) для удобства запоминания. Каждая инструкция состоит из программно-вычислительного кода (собственно инструкции) и параметров инструкции. Код занимает 1 программный шаг, а каждый из параметров 2 (16 бит) или 4 (32 бит) программных шага.

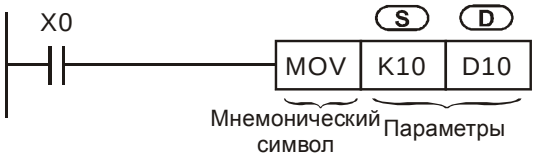
Под параметрами инструкции понимают исходные данные и результат, необходимые для работы инструкции. Например, для инструкции сложения ADD требуются два исходных числа и место, куда записать результат. Т.е. в данном случае инструкция имеет три обязательных параметра, которые должны быть заданы при написании программы.
Значения параметров могут задаваться константами К/Н и различными операндами контроллера – регистрами данных, счетчиками, таймерами, битовыми устройствами и т.п.

В настоящем Руководстве по программированию все прикладные инструкции имеют описание, выполненное в соответствии с типовой компоновкой, приведенной ниже.



① Номер прикладной инструкции.
② Указатель возможности работы в формате 32 бит. Если стоит буква «D», то данную инструкцию можно использовать в формате 32 бит, если нет, то только в формате 16 бит.
③ Мнемонический символ (имя) прикладной инструкции.
④ Указатель возможности работы инструкции в импульсном режиме. Если стоит буква «P», то данная инструкция может работать в импульсном режиме, если нет, то не может.
⑤ Параметры инструкции (задаются пользователем).
⑥ Функция инструкции. Кратко отражается задача, решаемая данной инструкцией.
⑦ Типы контроллеров, поддерживающих данную инструкцию. ES включает в себя также ES/EX/SS, SA включает SA/SX/SC, EH включает EH/EH2/SV.
⑧ Количество шагов, требующееся для выполнения инструкции в формате 16 бит, 32 бит и импульсном режиме.
⑨ Типы контроллеров, поддерживающих данную инструкцию в формате 16 бит, 32 бит и импульсном режиме. Соответствующий тип контроллера выделяется серым цветом.
⑩ Ячейки, помеченные значком «*» и выделенные серым цветом, означают, что соответствующий операнд может использоваться с индексными указателями E и F.
⑪ Ячейки, помеченные значком «*», означают, что данный операнд применим для задания исходных данных соответствующему параметру инструкции.
⑫ Название операнда контроллера.
⑬ Тип операнда контроллера (битовый или словный).

При написании программы ввод прикладных инструкций осуществляется с помощью Мастера или вручную по символу инструкции.



Для каждой инструкции необходимо задать параметры, которые имеют следующие обозначения:

S	Параметр – источник данных. Если источников несколько, то они нумеруются последовательно нижним индексом: **S_1, S_2**, …
D	Параметр – получатель результата. Если получателей результата несколько, то они нумеруются последовательно нижним индексом: **D_1, D_2**, …
Если параметр задается только константой K/H или регистром, то он обозначается как **m**, **m_1**, **m_2**, **n**, **n_1**, **n_2**, ….	

В приведенном выше примере используется инструкция MOV, которая данные из источника (S) перемещает к получателю (D).

Значения параметров инструкции могут иметь 16-ти или 32-х битный формат. Значение в 16 бит занимает 1 регистр памяти, а значение в 32 бит занимает 2 последовательных регистра памяти. Когда необходимо, чтобы инструкция выполнялась в формате 32 бит, перед ее символом добавляется буква «D».

Инструкция MOV в формате 16 бит

Когда X0 = 1, K10 будет переслано в D10.

Инструкция DMOV в формате 32 бит

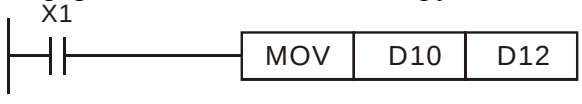
Когда X1 = 1, содержимое регистров (D11, D10) будет переслано в регистры (D21, D20).

Многие инструкции могут выполняться как в непрерывном, так и в импульсном режиме. Благодаря использованию инструкций в импульсном режиме, можно сократить общее время цикла программы, так как в импульсном режиме инструкция выполняется не постоянно. Когда необходимо, чтобы инструкция выполнялась в импульсном режиме, после ее символа добавляется буква «P». Для некоторых инструкций, например INC или DEC, применение в импульсном режиме является предпочтительным.

Импульсное выполнение инструкции

Когда X0 замыкается, инструкция MOVP выполниться один раз в течение текущего скана и больше выполняться не будет до начала следующего скана.

Непрерывное выполнение инструкции



Когда X1 замыкается, инструкция MOV будет выполняться постоянно в течение скана.

На обоих рисунках, когда X0 и X1 выключены, инструкция выполняться не будет и данные в регистре D12 меняться не будут.

Обозначение параметров в прикладных инструкциях

1. Битовые параметры X, Y, M и S в прикладных инструкциях можно объединять в слова, храня состояние битовых параметров в регистрах данных 16 бит в виде KnX, KnY, KnM и KnS.
2. Словные параметры: регистры D, таймеры T, счетчики C и индексные регистры E, F в прикладных инструкциях обозначаются одноименными операндами без каких-либо изменений.
3. В прикладных инструкциях параметр, как правило, 16 бит и занимает стандартный регистр D. Если параметр в прикладной инструкции используется в формате 32 бит, то он займет 2 последовательных регистра D.
4. Если в прикладной инструкции, работающей только в формате 32 бит, использовать регистр данных 16 бит, например D0, то он все равно займет 2 последовательных регистра D0 и D1. В данном случае D1 – это старшие 16 бит, а D0 – младшие 16 бит. Тоже самое касается и использования 16-ти битных таймеров и счетчиков C0 ~ C199 в 32-х битных инструкциях.
5. Использование 32-х разрядных счетчиков C200 ~ C255 возможно только с 32-х разрядными инструкциями, в т.ч. и при использовании данных счетчиков в качестве регистров данных.

Формат параметров в прикладных инструкциях

1. Параметры X, Y, M и S могут включать или выключать только одну точку, поэтому именуются битовыми параметрами.
2. Параметры D, C, и T и индексы E, F имеют формат словных регистров 16 или 32 бит, поэтому именуются словными параметрами.
3. Перед битовыми параметрами X, Y, M и S можно поставить коэффициент Kn, при помощи которого битовые параметры можно последовательно объединять в слова (регистры) для дальнейшей обработки. Множитель коэффициента «n=1» равен 4 битам. Для 16-ти битных инструкций «n» может принимать значения K1 ~ K4, а для 32-х битных K1 ~ K8. Например: K2M0 соответствует 8 битов M0 ~ M7.

Когда X0 = 1, содержимое M0 ~ M7 будет переслано в биты 0 ~ 7 регистра D10, а биты 8 ~ 15 будет равны 0.

Битовые параметры, объединенные в слова, могут формировать число, предельное значение которого определяется получившейся разрядностью, что поясняется в таблицах ниже:

16-ти битная инструкция	
Максимальное значение: K-32,768 ~ K32,767	
Предельные значения при K1 ~ K4	
K1 (4 бит)	0 ~ 15
K2 (8 бит)	0 ~ 255
K3 (12 бит)	0 ~ 4095
K4 (16 бит)	-32768 ~ +32767

32-х битная инструкция	
Максимальное значение: K-2,147,483,648 ~ K2,147,483,647	
Предельные значения при K1 ~ K8	
K1 (4 бит)	0 ~ 15
K2 (8 бит)	0 ~ 255
K3 (12 бит)	0 ~ 4095
K4 (16 бит)	0 ~ 65535
K5 (20 бит)	0 ~ 1 048 575
K6 (24 бит)	0 ~ 167 772 165
K7 (28 бит)	0 ~ 268 435 455
K8 (32 бит)	-2 147 483 648 ~ +2 147 483 647

Флаги

При обработке некоторых прикладных инструкций контроллер автоматически включает или отключает различные флаги (специальные реле). Используемый флаг показывает определенное состояние выполнения инструкции или программы в целом. Флаг каждый раз включается или отключается, если в программе активизируется соответствующая инструкция. По своему назначению флаги подразделяются на общие, флаги ошибок и флаги расширения функций.

Общие флаги:

M1020 – флаг нуля. Включается, если результат сложения или вычитания равен нулю

M1021 – флаг заимствования (Borrow). Включается, если результат вычитания меньше самого малого значения

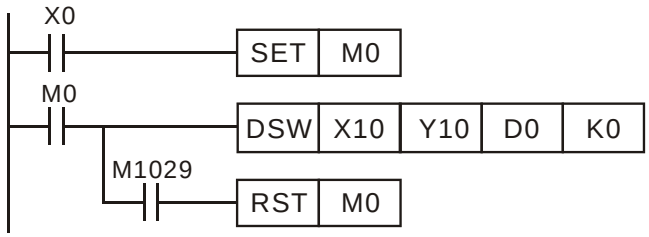
M1022 – флаг переноса (Carry). Включается при передаче значения числа, при суммировании или при передаче данных, при выполнении инструкции сдвига

M1029 – флаг завершения выполнения инструкции

Общие флаги используются во многих инструкциях, что отражено в соответствующих описаниях.

Пример
Использование реле M1029 совместно с инструкцией DSW (цифровой переключатель).

При срабатывании входного условия, инструкция DSW осуществляет цикличный опрос 4-х входов с частотой 0,1 сек с целью чтения выставленного значения цифрового переключателя. Если в ходе выполнения инструкции входное условие сбросится, то выполнение остановится, не дойдя до конца. Чтобы избежать этого, можно использовать схему, приведенную ниже:



Когда X0 = 1, DSW активируется.
Когда X0 = 0, M0 выключится только, когда инструкция DSW закончит цикл и M1029 = 1.

Флаги ошибок:

Ошибки при выполнении инструкций могут возникать при неправильной комбинации инструкций, или когда какой-либо из операндов выходит за допустимый диапазон. В данных случаях активируются нижеприведенные реле и регистры.

Операнд	Комментарии
M1067 D1067 D1069	M1067 включается при появлении ошибки. В регистре D1067 отображается код ошибки, а в D1069 шаг программы, в котором возникла ошибка. При появлении новых ошибок, содержимое D1067 и D1069 будет автоматически обновляться. Реле M1067 выключится после устранения ошибки.
M1068 D1068	M1068 включается при появлении ошибки. В регистре D1068 отображается шаг программы, где возникла ошибка. Появление новых ошибок не будет влиять на содержимое D1068 до тех пор, пока реле M1068 не будет принудительно сброшено командой RST.

Флаги расширения функций:

Некоторые инструкции могут иметь специальные флаги, включающие или отключающие дополнительные возможности данных функций. Например, у инструкции RS есть реле M1161, которое переключает между режимами 8 или 16 бит.

Ограничения при использовании прикладных инструкций

Некоторые прикладные инструкции могут использоваться в программе ограниченное количество раз. Ниже приводится описание данных инструкций. В таблицах, контроллеры ES также подразумевают EX и SS, SA включает SX и SC, EH включает EH2 и SV.

1. Инструкции, допускающие только однократное использование в программе

API 58 PWM (ES)	API 60 IST (ES/SA/EH)
API 74 SEGL (ES)	API 155 DABSR (SC/EH)

2. Инструкции, допускающие двукратное использование в программе

API 57 PLSY (ES)	API 59 PLSR (ES)
API 74 SEGL (EH)	API 77 PR (SA/EH)

3. Инструкции, которые могут быть использованы в программе не более 4-х раз

API 169 HOUR (SA)

4. Инструкции, которые могут быть использованы в программе не более 8-ми раз

API 64 TTMR (SA series MPU)

5. В контроллерах ES инструкции API 53 DHSCS и API 54 DHSCR совместно могут использоваться в программе не более 4-х раз.

6. В контроллерах SA инструкции API 53 DHSCS, API 54 DHSCR и API 55 DHSZ совместно могут использоваться в программе не более 6-ти раз.

Далее приводится список инструкций, по которым нет ограничений по количеству раз использования в программе, но есть ограничение на количество раз одновременного исполнения одной и той же инструкции.

1. Инструкции, допускающие только однократное одновременное исполнение в программе:

API 52 MTR (SA/EH), API 56 SPD (ES/SA/EH), API 69 SORT (SA/EH), API 70 TKY (SA/EH), API 71 HKY (SA/EH), API 72 DSW (SA), API 74 SEGL (SA), API 75 ARWS, API 80 RS (ES/SA/EH), API 100 MODRD (ES/SA/EH), API 101 MODWR (ES/SA/EH), API 102 FWD (ES/SA/EH), API 103 REV (ES/SA/EH), API 104 STOP (ES/SA/EH), API 105 RDST (ES/SA/EH), API 106 RSTEF (ES/SA/EH), API 150 MODRW (ES/SA/EH), API 151 PWD (EH).

2. Инструкции, допускающие только двукратное одновременное исполнение в программе:

API 57 PLSY (EH), API 58 PWM (SA/EH), API 59 PLSR (SA/EH), API 72 DSW (EH).

3. Инструкции, которые могут быть одновременно запущены в программе не более 4-х раз:

API 57 PLSY (EH2/SV), API 58 PWM (EH2/SV), API 169 HOUR (EH).

4. Инструкции, которые могут быть одновременно запущены в программе не более 8-ми раз:

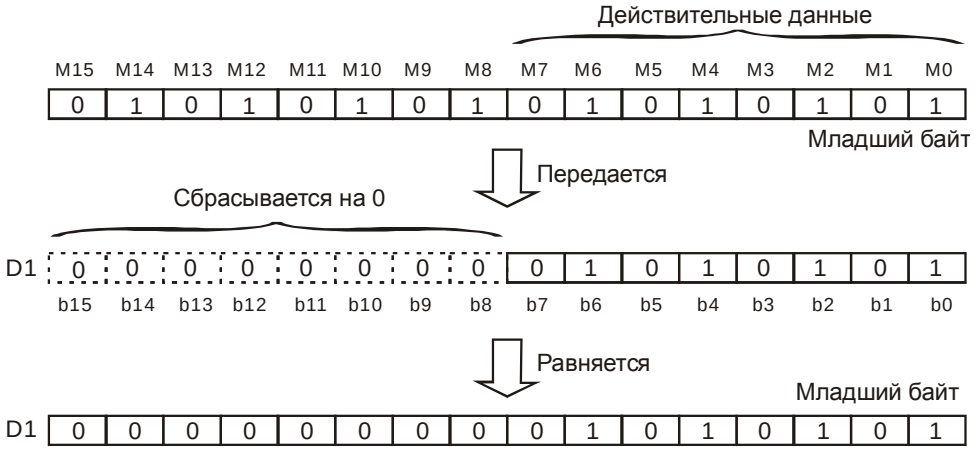
API 64 TTMR (EH).

5. В контроллерах SA нет ограничений на количество раз использования инструкций высокоскоростного импульсного выхода PLSY, PWM и PLSR, однако в каждом скане будет выполняться только одна из них.
6. В контроллерах EH нет ограничений на количество раз использования инструкций аппаратного высокоскоростного счетчика DHSCS, DHSCR и DHSZ, однако при одновременной активации трех и более инструкций необходимо учитывать, что каждая инструкций задействует регистр памяти и в совокупности их не должно быть более 8. Если будет задействовано более 8 регистров, контроллер выполнит те инструкции, которые стоят в программе раньше. Инструкции DHSCS и DHSCR задействуют по 1 регистру каждая, а инструкция DHSZ задействует 2 регистра.

5.2 Обработка числовых значений

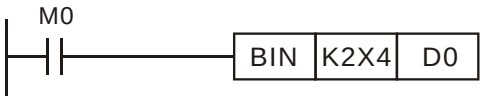
Параметры прикладных инструкций бывают битовыми, т.е. которые принимают два состояния – включен или выключен, а также словными, предназначенными исключительно для хранения числовой информации. К первой категории относятся параметры X, Y, M и S, ко второй D, T, C, E и F.

Битовые параметры могут быть объединены в слово для дальнейшей обработки с использованием коэффициента Kn, подробно рассмотренного в предыдущем параграфе.
Перенос данных осуществляется следующим образом (для примера в предыдущем параграфе, K2M0):



Не используемые биты регистра заполняются нулями.

Пример переноса битовых данных в один регистр:



Данные, получающиеся пре переносе битовых параметров X4 ~ X13, имеют изначально формата BCD (двоично-десятичный). Для преобразования в двоичный формат используется инструкция BIN, и в регистр D0 данные уже попадут в двоичном формате.

При переносе битовых параметров в словный регистр в качестве начального битового параметра можно использовать любой адрес. Однако, во избежание ошибок лучше начинать с нулевых адресов для X и Y – X0, X10, X20 (восьмеричная система) и т.д., а для M и S первый адрес должен быть кратен восьми (хотя начинать с нулевых адресов и здесь будет лучшим вариантом). Данные рекомендации продемонстрированы ниже в таблице:

K1X0	K1X4	K1X10	K1X14…
K2Y0	K2Y10	K2Y20	Y2X30…
K3M0	K3M12	K3M24	K3M36…
K4S0	K4S16	K4S32	K4S48…

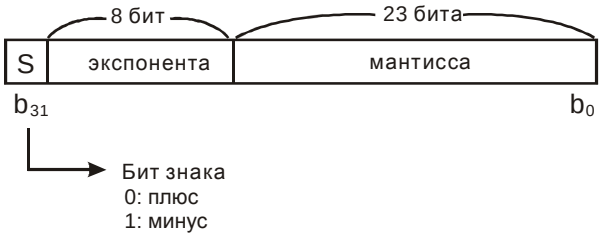
Также, если использовать операцию K4Y0 в 32-х битной инструкции, старшие 16 бит останутся не заполненными, поэтому лучше использовать операцию K8Y0.

Контроллеры Delta DVP осуществляют расчеты в двоичном формате целого числа. Следовательно, при операциях с числами дробная часть отбрасывается. Например, 40 делить на 3 получает 13 целых и 3 десятых, которые будут отброшены. При извлечении квадратного корня дробная часть также отбрасывается. Если необходимо осуществлять расчеты с точностью до знаков после запятой, то необходимо использовать специальные инструкции, перечисленные ниже:

API 49 (FLT)	API 110 (D ECMP)	API 111 (D EZCP)	API 112 (D MOVR)
API 116 (D RAD)	API 117 (D DEG)	API 118 (D EBCD)	API 119 (D EBIN)
API 120 (D EADD)	API 121 (D ESUB)	API 122 (D EMUL)	API 123 (D EDIV)
API 124 (D EXP)	API 125 (D LN)	API 126 (D LOG)	API 127 (D ESQR)
API 128 (D POW)	API 129 (INT)	API 130 (D SIN)	API 131 (D COS)
API 132 (D TAN)	API 133 (D ASIN)	API 134 (D ACOS)	API 135 (D ATAN)
API 136 (D SINH)	API 137 (D COSH)	API 138 (D TANH)	API 172 (D ADDR)
API 173 (D SUBR)	API 174 (D MULR)	API 175 (D DIVR)	

Инструкции в таблице используют операции с плавающей точкой, позволяющие осуществлять не только операции с дробными числами, но также с очень большими или малыми числами, когда объема стандартного регистра уже не хватает.

В контроллерах Delta DVP операции с плавающей точкой осуществляются в соответствии со стандартом IEEE754 по следующей схеме:



Мантисса – часть числа с плавающей запятой, содержащая числа после запятой.
Экспонента – показатель степени числа, в которую нужно возвести основание системы счисления, чтобы получить данное число.

Для двоичной системы пересчет числа в число с плавающей точкой осуществляется по следующей формуле:

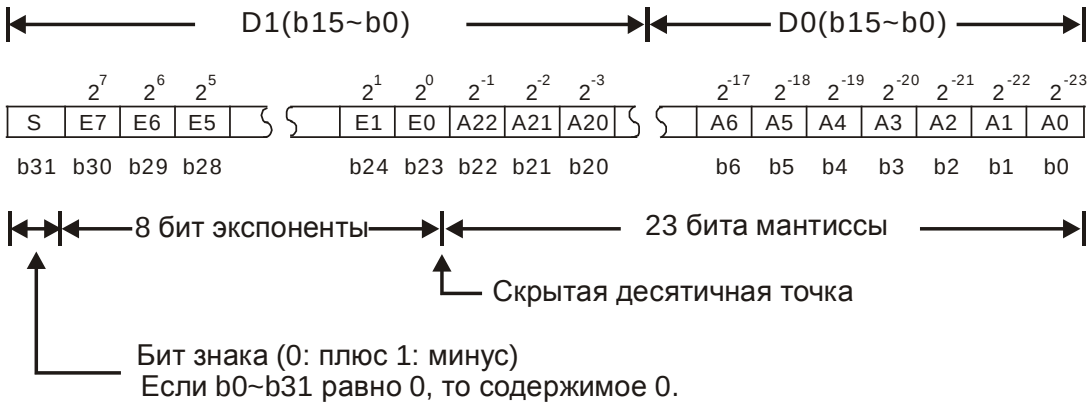
$$(-1)^S \times 2^{E-B} \times 1.M$$

Где S – знак числа, 2 – основание системы счисления (двоичная), M – мантисса, E – экспонента, B – константа равная числу 127.

Получается следующий диапазон 32-х разрядного числа с плавающей точкой:

$\pm 2^{-126} \sim \pm 2^{+128}$, или в десятичном формате $\pm 1.1755 \times 10^{-38} \sim \pm 3.4028 \times 10^{+38}$

Для хранения числа с плавающей точкой используются два последовательных регистра, например (D1, D0). Распределение битов на примере данных двух регистров показано ниже:



В качестве примера переведем десятичное число +23,0 в 32-х разрядное число с плавающей точкой.

Шаг 1.
Преобразуем десятичное число 23 в двоичное: 23,0 = 10111

Шаг 2.
Записываем число 10111 в научной форме (нормализуем):
$10111 = 1.0111 \times 2^4$, где 0111 мантисса, а 4 экспонента

Шаг 3.
Получаем экспоненту для числа с плавающей точкой:
$\because E - B = 4 \rightarrow E - 127 = 4 \ \therefore E = 131 = 10000011_2$

Шаг 4.
Соединяем вместе бит знака, экспоненту и мантиссу:

$$0\ 100\mathbf{0001}1\ 011\mathbf{1000}0000\mathbf{0000}0000\mathbf{0000}_2 = 41B80000_{16}$$

Если нужно было бы преобразовать отрицательное число -23,0, то для этого необходимо произвести все те же действия, но в бит знака записать «1».

Двоичный формат записи числа с плавающей точкой не удобен для восприятия, поэтому его можно конвертировать в формат десятичного числа с плавающей точкой. При этом необходимо помнить, что контроллер будет производить операции с десятичным числом с плавающей точкой в формате двоичного числа с плавающей точкой.

Десятичное число с плавающей точкой занимает два последовательных регистра, например (D1, D0), и будет иметь следующий вид:

Десятичное число с плавающей точкой = [константа D0] × $10^{[\text{экспонента D1}]}$

Константа D0 = ±1000 ~ ±9999

Экспонента D1 = - 41 ~ +35

Диапазон десятичного числа с плавающей точкой:

$\pm 1175 \times 10^{-41} \sim \pm 3402 \times 10^{+35}$.

Константа 100 не существует в регистре D0, так как 100 представляется как 1000×10^{-1}.

Десятичное число с плавающей точкой может использоваться в следующих инструкциях:

- D EBCD: Конвертация двоичного числа с плавающей точкой в десятичное число с плавающей точкой
- D EBIN: Конвертация десятичного числа с плавающей точкой в двоичное число с плавающей точкой

В операциях с плавающей точкой используются следующие флаги:

- Флаг нуля: M1020 = 1, если результат операции равен «0».
- Флаг заимствования: M1021 = 1, если результат операции превосходит минимальную единицу.
- Флаг переноса: M1022 = 1, если абсолютное значение результата операции выходит за допустимый диапазон.

5.3 Правила работы с индексными регистрами E и F

Индексные регистры служат для динамического изменения адреса какого-либо операнда путем прибавления значения индексного регистра к значению операнда.

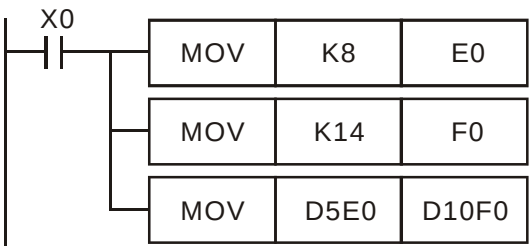
Индексные регистры имеют разрядность 16 бит. Если необходимо использовать индекс с разрядностью 32 бит, то индекс E и индекс F используются совместно. В индексе E будут хранится младшие 16 бит, а в индексе F будут хранится старшие 16 бит. Само 32-х разрядное значение записывается в индекс E, который при этом перекроет индекс F с таким же номером. В данном случае соответствующий индекс F будет уже не доступен. Комбинации 32-х разрядных индексных регистров будут следующие:

(F0, E0)
(F1, E1)
(F2, E2)
.....
(F7, E7)

Для обнуления 32-х разрядного регистра необходимо использовать команду DMOV K0 применительно к индексу E, при этом автоматически обнулиться и индекс F с тем же номером.

Пример использования индексных регистров для изменения адресации регистров D.

Когда X0 замкнется в регистр E0 запишется значение "8", в регистр F0 запишется значение "14". Далее произойдет суммирование адресов:
D5E0=D5 + E0= 5 + 8=13=D13
D10F0=D10 + F0= 10 + 14=24=D24
Таким образом, при текущих значениях индексов E и F произойдет запись содержимого регистра D13 в регистр D24.



В зависимости от тика контроллера индексные регистры могут добавляться к следующим операндам:

ES/EX/SS: P, X, Y, M, S, KnX, KnY, KnM, KnS, T, C, D.
SA/SX/SC: P, X, Y, M, S, KnX, KnY, KnM, KnS, T, C, D
EH/EH2/SV: P, I, X, Y, M, S, K, H, KnX, KnY, KnM, KnS, T, C, D

Индексные регистры E и F могут изменять адреса операндов, перечисленных выше, но не могут изменять себя, использоваться отдельно и изменять коэффициент Kn. Операция K4M0E0 является допустимой, а K0E0M0 недопустимой.

В начале описания каждой прикладной инструкции приводится сводная таблица, где серым цветом выделены ячейки с операндами, которые могут использоваться с индексными регистрами.

При индексировании констант в командном режиме (IL, список инструкций) WPLSoft необходимо использовать символ @. Например: MOV K10@E0 D0F0.

5.4 Общий перечень прикладных инструкций по номеру API

Ниже в таблице приводится общий перечень прикладных инструкций по номеру API. В номере страниц первая цифра указывает Главу, а после тире номер страницы в данной Главе. Колонка ES включает также контроллеры EX и SS, колонка SA включает также SX и SC, а колонка EH включает EH2 и SV.
Контроллеры типов ES/EX/SS не поддерживают импульсный режим выполнения инструкций.

Инструкции, помеченные «*» применимы только к контроллерам типов EH2/SV.
Значок **P** обозначает импульсное выполнение.

Категория	API	Символ		P	Функция	Применима к			Число шагов		Стр.
		16 бит	32 бит			ES	SA	EH	16-bit	32-bit	
Loop Control	**00**	CJ	-	✓	Conditional Jump	✓	✓	✓	3	-	6-1
	01	CALL	-	✓	Call Subroutine	✓	✓	✓	3	-	6-5
	02	SRET	-	-	Subroutine Return	✓	✓	✓	1	-	6-5
	03	IRET	-	-	Interrupt Return	✓	✓	✓	1	-	6-8
	04	EI	-	-	Enable Interrupts	✓	✓	✓	1	-	6-8
	05	DI	-	-	Disable Interrupts	✓	✓	✓	1	-	6-8
	06	FEND	-	-	The End of The Main Program (First End)	✓	✓	✓	1	-	6-12
	07	WDT	-	✓	Watchdog Timer Refresh	✓	✓	✓	1	-	6-14
	08	FOR	-	-	Start of a FOR-NEXT loop	✓	✓	✓	3	-	6-15
	09	NEXT	-	-	End of a FOR-NEXT loop	✓	✓	✓	1	-	6-15
Transmission Comparison	**10**	CMP	DCMP	✓	Compare	✓	✓	✓	7	13	6-18
	11	ZCP	DZCP	✓	Zone Compare	✓	✓	✓	9	17	6-19
	12	MOV	DMOV	✓	Move	✓	✓	✓	5	9	6-20
	13	SMOV	-	✓	Shift Move	–	✓	✓	11	-	6-21
	14	CML	DCML	✓	Compliment	✓	✓	✓	5	9	6-23
	15	BMOV	-	✓	Block Move	✓	✓	✓	7	–	6-24
	16	FMOV	DFMOV	✓	Fill Move	✓	✓	✓	7	13	6-26
	17	XCH	DXCH	✓	Exchange	✓	✓	✓	5	9	6-27
	18	BCD	DBCD	✓	Binary Coded Decimal	✓	✓	✓	5	9	6-29
	19	BIN	DBIN	✓	Binary	✓	✓	✓	5	9	6-30
Four Arithmetic Operation	**20**	ADD	DADD	✓	Addition	✓	✓	✓	7	13	6-32
	21	SUB	DSUB	✓	Subtraction	✓	✓	✓	7	13	6-34
	22	MUL	DMUL	✓	Multiplication	✓	✓	✓	7	13	6-35
	23	DIV	DDIV	✓	Division	✓	✓	✓	7	13	6-37
	24	INC	DINC	✓	Increment	✓	✓	✓	3	5	6-39
	25	DEC	DDEC	✓	Decrement	✓	✓	✓	3	5	6-40
	26	WAND	DAND	✓	Logical Word AND	✓	✓	✓	7	13	6-41
	27	WOR	DOR	✓	Logical Word OR	✓	✓	✓	7	13	6-42
	28	WXOR	DXOR	✓	Logical Exclusive OR	✓	✓	✓	7	13	6-43
	29	NEG	DNEG	✓	2's Complement (Negative)	✓	✓	✓	3	5	6-44
Rotation & Displacement	**30**	ROR	DROR	✓	Rotation Right	✓	✓	✓	5	9	6-46
	31	ROL	DROL	✓	Rotation Left	✓	✓	✓	5	9	6-47
	32	RCR	DRCR	✓	Rotation Right with Carry	✓	✓	✓	5	9	6-48
	33	RCL	DRCL	✓	Rotation Left with Carry	✓	✓	✓	5	9	6-49
	34	SFTR	-	✓	Bit Shift Right	✓	✓	✓	9	-	6-50
	35	SFTL	-	✓	Bit Shift Left	✓	✓	✓	9	-	6-51
	36	WSFR	-	✓	Word Shift Right	-	✓	✓	9	-	6-52
	37	WSFL	-	✓	Word Shift Left	-	✓	✓	9	-	6-54
	38	SFWR	-	✓	Shift Register Write	-	✓	✓	7	-	6-55
	39	SFRD	-	✓	Shift Register Read	-	✓	✓	7	-	6-56
Data Processing	**40**	ZRST	-	✓	Zero Reset	✓	✓	✓	5	-	6-57
	41	DECO	-	✓	Decode	✓	✓	✓	7	-	6-59
	42	ENCO	-	✓	Encode	✓	✓	✓	7	-	6-61
	43	SUM	DSUM	✓	Sum of Active Bits	✓	✓	✓	5	9	6-63
	44	BON	DBON	✓	Check Specified Bit Status	✓	✓	✓	7	13	6-64
	45	MEAN	DMEAN	✓	Mean	✓	✓	✓	7	13	6-65
	46	ANS	-	-	Timed Annunciator Set	-	✓	✓	7	-	6-66
	47	ANR	-	✓	Annunciator Reset	-	✓	✓	1	-	6-66
	48	SQR	DSQR	✓	Square Root	✓	✓	✓	5	9	6-72
	49	FLT	DFLT	✓	Floating Point	✓	✓	✓	5	9	6-70
High Speed Processing	**50**	REF	-	✓	Refresh	✓	✓	✓	5	-	7-1
	51	REFF	-	✓	Refresh and Filter Adjust	-	✓	✓	3	-	7-2
	52	MTR	-	-	Input Matrix	-	✓	✓	9	-	7-3
	53	-	DHSCS	-	High Speed Counter Set	✓	✓	✓	-	13	7-5
	54	-	DHSCR	-	High Speed Counter Reset	✓	✓	✓	–	13	7-15

Категория	API	Символ		P	Функция	Применима к			Число шагов		Стр.
		16 бит	32 бит			ES	SA	EH	16-bit	32-bit	
High Speed Processing	**55**	-	DHSZ	-	High Speed Zone Compare	-	✓	✓	-	17	7-17
	56	SPD	-	-	Speed Detection	✓	✓	✓	7	-	7-24
	57	PLSY	DPLSY	-	Pulse Y Output	✓	✓	✓	7	13	7-26
	58	PWM	-	-	Pulse Width Modulation	✓	✓	✓	7	-	7-32
	59	PLSR	DPLSR	-	Pulse Ramp	✓	✓	✓	9	17	7-35
Handy Instructions	**60**	IST	-	-	Initial State	✓	✓	✓	7	-	7-39
	61	SER	DSER	✓	Search a Data Stack	-	✓	✓	9	17	7-45
	62	ABSD	DABSD	-	Absolute Drum Sequencer	-	✓	✓	9	17	7-46
	63	INCD	-	-	Incremental Drum Sequencer	-	✓	✓	9	-	7-48
	64	TTMR	-	-	Teaching Timer	-	✓	✓	5	-	7-50
	65	STMR	-	-	Special Timer	-	✓	✓	7	-	7-52
	66	ALT	-	✓	Alternate State	✓	✓	✓	3	-	7-54
	67	RAMP	-	-	Ramp Variable Value	-	✓	✓	9	-	7-55
	69	SORT	-	-	Sort Tabulated Data	-	✓	✓	11	-	7-57
Display of External Settings	**70**	TKY	DTKY	-	Ten Key Input	-	✓	✓	7	13	7-59
	71	HKY	DHKY	-	Hexadecimal Key Input	-	✓	✓	9	17	7-61
	72	DSW	-	-	Digital Switch	-	✓	✓	9	-	7-64
	73	SEGD	-	✓	Seven Segment Decoder	✓	✓	✓	5	-	7-66
	74	SEGL	-	-	Seven Segment with Latch	✓	✓	✓	7	-	7-67
	75	ARWS	-	-	Arrow Switch	-	✓	✓	9	-	7-70
	76	ASC	-	-	ASCII Code Conversion	-	✓	✓	11	-	7-72
	77	PR	-	-	Print (ASCII Code Output)	-	✓	✓	5	-	7-73
Serial I/O	**78**	FROM	DFROM	✓	Read CR Data in Special Modules	✓	✓	✓	9	17	7-75
	79	TO	DTO	✓	Write CR Data into Special Modules	✓	✓	✓	9	17	7-76
	80	RS	-	-	Serial Communication Instruction	✓	✓	✓	9	-	7-80
	81	PRUN	DPRUN	✓	Parallel Run	-	✓	✓	5	9	7-93
	82	ASCI	-	✓	Converts Hex to ASCII	✓	✓	✓	7	-	7-94
	83	HEX	-	✓	Converts ASCII to Hex	✓	✓	✓	7	-	7-98
	84	CCD	-	✓	Check Code	-	✓	✓	7	-	7-101
	85	VRRD	-	✓	Volume Read	-	✓	✓	5	-	7-103
	86	VRSC	-	✓	Volume Scale	-	✓	✓	5	-	7-105
	87	ABS	DABS	✓	Absolute Value	✓	✓	✓	3	5	7-106
	88	PID	DPID	-	PID Control Loop	✓	✓	✓	9	17	7-107
Basic Instructions	**89**	PLS	-	-	Rising-edge Output	✓	✓	✓	3	-	3-13
	90	LDP	-	-	Rising-edge Detection Operation	✓	✓	✓	3	-	3-11
	91	LDF	-	-	Falling-edge Detection Operation	✓	✓	✓	3	-	3-12
	92	ANDP	-	-	Rising-edge Series Connection	✓	✓	✓	3	-	3-12
	93	ANDF	-	-	Falling-edge Series Connection	✓	✓	✓	3	-	3-12
	94	ORP	-	-	Rising-edge Parallel Connection	✓	✓	✓	3	-	3-13
	95	ORF	-	-	Falling-edge Parallel Connection	✓	✓	✓	3	-	3-13
	96	TMR	-	-	16-bit Timer	✓	✓	✓	4	-	3-8
	97	CNT	DCNT	-	16-bit / 32-bit Counter	✓	✓	✓	4	6	3-9
	98	INV	-	-	Inverting Operation	✓	✓	✓	1	-	3-15
	99	PLF	-	-	Falling-edge Output	✓	✓	✓	3	-	3-14
Communication	**100**	MODRD	-	-	Read Modbus Data	✓	✓	✓	7	-	8-1
	101	MODWR	-	-	Write Modbus Data	✓	✓	✓	7	-	8-6
	102	FWD	-	-	Forward Running of VFD-A	✓	✓	✓	7	-	8-11
	103	REV	-	-	Reverse Running of VFD-A	✓	✓	✓	7	-	8-11
	104	STOP	-	-	Stop VFD-A	✓	✓	✓	7	-	8-11
	105	RDST	-	-	Read VFD-A Status	✓	✓	✓	5	-	8-14
	106	RSTEF	-	-	Reset Abnormal VFD-A	✓	✓	✓	5	-	8-16
	107	LRC	-	✓	Checksum LRC Mode	✓	✓	✓	7	-	8-17
	108	CRC	-	✓	Checksum CRC Mode	✓	✓	✓	7	-	8-19
	109	SWRD	-	✓	Read Digital Switch	-	-	✓	3	-	8-22

Категория	API	Символ		P	Функция	Применима к			Число шагов		Стр.
		16 бит	32 бит			ES	SA	EH	16-bit	32-bit	
Floating Point Operation	**110**	–	DECMP	✓	Floating Point Compare	✓	✓	✓	-	13	8-23
	111	–	DEZCP	✓	Floating Point Zone Compare	✓	✓	✓	-	17	8-24
	112	-	DMOVR	✓	Move Floating Point Data	✓	✓	✓	-	9	8-25
	116	-	DRAD	✓	Angle → Radian	-	✓	✓	-	9	8-26
	117	-	DDEG	✓	Radian → Angle	-	✓	✓	-	9	8-27
	118	-	DEBCD	✓	Float to Scientific Conversion	✓	✓	✓	-	9	8-28
	119	-	DEBIN	✓	Scientific to Float Conversion	✓	✓	✓	-	9	8-29
	120	-	DEADD	✓	Floating Point Addition	✓	✓	✓	-	13	8-31
	121	-	DESUB	✓	Floating Point Subtraction	✓	✓	✓	-	13	8-32
	122	-	DEMUL	✓	Floating Point Multiplication	✓	✓	✓	-	13	8-33
	123	-	DEDIV	✓	Floating Point Division	✓	✓	✓	-	13	8-34
	124	-	DEXP	✓	Exponent of Binary Floating Point	✓	✓	✓	-	9	8-35
	125	-	DLN	✓	Natural Logarithm of Binary Floating Point	✓	✓	✓	-	9	8-36
	126	-	DLOG	✓	Logarithm of Binary Floating Point	✓	✓	✓	-	13	8-37
	127	-	DESQR	✓	Floating Point Square Root	✓	✓	✓	-	9	8-39
	128	-	DPOW	✓	Floating Point Power Operation	✓	✓	✓	-	13	8-40
	129	INT	DINT	✓	Float to Integer	✓	✓	✓	5	9	8-42
	130	-	DSIN	✓	Sine	✓	✓	✓	-	9	8-43
	131	-	DCOS	✓	Cosine	✓	✓	✓	-	9	8-45
	132	-	DTAN	✓	Tangent	✓	✓	✓	-	9	8-47
	133	-	DASIN	✓	Arc Sine	-	✓	✓	-	9	8-49
	134	-	DACOS	✓	Arc Cosine	-	✓	✓	-	9	8-50
	135	-	DATAN	✓	Arc Tangent	-	✓	✓	-	9	8-51
	136	-	DSINH	✓	Hyperbolic Sine	-	-	✓	-	9	8-52
	137	-	DCOSH	✓	Hyperbolic Cosine	-	-	✓	-	9	8-53
	138	-	DTANH	✓	Hyperbolic Tangent	-	-	✓	-	9	8-54
Others	**143**	DELAY	-	✓	Delay Instruction	-	✓	✓	3	-	8-55
	144	GPWM	-	-	General PWM Output	-	✓	✓	7	-	8-57
	145	FTC	-	-	Fuzzy Temperature Control	-	✓	✓	9	-	8-58
	146	CVM	-	-	Valve Control (*)	-	-	✓	7	-	8-63
	147	SWAP	DSWAP	✓	Byte Swap	✓	✓	✓	3	5	8-66
	148	MEMR	DMEMR	✓	Read File Register	-	✓	✓	7	13	8-67
	149	MEMW	DMEMW	✓	Write File Register	-	✓	✓	7	13	8-68
	150	MODRW	-	-	Read/Write MODBUS Data	✓	✓	✓	11	-	9-1
	151	PWD	-	-	Detection of Input Pulse Width	-	-	✓	5	-	9-10
	152	RTMU	-	-	Start of the Measurement of Execution Time of I Interruption	-	-	✓	5	-	9-11
	153	RTMD	-	-	End of the Measurement of the Execution Time of I Interruption	-	-	✓	3	-	9-11
	154	RAND	-	✓	Random Number	-	✓	✓	7	-	9-13
Position Control	**155**	-	DABSR	-	Read the Absolute Position from a Servo Motor	-	✓	✓	7	13	9-14
	156	ZRN	DZRN	-	Zero Return	-	-	✓	9	17	9-19
	157	PLSV	DPLSV	-	Adjustable Speed Pulse Output	-	-	✓	7	13	9-23
	158	DRVI	DDRVI	-	Drive to Increment	-	-	✓	9	17	9-24
	159	DRVA	DDRVA	-	Drive to Absolute	-	-	✓	9	17	9-30
Real Time Calendar	**160**	TCMP	-	✓	Time Compare	-	✓	✓	11	-	9-39
	161	TZCP	-	✓	Time Zone Compare	-	✓	✓	9	-	9-40
	162	TADD	-	✓	Time Addition	-	✓	✓	7	-	9-41
	163	TSUB	-	✓	Time Subtraction	-	✓	✓	7	-	9-42
	166	TRD	-	✓	Time Read	-	✓	✓	3	-	9-43
	167	TWR	-	✓	Time Write	-	✓	✓	3	-	9-45
	169	HOUR	DHOUR	-	Hour Meter	-	✓	✓	7	13	9-47
	170	GRY	DGRY	✓	BIN → Gray Code	-	✓	✓	5	9	9-49
	171	GBIN	DGBIN	✓	Gray Code → BIN	-	✓	✓	5	9	9-50

Категория	API	Символ		P	Функция	Применима к			Число шагов		Стр.
		16 бит	32 бит			ES	SA	EH	16-bit	32-bit	
Floating Point Operation	**172**	-	DADDR	✓	Floating Point Addition	✓	✓	✓	-	13	9-51
	173	-	DSUBR	✓	Floating Point Subtraction	✓	✓	✓	-	13	9-53
	174	-	DMULR	✓	Floating Point Multiplication	✓	✓	✓	-	13	9-55
	175	-	DDIVR	✓	Floating Point Division	✓	✓	✓	-	13	9-57
Matrix	**180**	MAND	-	✓	Matrix 'AND' Operation	-	✓	✓	9	-	9-59
	181	MOR	-	✓	Matrix 'OR' Operation	-	✓	✓	9	-	9-61
	182	MXOR	-	✓	Matrix 'XOR' Operation	-	✓	✓	9	-	9-62
	183	MXNR	-	✓	Matrix 'XNR' Operation	-	✓	✓	9	-	9-63
	184	MINV	-	✓	Matrix Inverse Operation	-	✓	✓	7	-	9-64
	185	MCMP	-	✓	Matrix Compare	-	✓	✓	9	-	9-65
	186	MBRD	-	✓	Read Matrix Bit	-	✓	✓	7	-	9-67
	187	MBWR	-	✓	Write Matrix Bit	-	✓	✓	7	-	9-69
	188	MBS	-	✓	Matrix Bit Displacement	-	✓	✓	7	-	9-71
	189	MBR	-	✓	Matrix Bit Rotation	-	✓	✓	7	-	9-73
	190	MBC	-	✓	Matrix Bit Status Counting	-	✓	✓	7	-	9-75
Positioning Instruction	**191**	-	DPPMR	-	2-Axis Relative Point to Point Motion (*)	-	-	✓	-	17	9-76
	192	-	DPPMA	-	2-Axis Absolute Point to Point Motion (*)	-	-	✓	-	17	9-79
	193	-	DCIMR	-	2-Axis Relative Position Arc Interpolation (*)	-	-	✓	-	17	9-81
	194	-	DCIMA	-	2-Axis Absolute Position Arc Interpolation (*)	-	-	–	-	17	9-86
	195	-	DPTPO	-	Single-Axis Pulse Output by Table (*)	-	-	–	-	13	9-91
	196	HST	-	✓	High Speed Timer	-	-	✓	3	–	9-93
	197	-	DCLLM	-	Close Loop Position Control (*)	-	-	✓	-	17	9-95
	202	SCAL	-	✓	Proportional Value Calculation	✓	✓	✓	9	-	10-1
	203	SCLP	-	✓	Parameter Proportional Value Calculation	✓	✓	✓	9	-	10-3
Contact Type Logic Operation	**215**	LD&	DLD&	-	S_1 & S_2	-	✓	✓	5	9	10-7
	216	LD|	DLD|	-	S_1 | S_2	-	✓	✓	5	9	10-7
	217	LD^	DLD^	-	S_1 ^ S_2	-	✓	✓	5	9	10-7
	218	AND&	DAND&	-	S_1 & S_2	-	✓	✓	5	9	10-8
	219	AND|	DAND|	-	S_1 | S_2	-	✓	✓	5	9	10-8
	220	AND^	DAND^	-	S_1 ^ S_2	-	✓	✓	5	9	10-8
	221	OR&	DOR&	-	S_1 & S_2	-	✓	✓	5	9	10-9
	222	OR|	DOR|	-	S_1 | S_2	-	✓	✓	5	9	10-9
	223	OR^	DOR^	-	S_1 ^ S_2	-	✓	✓	5	9	10-9
Contact Type Comparison Instruction	**224**	LD=	DLD=	-	$S_1 = S_2$	✓	✓	✓	5	9	10-10
	225	LD>	DLD>	-	$S_1 > S_2$	✓	✓	✓	5	9	10-10
	226	LD<	DLD<	-	$S_1 < S_2$	✓	✓	✓	5	9	10-10
	228	LD<>	DLD<>	-	$S_1 \neq S_2$	✓	✓	✓	5	9	10-10
	229	LD<=	DLD<=	-	$S_1 \leqq S_2$	✓	✓	✓	5	9	10-10
	230	LD>=	DLD>=	-	$S_1 \geqq S_2$	✓	✓	✓	5	9	10-10
	232	AND=	DAND=	-	$S_1 = S_2$	✓	✓	✓	5	9	10-11
	233	AND>	DAND>	-	$S_1 > S_2$	✓	✓	✓	5	9	10-11
	234	AND<	DAND<	-	$S_1 < S_2$	✓	✓	✓	5	9	10-11
	236	AND<>	DAND<>	-	$S_1 \neq S_2$	✓	✓	✓	5	9	10-11
	237	AND<=	DAND<=	-	$S_1 \leqq S_2$	✓	✓	✓	5	9	10-11
	238	AND>=	DAND>=	-	$S_1 \geqq S_2$	✓	✓	✓	5	9	10-11
	240	OR=	DOR=	-	$S_1 = S_2$	✓	✓	✓	5	9	10-12
	241	OR>	DOR>	-	$S_1 > S_2$	✓	✓	✓	5	9	10-12
	242	OR<	DOR<	-	$S_1 < S_2$	✓	✓	✓	5	9	10-12
	244	OR<>	DOR<>	-	$S_1 \neq S_2$	✓	✓	✓	5	9	10-12

Категория	API	Символ		P	Функция	Применима к			Число шагов		Стр.
		16 бит	32 бит			ES	SA	EH	16-bit	32-bit	
	245	OR<=	DOR<=	-	$S_1 \leqq S_2$	✓	✓	✓	5	9	10-12
	246	OR>=	DOR>=	-	$S_1 \geqq S_2$	✓	✓	✓	5	9	10-12

5.5 Общий перечень прикладных инструкций по алфавиту

Ниже в таблице приводится общий перечень прикладных инструкций по алфавиту. В номере страниц первая цифра указывает Главу, а после тире номер страницы в данной Главе. Колонка ES включает также контроллеры EX и SS, колонка SA включает также SX и SC, а колонка EH включает EH2 и SV.
Контроллеры типов ES/EX/SS не поддерживают импульсный режим выполнения инструкций.
Инструкции, помеченные «*» применимы только к контроллерам типов EH2/SV.
Значок **P** обозначает импульсное выполнение.

Категория	API	Символ		P	Функция	Применима к			Стр.
		16 бит	32 бит			ES	SA	EH	
A	**20**	ADD	DADD	✓	Addition	✓	✓	✓	6-32
	46	ANS	-	-	Timed Annunciator Set	-	✓	✓	6-66
	47	ANR	-	✓	Annunciator Reset	-	✓	✓	6-66
	62	ABSD	DABSD	-	Absolute Drum Sequencer	-	✓	✓	7-46
	66	ALT	-	✓	Alternate State	✓	✓	✓	7-54
	75	ARWS	-	-	Arrow Switch	-	✓	✓	7-70
	76	ASC	-	-	ASCII Code Conversion	-	✓	✓	7-72
	82	ASCI	-	✓	Converts Hex to ASCII	✓	✓	✓	7-94
	87	ABS	DABS	✓	Absolute Value	✓	✓	✓	7-106
	133	-	DASIN	✓	Arc Sine	-	✓	✓	8-49
	134	-	DACOS	✓	Arc Cosine	-	✓	✓	8-50
	135	-	DATAN	✓	Arc Tangent	-	✓	✓	8-51
	155	-	DABSR	-	Read the Absolute Position from a Servo Motor	-	✓	✓	9-14
	172	-	DADDR	✓	Floating Point Addition	✓	✓	✓	9-51
	218	AND&	DAND&	-	S_1 & S_2	-	✓	✓	10-8
	219	AND|	DAND|	-	S_1 | S_2	-	✓	✓	10-8
	220	AND^	DAND^	-	S_1 ^ S_2	-	✓	✓	10-8
	232	AND=	DAND=	-	$S_1 = S_2$	✓	✓	✓	10-11
	233	AND>	DAND>	-	$S_1 > S_2$	✓	✓	✓	10-11
	234	AND<	DAND<	-	$S_1 < S_2$	✓	✓	✓	10-11
	236	AND<>	DAND<>	-	$S_1 \neq S_2$	✓	✓	✓	10-11
	237	AND<=	DAND<=	-	$S_1 \leqq S_2$	✓	✓	✓	10-11
	238	AND>=	DAND>=	-	$S_1 \geqq S_2$	✓	✓	✓	10-11
B	**15**	BMOV	-	✓	Block Move	✓	✓	✓	6-24
	18	BCD	DBCD	✓	Binary Coded Decimal	✓	✓	✓	6-29
	19	BIN	DBIN	✓	Binary	✓	✓	✓	6-30
	44	BON	DBON	✓	Check Specified Bit Status	✓	✓	✓	6-64
C	**00**	CJ	-	✓	Conditional Jump	✓	✓	✓	6-1
	01	CALL	-	✓	Call Subroutine	✓	✓	✓	6-5
	10	CMP	DCMP	✓	Compare	✓	✓	✓	6-18
	14	CML	DCML	✓	Compliment	✓	✓	✓	6-23
	84	CCD	-	✓	Check Code	-	✓	✓	7-101
	108	CRC	-	✓	Checksum CRC Mode	-	✓	✓	8-19
	131	-	DCOS	✓	Cosine	✓	✓	✓	8-45
	137	-	DCOSH	✓	Hyperbolic Cosine	-	-	✓	8-53
	146	CVM	-	-	Valve Control (*)	-	-	✓	8-63
	193	-	DCIMR	-	2-Axis Relative Position Arc Interpolation (*)	-	-	✓	9-81
	194	-	DCIMA	-	2-Axis Absolute Position Arc Interpolation (*)	-	-	✓	9-86
	197	-	DCLLM	-	Close Loop Position Control (*)	-	-	✓	9-95

Категория	API	Символ		P	Функция	Применима к			Стр.
		16 бит	32 бит			ES	SA	EH	
D	**05**	DI	-	-	Disable Interrupts	✓	✓	✓	6-8
	23	DIV	DDIV	✓	Division	✓	✓	✓	6-37
	25	DEC	DDEC	✓	Decrement	✓	✓	✓	6-40
	41	DECO	-	✓	Decode	✓	✓	✓	6-59
	72	DSW	-	-	Digital Switch	-	✓	✓	7-64
	117	-	DDEG	✓	Radian → Angle	-	✓	✓	8-27
	143	DELAY	-	✓	Delay Instruction	-	✓	✓	8-55
D	**158**	DRVI	DDRVI	-	Drive to Increment	-	-	✓	9-24
	159	DRVA	DDRVA	-	Drive to Absolute	-	-	✓	9-30
	175	-	DDIVR	✓	Floating Point Division	✓	✓	✓	9-57
E	**04**	EI	-	-	Enable Interrupts	✓	✓	✓	6-8
	42	ENCO	-	✓	Encode	✓	✓	✓	6-61
	110	-	DECMP	✓	Floating Point Compare	✓	✓	✓	8-23
	111	-	DEZCP	✓	Floating Point Zone Compare	✓	✓	✓	8-24
	118	-	DEBCD	✓	Float to Scientific Conversion	✓	✓	✓	8-28
	119	-	DEBIN	✓	Scientific to Float Conversion	✓	✓	✓	8-29
	120	-	DEADD	✓	Floating Point Addition	✓	✓	✓	8-31
	121	-	DESUB	✓	Floating Point Subtraction	✓	✓	✓	8-32
	122	-	DEMUL	✓	Floating Point Multiplication	✓	✓	✓	8-33
	123	-	DEDIV	✓	Floating Point Division	✓	✓	✓	8-34
	124	-	DEXP	✓	Exponent of Binary Floating Point	✓	✓	✓	8-35
	127	-	DESQR	✓	Floating Point Square Root	✓	✓	✓	8-39
F	**06**	FEND	-	-	The End of the Main Program (First End)	✓	✓	✓	6-12
	08	FOR	-	-	Start of a FOR-NEXT Loop	✓	✓	✓	6-15
	16	FMOV	DFMOV	✓	Fill Move	✓	✓	✓	6-26
	49	FLT	DFLT	✓	Floating Point	✓	✓	✓	6-70
	78	FROM	DFROM	✓	Read CR Data in Special Modules	✓	✓	✓	7-75
	102	FWD	-	-	Forward Running of VFD-A	✓	✓	✓	8-11
	145	FTC	-	-	Fuzzy Temperature Control	-	✓	✓	8-58
G	**144**	GPWM	-	-	General PWM Output	-	✓	✓	8-57
	170	GRY	DGRY	✓	BIN → Gray Code	-	✓	✓	9-49
	171	GBIN	DGBIN	✓	Gray Code → BIN	-	✓	✓	9-50
H	**53**	-	DHSCS	-	High Speed Counter Set	✓	✓	✓	7-5
	54	-	DHSCR	-	High Speed Counter Reset	✓	✓	✓	7-15
	55	-	DHSZ	-	High Speed Zone Compare	-	✓	✓	7-17
	71	HKY	DHKY	-	Hexadecimal Key Input	-	✓	✓	7-61
	83	HEX	-	✓	Converts ASCII to Hex	✓	✓	✓	7-98
	169	HOUR	DHOUR	-	Hour Meter	-	✓	✓	9-47
	196	HST	-	✓	High Speed Timer	-	-	✓	9-93
I	**03**	IRET	-	-	Interrupt Return	✓	✓	✓	6-8
	24	INC	DINC	✓	Increment	✓	✓	✓	6-39
	60	IST	-	-	Initial State	✓	✓	✓	7-39
	63	INCD	-	-	Increment Drum Sequencer	-	✓	✓	7-48
	129	INT	DINT	✓	Float to Integer	✓	✓	✓	8-42
L	**107**	LRC	-	✓	Checksum LRC Mode	✓	✓	✓	8-17
	125	-	DLN	✓	Natural Logarithm of Binary Floating Point	✓	✓	✓	8-36
	126	-	DLOG	✓	Logarithm of Binary Floating Point	✓	✓	✓	8-37
	215	LD&	DLD&	-	S_1 & S_2	-	✓	✓	10-7
	216	LD|	DLD|	-	S_1 | S_2	-	✓	✓	10-7

Категория	API	Символ		P	Функция	Применима к			Стр.
		16 бит	32 бит			ES	SA	EH	
	217	LD^	DLD^	-	$S_1 \wedge S_2$	-	✓	✓	10-7
	224	LD=	DLD=	-	$S_1 = S_2$	✓	✓	✓	10-10
	225	LD>	DLD>	-	$S_1 > S_2$	✓	✓	✓	10-10
	226	LD<	DLD<	-	$S_1 < S_2$	✓	✓	✓	10-10
	228	LD<>	DLD<>	-	$S_1 \neq S_2$	✓	✓	✓	10-10
L	**229**	LD<=	DLD<=	-	$S_1 \leqq S_2$	✓	✓	✓	10-10
	230	LD>=	DLD>=	-	$S_1 \geqq S_2$	✓	✓	✓	10-10
	12	MOV	DMOV	✓	Move	✓	✓	✓	6-20
	22	MUL	DMUL	✓	Multiplication	✓	✓	✓	6-35
	45	MEAN	DMEAN	✓	Mean	✓	✓	✓	6-65
	52	MTR	-	-	Input Matrix	-	✓	✓	7-3
	100	MODRD	-	-	Read Modbus Data	✓	✓	✓	8-1
	101	MODWR	-	-	Write Modbus Data	✓	✓	✓	8-6
	112	-	DMOVR	✓	Move Floating Point Data	✓	✓	✓	8-25
	148	MEMR	DMEMR	✓	Read File Register	-	✓	✓	8-67
	149	MEMW	DMEMW	✓	Write File Register	-	✓	✓	8-68
	150	MODRW	-	-	Read/Write MODBUS Data	✓	✓	✓	9-1
	174	-	DMULR	✓	Floating Point Multiplication	✓	✓	✓	9-55
M	**180**	MAND	-	✓	Matrix ‘AND’ Operation	-	✓	✓	9-59
	181	MOR	-	✓	Matrix ‘OR’ Operation	-	✓	✓	9-61
	182	MXOR	-	✓	Matrix ‘XOR’ Operation	-	✓	✓	9-62
	183	MXNR	-	✓	Matrix ‘NOR’ Operation	-	✓	✓	9-63
	184	MINV	-	✓	Matrix Inverse Operation	-	✓	✓	9-64
	185	MCMP	-	✓	Matrix Compare	-	✓	✓	9-65
	186	MBRD	-	✓	Read Matrix Bit	-	✓	✓	9-67
	187	MBWR	-	✓	Write Matrix Bit	-	✓	✓	9-69
	188	MBS	-	✓	Matrix Bit Displacement	-	✓	✓	9-71
	189	MBR	-	✓	Matrix Bit Rotation	-	✓	✓	9-73
	190	MBC	-	✓	Matrix Bit Status Counting	-	✓	✓	9-75
N	**09**	NEXT	-	-	End of a FOR-NEXT Loop	✓	✓	✓	6-15
	29	NEG	DNEG	✓	2’s Complement (Negative)	✓	✓	✓	6-44
	221	OR&	DOR&	-	$S_1 \& S_2$	-	✓	✓	10-9
	222	OR|	DOR|	-	$S_1 \mid S_2$	-	✓	✓	10-9
	223	OR^	DOR^	-	$S_1 \wedge S_2$	-	✓	✓	10-9
	240	OR=	DOR=	-	$S_1 = S_2$	✓	✓	✓	10-12
O	**241**	OR>	DOR>	-	$S_1 > S_2$	✓	✓	✓	10-12
	242	OR<	DOR<	-	$S_1 < S_2$	✓	✓	✓	10-12
	244	OR<>	DOR<>	-	$S_1 \neq S_2$	✓	✓	✓	10-12
	245	OR<=	DOR<=	-	$S_1 \leqq S_2$	✓	✓	✓	10-12
	246	OR>=	DOR>=	-	$S_1 \geqq S_2$	✓	✓	✓	10-12
	57	PLSY	DPLSY	-	Pulse Y Output	✓	✓	✓	7-26
	58	PWM	-	-	Pulse Width Modulation	✓	✓	✓	7-32
	59	PLSR	DPLSR	-	Pulse Ramp	✓	✓	✓	7-35
	77	PR	-	-	Print (ASCII Code Output)	-	✓	✓	7-73
P	**81**	PRUN	DPRUN	✓	Parallel Run	-	✓	✓	7-93
	88	PID	DPID	-	PID Control Loop	✓	✓	✓	7-107
	128	-	DPOW	✓	Floating Point Power Operation	✓	✓	✓	8-40
	151	PWD	-	-	Detection of Input Pulse Width	-	-	✓	9-10
	157	PLSV	DPLSV	-	Adjustable Speed Pulse Output	-	-	✓	9-23
	191	-	DPPMR	-	2-Axis Relative Point to Point Motion (*)	-	-	✓	9-76

Категория	API	Символ		P	Функция	Применима к			Стр.
		16 бит	32 бит			ES	SA	EH	
	192	-	DPPMA	-	2-Axis Absolute Point to Point Motion (*)	-	-	✓	9-79
	195	-	DPTPO	-	Single-Axis Pulse Output by Table (*)	-	-	✓	9-91
R	**30**	ROR	DROR	✓	Rotation Right	✓	✓	✓	6-46
	31	ROL	DROL	✓	Rotation Left	✓	✓	✓	6-47
R	**32**	RCR	DRCR	✓	Rotation Right with Carry	✓	✓	✓	6-48
	33	RCL	DRCL	✓	Rotation Left with Carry	✓	✓	✓	6-49
	50	REF	-	✓	Refresh	✓	✓	✓	7-1
	51	REFF	-	✓	Refresh and Filter Adjust	-	✓	✓	7-2
	67	RAMP	-	-	Ramp Variable Value	-	✓	✓	7-55
	80	RS	-	-	Serial Communication Instruction	✓	✓	✓	7-80
	103	REV	-	-	Reverse Running of VFD-A	✓	✓	✓	8-11
	105	RDST	-	-	Read VFD-A Status	✓	✓	✓	8-14
	106	RSTEF	-	-	Reset Abnormal VFD-A	✓	✓	✓	8-16
	116	-	DRAD	✓	Angle → Radian	-	✓	✓	8-26
	152	RTMU	-	-	Start of the Measurement of Execution Time of I Interruption	-	-	✓	9-11
	153	RTMD	-	-	End of the Measurement of the Execution Time of I Interruption	-	-	✓	9-11
	154	RAND	✓	-	Random Number	-	✓	✓	9-13
S	**02**	SRET	-	-	Subroutine Return	✓	✓	✓	6-5
	13	SMOV	-	✓	Shift Move	-	✓	✓	6-21
	21	SUB	DSUB	✓	Subtraction	✓	✓	✓	6-34
	34	SFTR	-	✓	Bit Shift Right	✓	✓	✓	6-50
	35	SFTL	-	✓	Bit Shift Left	✓	✓	✓	6-51
	38	SFWR	-	✓	Shift Register Write	-	✓	✓	6-55
	39	SFRD	-	✓	Shift Register Read	-	✓	✓	6-56
	43	SUM	DSUM	✓	Sum of Active Bits	✓	✓	✓	6-63
	48	SQR	DSQR	✓	Square Root	✓	✓	✓	6-72
	56	SPD	-	-	Speed Detection	✓	✓	✓	7-24
	61	SER	DSER	✓	Search a Data Stack	-	✓	✓	7-45
	65	STMR	-	-	Special Timer	-	✓	✓	7-52
	69	SORT	-	-	Sort Tabulated Data	-	✓	✓	7-57
	73	SEGD	-	✓	Seven Segment Decoder	✓	✓	✓	7-66
	74	SEGL	-	-	Seven Segment with Latch	✓	✓	✓	7-67
	104	STOP	-	-	Stop VFD-A	✓	✓	✓	8-11
	109	SWRD	-	✓	Read Digital Switch	-	-	✓	8-22
	130	-	DSIN	✓	Sine	✓	✓	✓	8-43
	136	-	DSINH	✓	Hyperbolic Sine	-	-	✓	8-52
	147	SWAP	DSWAP	✓	Byte Swap	✓	✓	✓	8-66
	173	-	DSUBR	✓	Floating Point Subtraction	✓	✓	✓	9-53
	202	SCAL	-	✓	Proportional Value Calculation	✓	✓	✓	10-1
	203	SCLP	-	✓	Parameter Proportional Value Calculation	✓	✓	✓	10-3
T	**64**	TTMR	-	-	Teaching Timer	-	✓	✓	7-50
	70	TKY	DTKY	-	Ten Key Input	-	✓	✓	7-59
	79	TO	DTO	✓	Write CR Data into Special Modules	✓	✓	✓	7-76
	132	-	DTAN	✓	Tangent	✓	✓	✓	8-47
	138	-	DTANH	✓	Hyperbolic Tangent	-	-	✓	8-54
	160	TCMP	-	✓	Time Compare	-	✓	✓	9-39
	161	TZCP	-	✓	Time Zone Compare	-	✓	✓	9-40
	162	TADD	-	✓	Time Addition	-	✓	✓	9-41

Категория	API	Символ		P	Функция	Применима к			Стр.
		16 бит	32 бит			ES	SA	EH	
	163	TSUB	-	✓	Time Subtraction	-	✓	✓	9-42
	166	TRD	-	✓	Time Read	-	✓	✓	9-43
T	**167**	TWR	-	✓	Time Write	-	✓	✓	9-45
V	**85**	VRRD	-	✓	Volume Read	-	✓	✓	7-103
	86	VRSC	-	✓	Volume Scale	-	✓	✓	7-105
W	**07**	WDT	-	✓	Watchdog Timer Refresh	✓	✓	✓	6-14
	26	WAND	DAND	✓	Logical Word AND	✓	✓	✓	6-41
	27	WOR	DOR	✓	Logical Word OR	✓	✓	✓	6-42
	28	WXOR	DXOR	✓	Logical Exclusive OR	✓	✓	✓	6-43
	36	WSFR	-	✓	Word Shift Right	-	✓	✓	6-52
	37	WSFL	-	✓	Word Shift Left	-	✓	✓	6-54
X	**17**	XCH	DXCH	✓	Exchange	✓	✓	✓	6-27
Z	**11**	ZCP	DZCP	✓	Zone Compare	✓	✓	✓	6-19
	40	ZRST	-	✓	Zero Reset	✓	✓	✓	6-57
	156	ZRN	DZRN	-	Zero Return	-	-	✓	9-19

API	Mnemonic		Operands	Function	Controllers		
					ES/EX/SS	SA/SX/SC	EH/SV
00	CJ	P	(S)	Conditional Jump			

OP	Range	Program Steps
(S)	P0~P255	CJ, CJP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

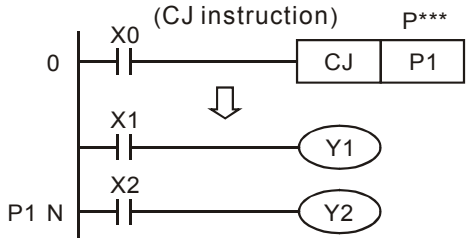
S: The destination pointer of conditional jump

Explanations:

1. Operand **S** can designate P
2. P can be modified by index register E, F
3. In ES/EX/SS series models: Operand S can designate P0 ~ P63
4. In SA/SX/SC/EH/EH2/SV series models: Operand S can designate P0 ~ P255
5. When the user does not wish a particular part of PLC program in order to shorten the scan time and execute dual outputs, CJ instruction or CJP instruction can be adopted.
6. When the program designated by pointer P is prior to CJ instruction, WDT timeout will occur and PLC will stop running. Please use it carefully.
7. CJ instruction can designate the same pointer P repeatedly. However, CJ and CALL cannot designate the same pointer P; otherwise an error will occur.
8. Actions of all devices while conditional jumping is being executed.
   a) Y, M and S remain their previous status before the conditional jump takes place.
   b) Timer 10ms and 100ms that is executing stops.
   c) Timer T192 ~ T199 that execute the subroutine program will continue and the output contact executes normally.
   d) The high-speed counter that is executing the counting continues counting and the output contact executes normally.
   e) The ordinary counters stop executing.
   f) If the “reset instruction” of the timer is executed before the conditional jump, the device will still be in the reset status while conditional jumping is being executed.
   g) Ordinary application instructions are not executed.
   h) The application instructions that are being executed, i.e. API 53 DHSCS, API 54 DHSCR, API 55 DHSZ, API 56 SPD, API 57 PLSY, API 58 PWM, API 59 PLSR, API 157 PLSV, API 158 DRVI, API 159 DRVA, continue being executed.

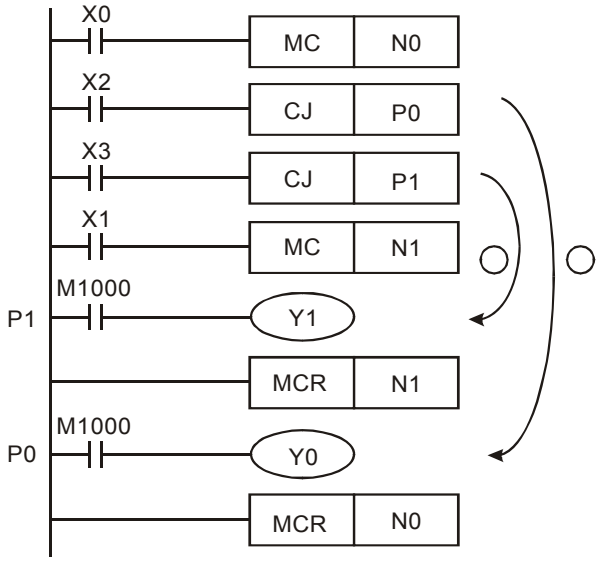
Program Example 1:

1. When X0 = On, the program automatically jumps from address 0 to N (the designated label P1) and keeps its execution. The addresses between 0 and N will not be executed.
2. When X0 = Off, as an ordinary program, the program keeps on executing from address 0. CJ instruction will not be executed at this time.



Program Example 2:

1. CJ instruction can be used in the following 5 conditions between MC and MCR instructions.
   a) Without MC ~ MCR.
   b) From without MC to within MC. Valid in the loop P1 as shown in the figure below.
   c) In the same level N, inside of MC~MCR.
   d) From within MC to without MCR.
   e) Jumping from this MC ~ MCR to another MC ~ MCR[1].
2. Actions in ES/EX/SS series models V4.7 (and below): When CJ instruction is used between MC and MCR, it can only be applied without MC ~ MCR or in the same N layer of MC ~ MCR. Jumping from this MC ~ MCR to another MC ~ MCR will result in errors, i.e. a) and c) as stated above can ensure correct actions; others will cause errors.
3. When MC instruction is executed, PLC will push the status of the switch contact into the self-defined stack in PLC. The stack will be controlled by the PLC, and the user cannot change it. When MCR instruction is executed, PLC will obtain the previous status of the switch contact from the top layer of the stack. Under the conditions as stated in b), d) and e), the times of pushing-in and obtaining stack may be different. In this case, the maximum stack available to be pushed in is 8 and the obtaining of stacks cannot resume once the stack becomes empty. Thus, when using CALL or CJ instructions, the user has to be aware of the pushing-in and obtaining of stacks.



[1] This function is only available in ES/EX/SS series models V4.9 (and above) and SA/SX/SC/EH/EH2/SV series models.

Program Example 3:

1. The states of each device

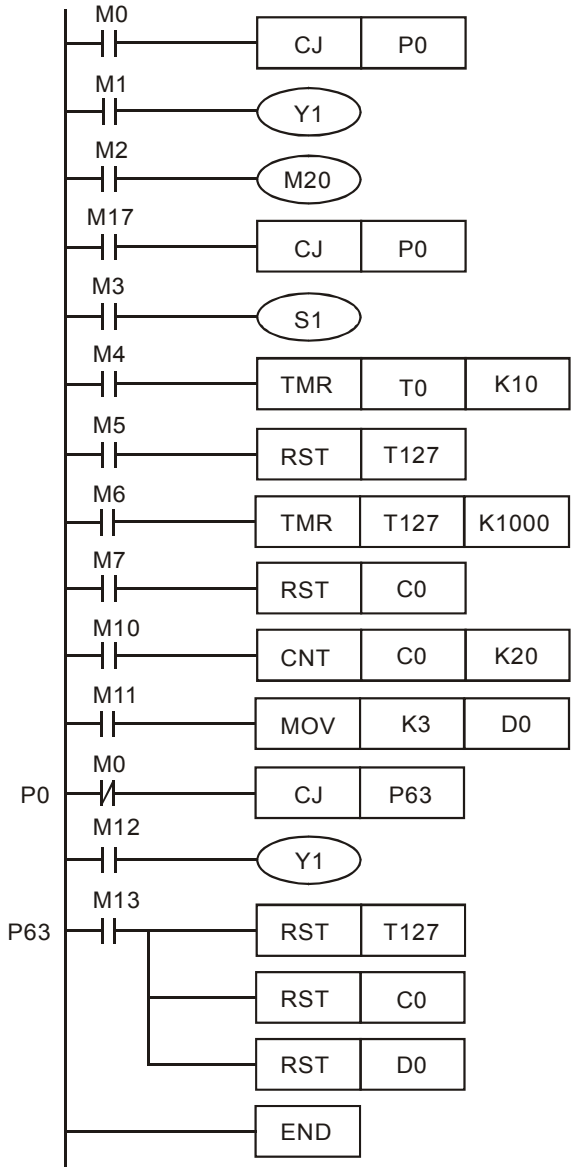
Device	Contact state before CJ is executed	Contact state when CJ is being executed	Output coil state when CJ is being executed
Y, M, S	M1, M2, M3 Off	M1, M2, M3 Off→On	Y1 *1, M20, S1 Off
	M1, M2, M3 On	M1, M2, M3 On→Off	Y1 *1, M20, S1 On
10ms, 100ms Timer ES/SA/EH	M4 Off	M4 Off→On	Timer T0 is not enabled.
	M4 On	M4 On→Off	Timer T0 immediately stops and is latched. M0 On→Off, T0 is reset as 0.
1ms, 10ms, 100ms Timer *2 (accumulative) SA/EH	M6 Off	M6 Off→On	Timer T240 is not enabled.
	M6 On	M6 On→Off	Once the timer function is enabled and when met with CJ instruction, all accumulative timers will stop timing and stay latched. M0 On→Off. T240 remains unchanged.
C0 ~ C234 *3	M7, M10 Off	M10 On/Off trigger	Counter does not count.
	M7 Off, M10 On/Off trigger	M10 On/Off trigger	Counter C0 stops counting and stays latched. After M0 goes Off, C0 resumes its counting.
Application instruction	M11 Off	M11 Off→On	Application instructions are not executed.
	M11 On	M11 On→Off	The skipped application instructions are not executed, but API 53 ~ 59, API 157 ~ 159 keep being executed.

*1: Y1 is a dual output. When M0 is Off, M1 will control Y1. When M0 is On, M12 will control Y1.

*2: When the timers (T192 ~ T199, applicable in SA/EH series MPU) used by a subroutine re driven and encounter the execution of CJ instruction, the timing will resume. After the timing target is reached, the output contact of the timer will be On.

*3: When the high-speed counters (C235 ~ C255) are driven and encounter the execution of CJ instruction, the counting will resume, as well as the action of the output points.

2. Y1 is a dual output. When M0 = Off, Y1 is controlled by M1. When M0 = On, Y1 is controlled by M12.



API	Mnemonic	Operands	Function	Controllers
01	CALL P	(S)	Call Subroutine	ES/EX/SS | SA/SX/SC | EH/SV

OP	Range	Program Steps
(S)	P0 ~ P255	CALL, CALLP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: The pointer of call subroutine.

Explanations:

1. Operand S can designate P.
2. P can be modified by index register E, F.
3. In ES/EX/SS series models: Operand S can designate P0 ~ P63.
4. In SA/SX/SC/EH/EH2/SV series models: Operand S can designate P0 ~ P255.
5. Edit the subroutine designated by the pointer after FEND instruction.
6. The number of pointer P, when used by CALL, cannot be the same as the number designated by CJ instruction.
7. If only CALL instruction is in use, it can call subroutines of the same pointer number with no limit on times.
8. Subroutine can be nested for 5 levels including the initial CALL instruction. (If entering the sixth level, the subroutine won't be executed.)

API	Mnemonic	Function	Controllers
02	SRET	Subroutine Return	ES/EX/SS | SA/SX/SC | EH/SV

OP	Descriptions	Program Steps
N/A	Automatically returns to the step immediately following the CALL instruction which activated the subroutine	SRET: 1 steps

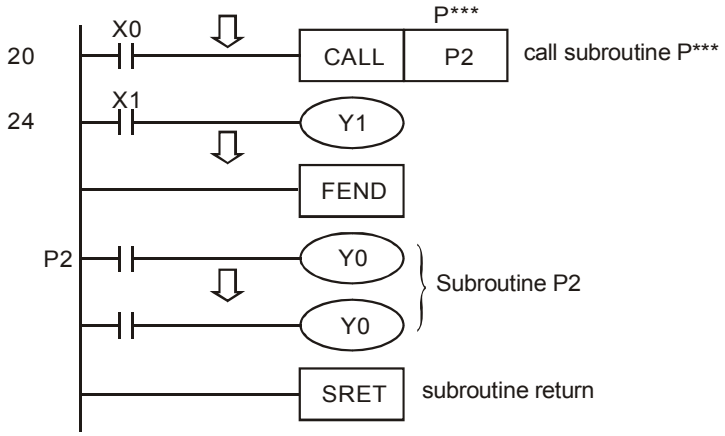
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Explanations:

1. No operand. No contact to drive the instruction is required.
2. The subroutine will return to main program by SRET after the termination of subroutine and execute the sequence program located at the next step to the CALL instruction.

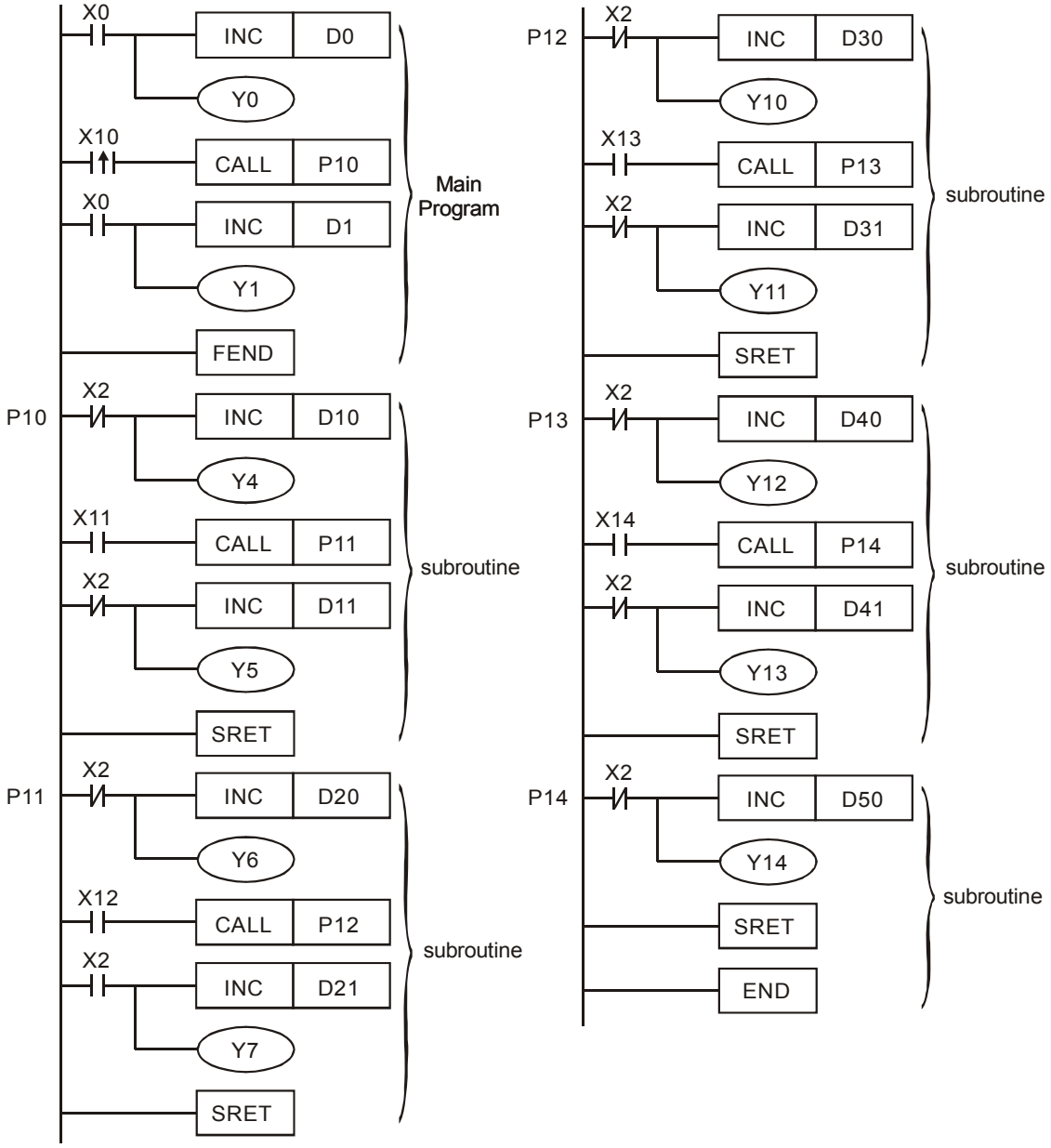
Program Example 1:

When X0 = On, CALL instruction is executed and the program jumps to the subroutine designated by P2. When SRET instruction is executed, the program returns to address 24 and continues its execution.



Program Example 2:

1. When X10 goes from Off to On, its rising-edge trigger executes CALL P10 instruction and the program jumps to the subroutine designated by P10.
2. When X11 is On, CALL P11 is executed and the program jumps to the subroutine designated by P11.
3. When X12 is On, CALL P12 is executed and the program jumps to the subroutine designated by P12.
4. When X13 is On, CALL P13 is executed and the program jumps to the subroutine designated by P13.
5. When X14 is On, CALL P14 is executed and the program jumps to the subroutine designated by P14. When SRET is executed, the program returns to the previous P※ subroutine and continues its execution.
6. After SRET instruction is executed in P10 subroutine, returning to the main program.



API	Mnemonic	Function	Controllers
03	IRET	Interrupt Return	ES/EX/SS | SA/SX/SC | EH/SV

OP	Descriptions	Program Steps
N/A	IRET ends the processing of an interruption subroutine and returns to the execution of the main program.	IRET: 1 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Explanations:

1. No operand. No contact to drive the instruction is required.
2. Interruption return refers to interrupt the subroutine.
3. After the interruption is over, returning to the main program from IRET to execute the next instruction where the program was interrupted.

API	Mnemonic	Function	Controllers
04	EI	Enable Interrupts	ES/EX/SS | SA/SX/SC | EH/SV

OP	Descriptions	Program Steps
N/A	See more details of the explanation on this instruction in DI (Disable Interruption) instruction.	EI: 1 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Explanations:

1. No operand. No contact to drive the instruction is required.
2. The pulse width of the interruption signal should be >200us.
3. See DI instruction for the range of the No. of I for all models.
4. See DI instruction for more details about M1050 ~ M1059, M1280 ~ M1299.

API	Mnemonic	Function	Controllers
05	DI	Disable Interrupts	ES/EX/SS | SA/SX/SC | EH/SV

OP	Descriptions	Program Steps
N/A	When the special auxiliary relay M1050 ~ M1059, M1280 ~ M1299 for disabling interruption is driven, the corresponding interruption request will not be executed even in the range allowed for interruptions.	DI: 1 step

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

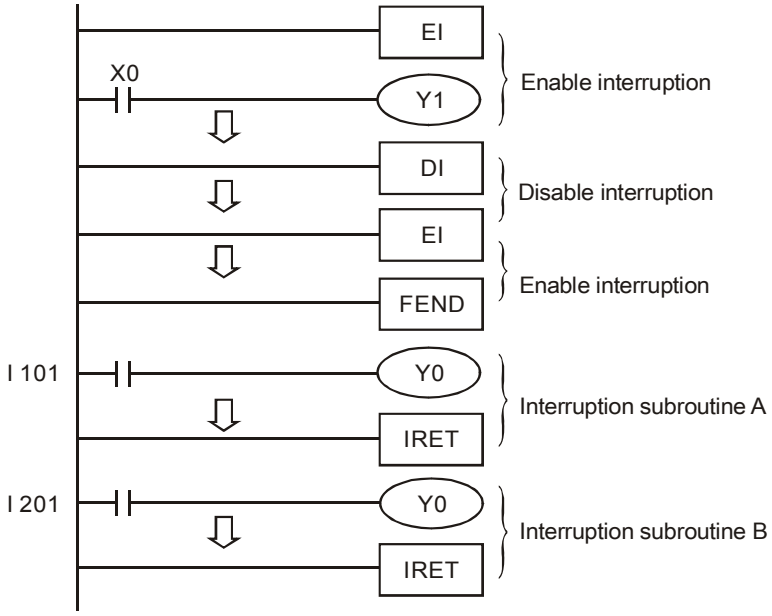
Explanations:

1. No operand. No contact to drive the instruction is required.
2. EI instruction allows interrupting subroutine in the program, e.g. external interruption, timed interruption, and high-speed counter interruption.

3. In the program, using interruption subroutine between EI and DI instruction is allowed. However, you can choose not to use DI instruction if there is no interruption-disabling section in the program.
4. When M1050 ~ M1059 are the special auxiliary relays to drive disabling interruption in ES/SA, or M1280 ~ M1299 are the special auxiliary relays to drive disabling interruption in EH/EH2/SV, the corresponding interruptions will not be executed even in the area allowed for interruptions.
5. Pointer for interruption (I) must be placed after FEND instruction.
6. Other interruptions are not allowed during the execution of interruption subroutine.
7. When many interruptions occur, the priority is given to the firstly executed interruption. If several interruptions occur simultaneously, the priority is given to the interruption with the smaller pointer No.
8. The interruption request occurring between DI and EI instructions that cannot be executed immediately will be memorized and will be executed in the area allowed for interruption.
9. The time interruptions in ES/SA will not be memorized.
10. When using the interruption pointer, DO NOT repeatedly use the high-speed counter driven by the same X input contact.
11. When immediate I/O is required during the interruption, write REF instruction in the program to update the status of I/O.

Program Example:

During the operation of PLC, when the program scans to the area between EI and DI instructions and X1 = Off→On or X2 = Off→On, interruption subroutine A or B will be executed. When the subroutine executes to IRET, the program will return to the main program and resumes its execution.



Remarks:

1. No. of interruption pointer I in ES/EX/SS:
   a) External interruptions: (I001, X0), (I101, X1), (I201, X2), (I301, X3) 4 points[2].
   b) Time interruptions: I6□□, 1 point (□□ = 10 ~ 99, time base = 1ms) (support V5.7 and above)

[2] Input points occupied by external interruptions cannot be used for inputs of high-speed counters; otherwise grammar check errors may occur when the program is written in PLC.

c) Communication interruption for receiving specific words (I150) (support V5.7 and above)

2. No. of interruption pointer I in SA/SX/SC:

a) External interruptions: (I001, X0), (I101, X1), (I201, X2), (I301, X3), (I401, X4), (I501, X5) 6 points.

b) Time interruptions: I6□□, I7□□ 2 points. (□□ = 1 ~ 99ms, time base = 1ms)

c) High-speed counter interruptions: I010, I020, I030, I040 4 points. (used with API 53 DHSCS instruction to generate interruption signals)

d) Communication interruption for receiving specific words .(I150)

e) The order for execution of interruption pointer I: high-speed counter interruption, external interruption, time interruption and communication interruption for receiving specific words.

f) Among the following 6 interruption No., (I001, I010), (I101, I020), (I201, I030), (I301, I040), (I401, I050), (I501, I060), the program allows the user to use only one of the two numbers in a pair. If the user uses the two numbers in the pair, grammar check errors may occur when the program is written into PLC.

3. No. of interruption pointer I in EH/EH2/SV:

a) External interruptions: (I00□, X0), (I10□, X1), (I20□, X2), (I30□, X3), (I40□, X4), (I50□, X5) 6 points. (□ = 0 designates interruption in falling-edge, □ = 1 designates interruption in rising-edge)

b) Time interruptions: I6□□, I7□□, 2 points. (□□ = 1~99ms, time base = 1ms)

I8□□ 1 point. (□□ = 1 ~ 99ms, time base = 0.1ms)

c) High-speed counter interruptions: I010, I020, I030, I040, 1050, 1060 6 points. (used with API 53 DHSCS instruction to generate interruption signals)

d) When pulse output interruptions I110, I120 (triggered when pulse output is finished), I130, I140 (triggered when the first pulse output starts) are executed, the currently executed program is interrupted and jumps to the designated interruption subroutine.

e) Communication interruption: I150, I160, I170

f) Frequency measurement card interruption: I180

g) The order for execution of interruption pointer I: external interruption, time interruption, high-speed counter interruption, pulse interruption, communication interruption and frequency measurement card interruption.

4. “Disable interruption” flags in ES/EX/SS:

Flag	Function
M1050	Disable external interruption I001
M1051	Disable external interruption I101
M1052	Disable external interruption I201
M1053	Disable external interruption I301
M1056	Disable time interruption I6□□

5. “Disable interruption” flags in SA/SX/SC:

Flag	Function
M1050	Disable external interruption I001
M1051	Disable external interruption I101
M1052	Disable external interruption I201
M1053	Disable external interruption I301
M1054	Disable external interruption I401
M1055	Disable external interruption I501

M1056	Disable time interruption I6□□
M1057	Disable time interruption I7□□
M1059	Disable high-speed counter interruption I010 ~ I060

6. “Disable interruption” flags in EH/EH2/SV:

Flag	Function
M1280	Disable external interruption I00□
M1281	Disable external interruption I10□
M1282	Disable external interruption I20□
M1283	Disable external interruption I30□
M1284	Disable external interruption I40□
M1285	Disable external interruption I50□
M1286	Disable time interruption I6□□
M1287	Disable time interruption I7□□
M1288	Disable time interruption I8□□
M1289	Disable high-speed counter interruption I010
M1290	Disable high-speed counter interruption I020
M1291	Disable high-speed counter interruption I030
M1292	Disable high-speed counter interruption I040
M1293	Disable high-speed counter interruption I050
M1294	Disable high-speed counter interruption I060
M1295	Disable pulse output interruption I110
M1296	Disable pulse output interruption I120
M1297	Disable pulse output interruption I130
M1298	Disable pulse output interruption I140
M1299	Disable communication interruption I150
M1300	Disable communication interruption I160
M1301	Disable communication interruption I170
M1302	Disable frequency measurement card interruption I180
M1340	Generate interruption I110 after CH0 pulse is sent
M1341	Generate interruption I120 after CH1 pulse is sent
M1342	Generate interruption I130 when CH0 pulse is being sent
M1343	Generate interruption I140 when CH1 pulse is being sent

API	Mnemonic	Function	Controllers
06	FEND	The End of The Main Program (First End)	ES/EX/SS | SA/SX/SC | EH/SV

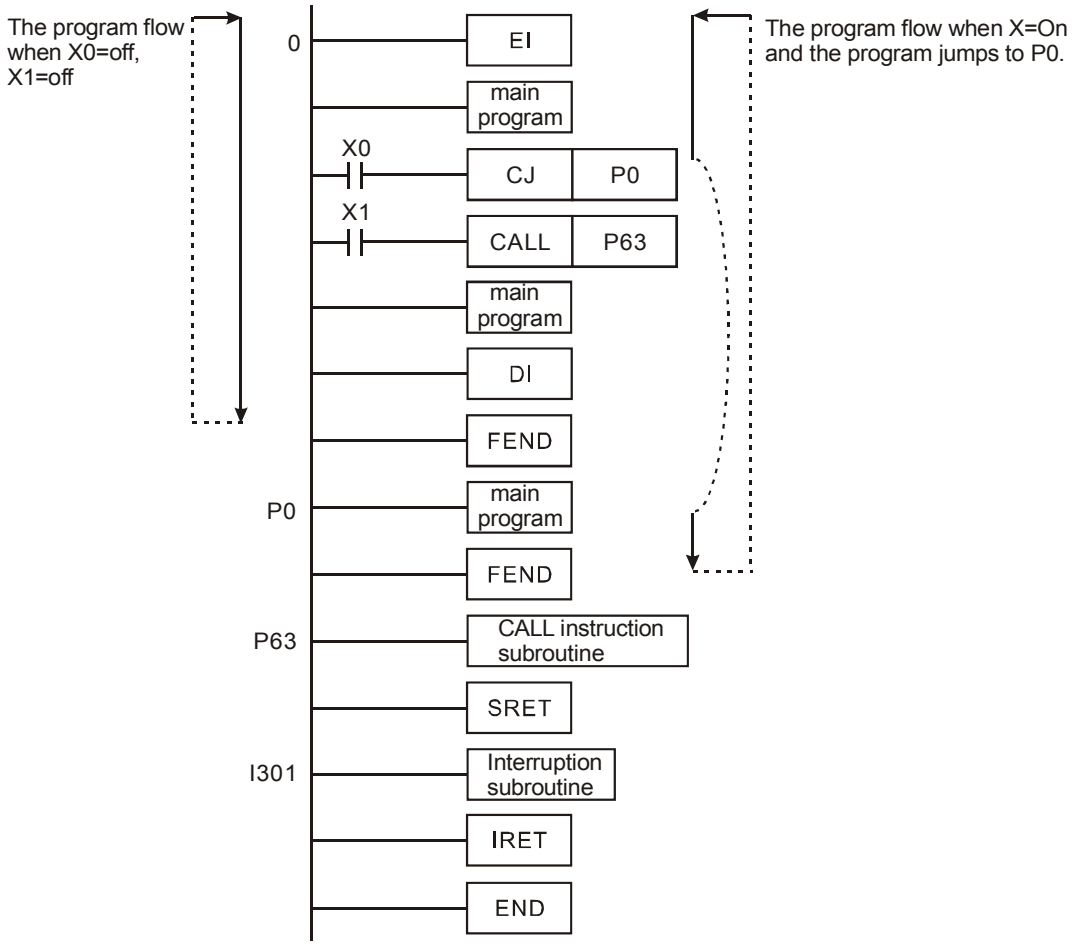
OP	Descriptions	Program Steps
N/A	No contact to drive the instruction is required.	FEND: 1 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

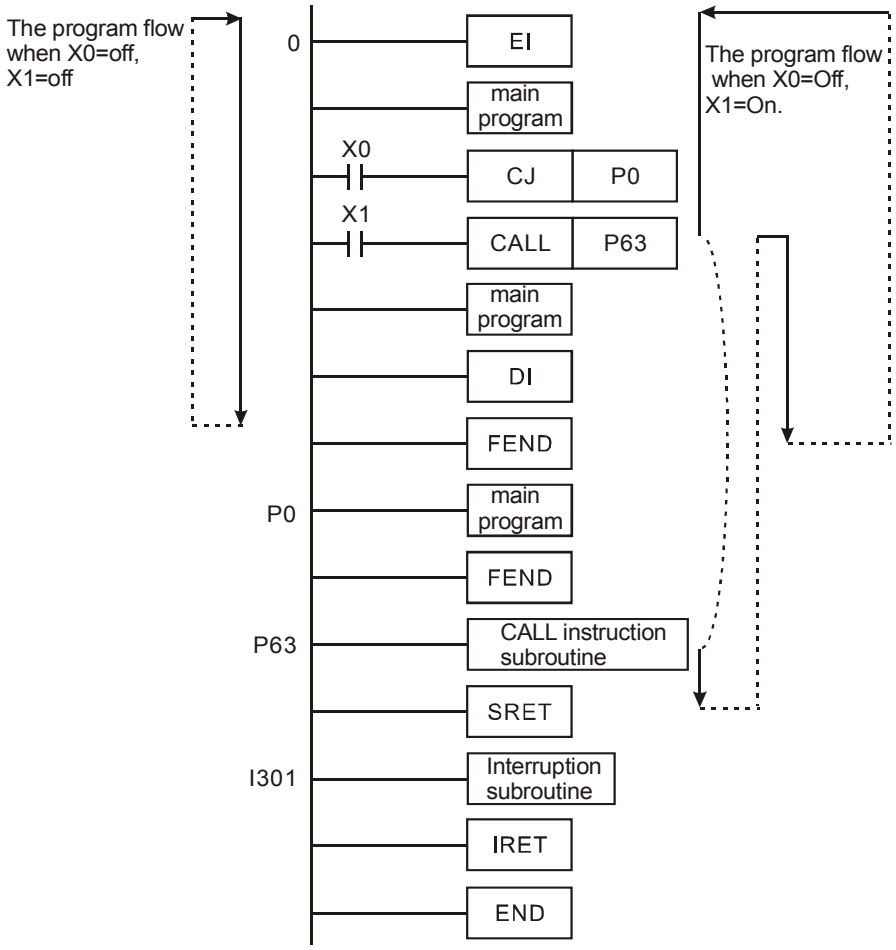
Explanations:

1. This instruction denotes the end of the main program. It has the same function as that of END instruction when being executed by PLC.
2. CALL must be written after FEND instruction and add SRET instruction in the end of its subroutine. Interruption program has to be written after FEND instruction and IRET must be added in the end of the service program.
3. If several FEND instructions are in use, place the subroutine and interruption service programs between the final FEND and END instruction.
4. After CALL instruction is executed, executing FEND before SRET will result in errors in the program.
5. After FOR instruction is executed, executing FEND before NEXT will result in errors in the program.

CJ Instruction Program Flow:



CALL Instruction Program Flow:



API	Mnemonic		Function	Controllers
07	WDT	P	Watchdog Timer Refresh	ES/EX/SS | SA/SX/SC | EH/SV

OP	Descriptions	Program Steps
N/A		WDT, WDTP: 1 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

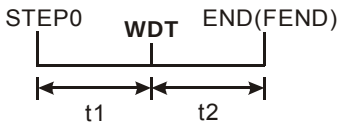
Explanations:

1. No operand.
2. The watchdog timer in DVP series PLCs is used for monitoring the operation of the PLC system.
3. WDT instruction can be used to reset Watch Dog Timer. If the PLC scan time (from step 0 to END or when FEND instruction is executed) exceeds 200ms, PLC ERROR LED will flash. The user will have to turn off PLC and back On again. PLC will determine RUN/STOP status by RUN/STOP switch. If there is no RUN/STOP switch, PLC will return to STOP status automatically.
4. When to use WDT:

a) When errors occur in the PLC system.

b) When the executing time of the program is too long, resulting in the scan time being larger than the content in D1000, the user can improve the problem by the following two methods.

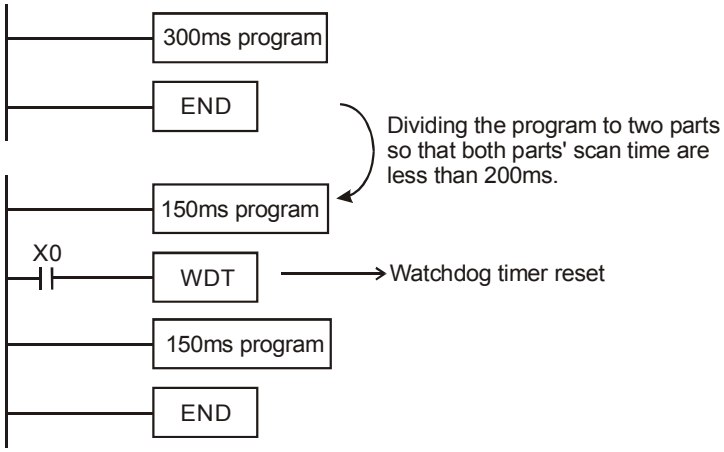
- Using WDT instruction



- Using the set value in D1000 (default value: 200ms) to change the time for watchdog.

Program Example:

Assume the scan time of the program is 300ms, divide the program into two parts and place WDT instruction in the middle of the two parts, making scan time of the first half and second half of the program being less than 200ms.



API	Mnemonic	Operands	Function	Controllers
08	FOR	(S)	Start of a FOR-NEXT Loop	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	FOR: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: The number of repeated nested loops

Explanations:

1. No contact to drive the instruction is required.
2. See the specifications of each model for their range of use.

API	Mnemonic	Function	Controllers
09	NEXT	End of a FOR-NEXT Loop	ES/EX/SS SA/SX/SC EH/SV

OP	Descriptions	Program Steps
N/A		NEXT: 1 steps

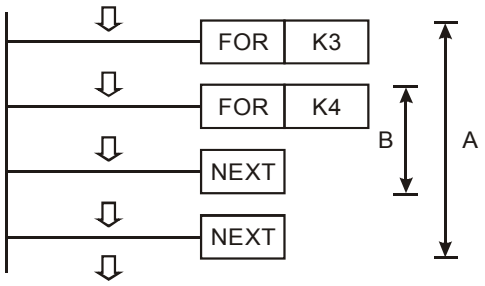
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Explanations:

1. No operand. No contact to drive the instruction is required.
2. FOR instruction indicates FOR ~ NEXT loops executing back and forth N times before escaping for the next execution.
3. N = K1 ~ K32,767. N is regarded as K1 when N ≤ 1.
4. When FOR~NEXT loops are not executed, the user can use the CJ instruction to escape the loops.
5. Error will occur when
   a) NEXT instruction is before FOR instruction.
   b) FOR instruction exists but NEXT instruction does not exist.
   c) There is NEXT instruction after FEND or END instruction.
   d) The number of instructions between FOR ~ NEXT differs.
6. FOR~NEXT loops can be nested for maximum five levels. Be careful that if there are too many loops, the increased PLC scan time may cause timeout of watchdog timer and error. Users can use WDT instruction to modify this problem.

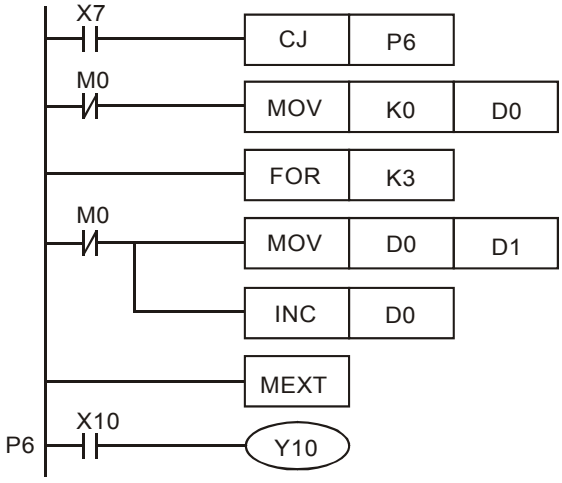
Program Example 1:

After program A has been executed for 3 times, it will resume its execution after NEXT instruction. Program B will be executed for 4 times whenever program A is executed once. Therefore, program B will be executed 3 × 4 = 12 times in total.



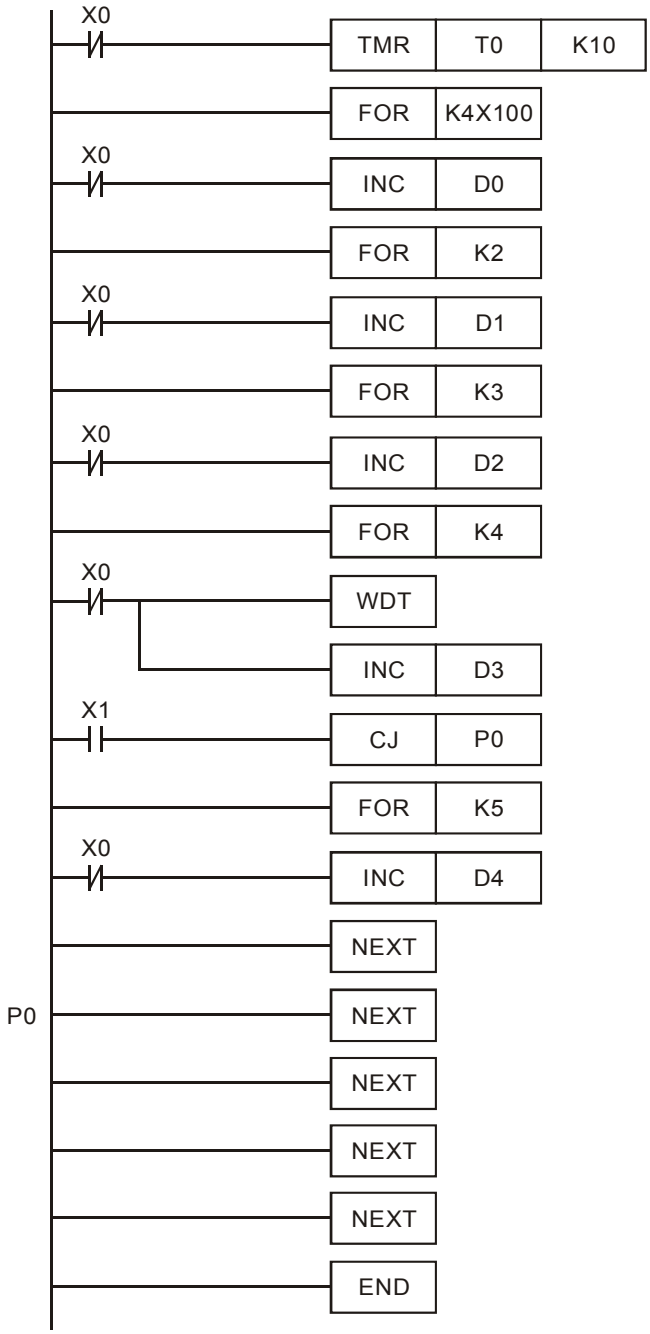
Program Example 2:

When X7 = Off, PLC will execute the program between FOR ~ NEXT. When X7 = On, CJ instruction jumps to P6 and avoids executing the programs between FOR ~ NEXT.



Program Example 3:

When the programs between FOR ~ NEXT are not to be executed, the user can adopt CJ instruction for a jumping. When the most inner FOR ~ NEXT loop is in the status of X1 = On, CJ instruction executes jumping to P0 and skips the execution on P0.



API	Mnemonic			Operands	Function	Controllers
10	D	CMP	P	(S1) (S2) (D)	Compare	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	CMP, CMPP: 7 steps DCMP, DCMPP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

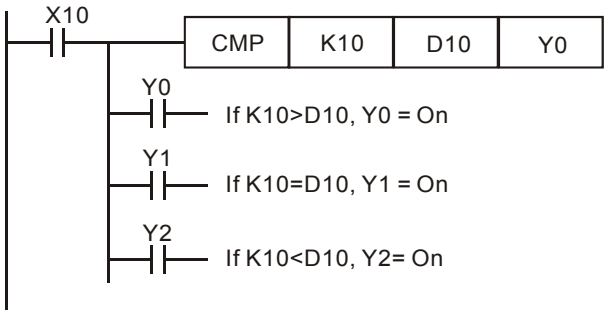
S_1: Comparison Value 1 **S_2**: Comparison Value 2 **D**: Comparison result

Explanations:

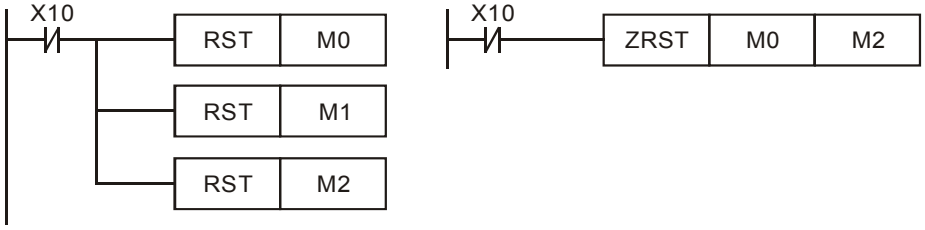
1. If **S_1** and **S_2** are used in device F, only 16-bit instruction is applicable.
2. Operand **D** occupies 3 consecutive devices.
3. See the specifications of each model for their range of use.
4. The contents in **S_1** and **S_2** are compared and the result will be stored in **D**.
5. The two comparison values are compared algebraically and the two values are signed binary values. When b15 = 1 in 16-bit instruction or b31 = 1 in 32-bit instruction, the comparison will regard the value as negative binary values.

Program Example:

1. Designate device Y0, and operand D automatically occupies Y0, Y1, and Y2.
2. When X10 = On, CMP instruction will be executed and one of Y0, Y1, and Y2 will be On. When X10 = Off, CMP instruction will not be executed and Y0, Y1, and Y2 remain their status before X10 = Off.
3. If the user need to obtain a comparison result with ≥ ≤, and ≠, make a series parallel connection between Y0 ~ Y2.



4. To clear the comparison result, use RST or ZRST instruction.



API	Mnemonic			Operands	Function	Controllers
11	D	ZCP	P	S_1 S_2 S D	Zone Compare	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	ZCP, ZCPP: 9 steps DZCP, DZCPP: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
S					*	*	*	*	*	*	*	*	*	*	*	
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Lower bound of zone comparison **S_2**: Upper bound of zone comparison **S**: Comparison value

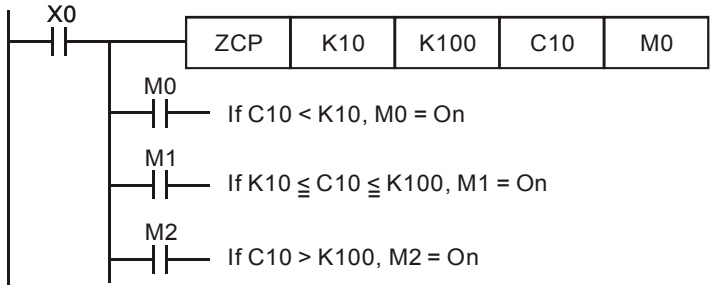
D: Comparison result

Explanations:

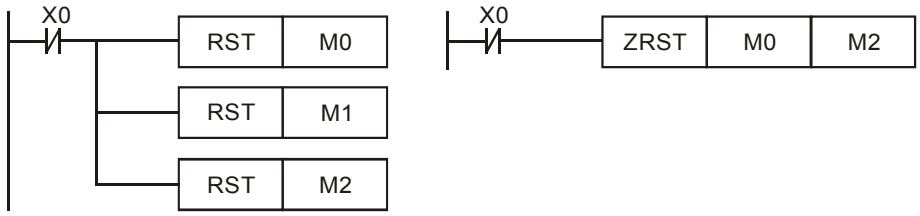
1. If **S_1**, **S_2** and **S** are used in device F, only 16-bit instruction is applicable.
2. The content in **S_1** should be smaller than the content in **S_2**.
3. Operand **D** occupies 3 consecutive devices.
4. See the specifications of each model for their range of use.
5. **S** is compared with its **S_1**, **S_2** and the result is stored in **D**.
6. When **S_1** > **S_2**, the instruction performs comparison by using **S_1** as the lower/upper bound.
7. The two comparison values are compared algebraically and the two values are signed binary values. When b15 = 1 in 16-bit instruction or b31 = 1 in 32-bit instruction, the comparison will regard the value as negative binary values.

Program Example:

1. Designate device M0, and operand D automatically occupies M0, M1 and M2.
2. When X0 = On, ZCP instruction will be executed and one of M0, M1, and M2 will be On. When X0 = Off, ZCP instruction will not be executed and M0, M1, and M2 remain their status before X0 = Off.



3. To clear the comparison result, use RST or ZRST instruction.



API	Mnemonic			Operands	Function	Controllers	
12	D	MOV	P	(S) (D)	Move	ES/EX/SS	SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	MOV, MOVP: 5 steps DMOV, DMOVP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

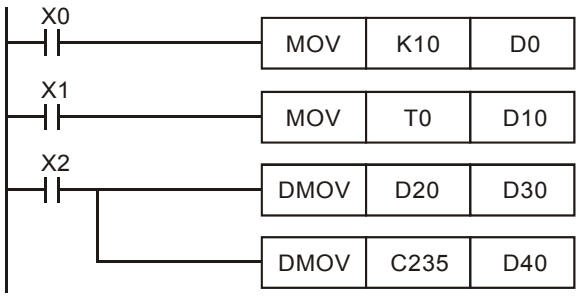
S: Source of data **D**: Destination of data

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. When this instruction is executed, the content of **S** will be moved directly to **D**. When this instruction is not executed, the content of **D** remains unchanged.
4. If the operation result refers to a 32-bit output, (i.e. application instruction MUL and so on), and the user needs to move the present value in the 32-bit high-speed counter, DMOV instruction has to be adopted.

Program Example:

1. MOV instruction has to be adopted in the moving of 16-bit data.
   a) When X0 = Off, the content in D10 will remain unchanged. If X0 = On, the value K10 will be moved to D10 data register.
   b) When X1 = Off, the content in D10 will remain unchanged. If X1 = On, the present value T0 will be moved to D10 data register.
2. DMOV instruction has to be adopted in the moving of 32-bit data.
   When X2 = Off, the content in (D31, D30) and (D41, D40) will remain unchanged. If X2 = On, the present value of (D21, D20) will be sent to (D31, D30) data register. Meanwhile, the present value of C235 will be moved to (D41, D40) data register.



API	Mnemonic		Operands	Function	Controllers
13	SMOV	P	S m1 m2 D n	Shift Move	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*	*	*	SMOV, SMOVP: 11 steps
m_1					*	*										
m_2					*	*										
D								*	*	*	*	*	*	*	*	
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

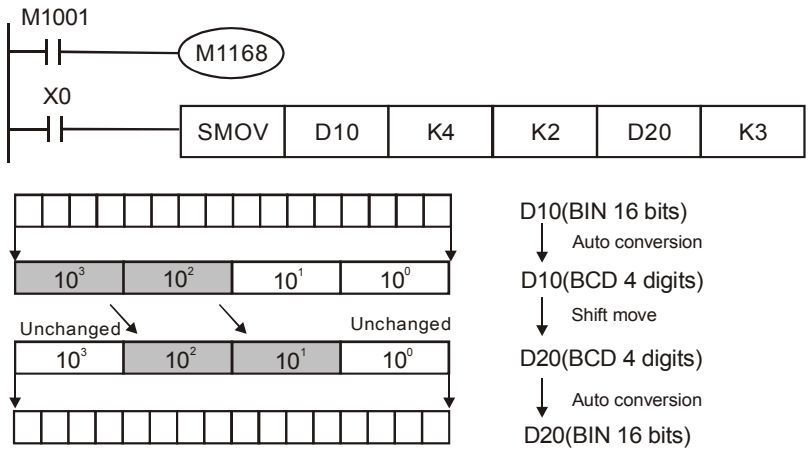
S: Source of data **m_1**: Start digit to be moved of the source data **m_2**: Number of digits (nibbles) to be moved of the source data **D**: Destination device **n**: Start digit of the destination position for the moved digits

Explanations:

1. This instruction is able to re-allocate or combine data. When the instruction is executed, **m_2** digits of contents starting from digit **m_1** (from high digit to low digit) of **S** will be sent to **m_2** digits starting from digit **n** (from high digit to low digit) of **D**.
2. Range: **m_1** = 1 ~ 4; **m_2** = 1 ~ **m_1**; **n** = **m_2** ~ 4
3. See the specifications of each model for their range of use.
4. M1168 is designated by SMOV working mode. When M1168 = On, the program is in BIN mode. When M1168 = Off, the program is in BCD mode.

Program Example 1:

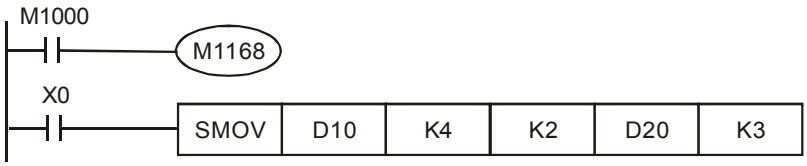
1. When M1168 = Off (in BCD mode) and X0 = On, the 4th (thousand) and 3rd (hundred) digit of the decimal value in D10 start to move to the 3rd (hundred) and 2nd (ten) digit of the decimal value in D20. 10^3 and 10^0 of D20 remain unchanged after this instruction is executed.
2. When the BCD value exceeds the range of 0 ~ 9,999, PLC will determine an operation error and will not execute the instruction. M1067, M1068 = On and D1067 records the error code OE18 (hex).

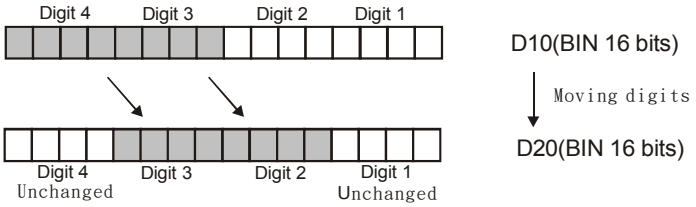


Before the execution, assume D10 = K1234 and D20 = K5678. After the execution, D10 will remain unchanged and D20 will become K5128.

Program Example 2:

When M1168 = On (in BIN mode) and SMOV instruction is in use, D10 and D20 will not be converted in BCD format but be moved in BIN format (4 digits as a unit).

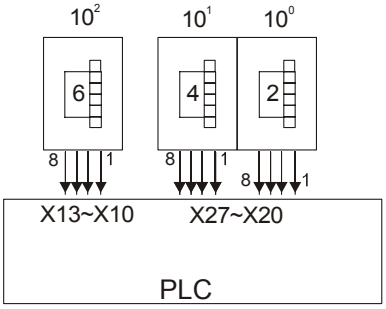


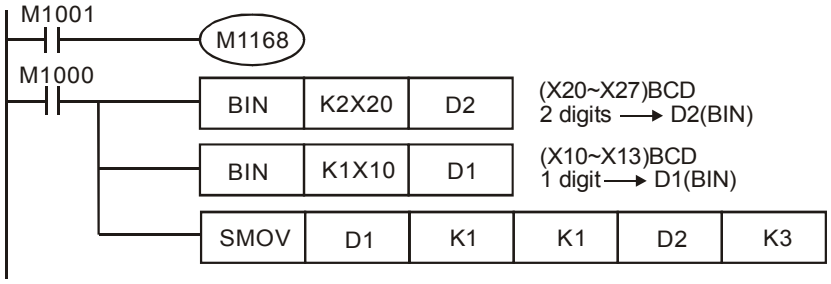


Before the execution, assume D10 = H1234 and D20 = H5678. After the execution, D10 will remain unchanged and D20 will become H5128.

Program Example 3:

1. This instruction can be used to combine the DIP switches connected to the input terminals with interrupted No.
2. Move the 2nd right digit of the DIP switch to the 2nd right digit of D2, and the 1st left digit of the DIP switch to the 1st right digit of D1.
3. Use SMOV instruction to move the 1st digit of D1 to the 3rd digit of D2 and combine the two DIP switches into one.





API	Mnemonic			Operands	Function	Controllers
14	D	CML	P	(S) (D)	Compliment	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	CML, CMLP: 5 steps DCML, DCMLP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

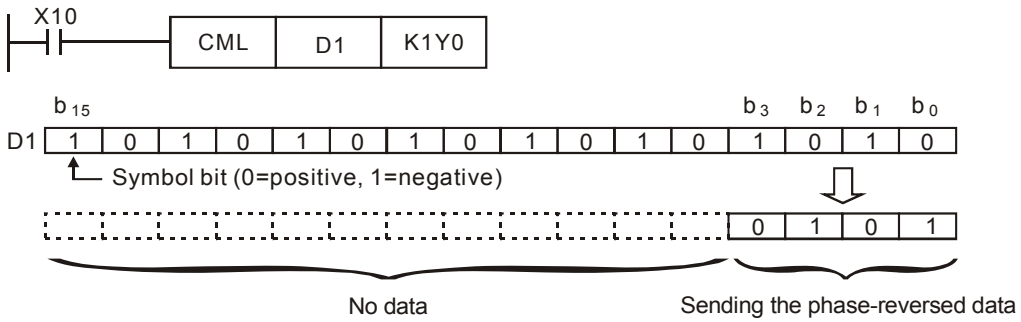
S: Source of data **D**: Destination device

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. This instruction can be used for phase-reversed output.
4. Reverse the phase (0→1, 1→0) of all the contents in **S** and send the contents to **D**. Given that the content is a constant K, K will be automatically converted into a BIN value.

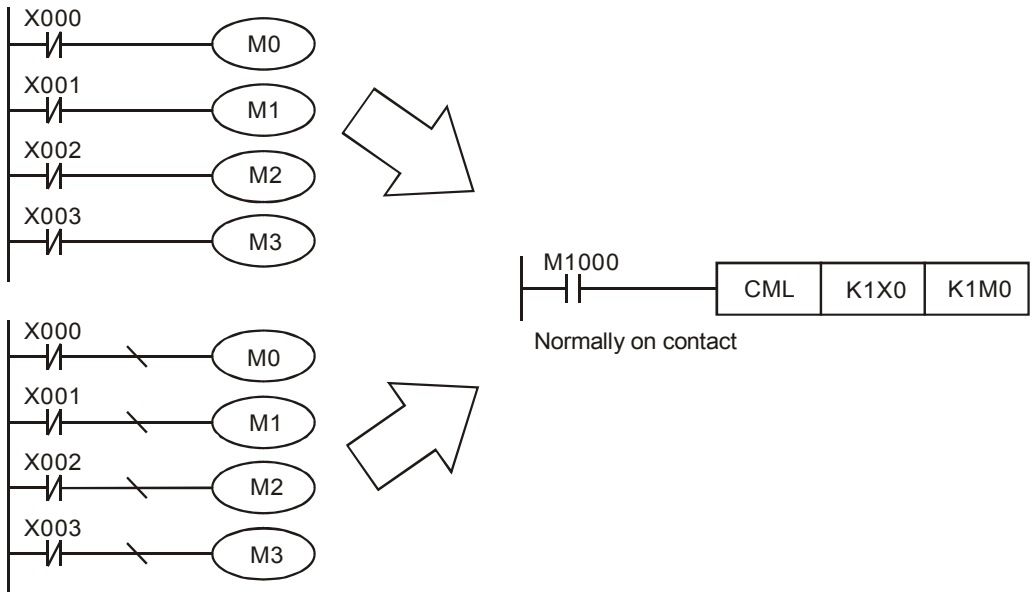
Program Example 1:

1. When X10 = On, b0 ~ b3 in D1 will be phase-reversed and send to Y0 ~ Y3.



Program Example 2:

The loop below can also adopt CML instruction (see right below).



API	Mnemonic		Operands	Function	Controllers	
15	BMOV	P	(S) (D) (n)	Block Move	ES/EX/SS	SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			BMOV, BMOVP: 7 steps
D								*	*	*	*	*	*			
n					*	*					*	*	*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

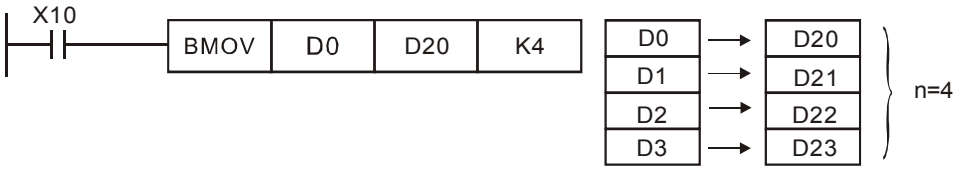
S: Start of source devices **D**: Start of destination devices **n**: Number of data to be moved

Explanations:

1. Range of **n**: 1 ~ 512
2. See the specifications of each model for their range of use.
3. The contents in n registers starting from the device designated by **S** will be moved to n registers starting from the device designated by **D**. If n exceeds the actual number of available source devices, only the devices that fall within the valid range will be used.

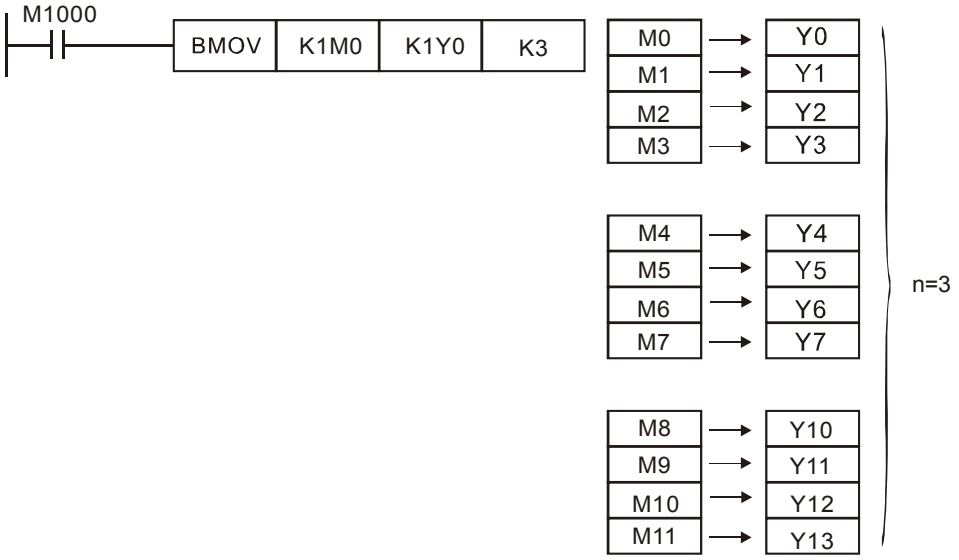
Program Example 1:

When X10 = On, the contents in registers D0 ~ D3 will be moved to the 4 registers D20 ~ D23.



Program Example 2:

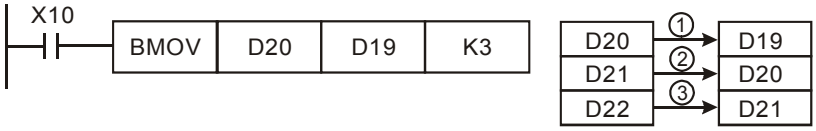
1. Assume the bit devices KnX, KnY, KnM and KnS are designated for moving, the number of digits of **S** and **D** has to be the same, i.e. their n has to be the same.
2. ES/EX/SS do not support the use of KnX, KnY, KnM, KnS and E, F index register modification.



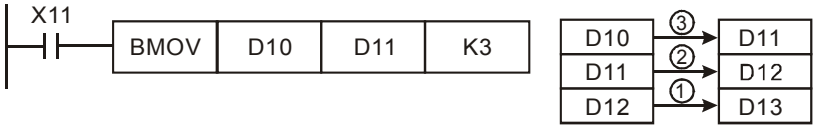
Program Example 3:

To avoid coincidence of the device numbers to be moved designated by the two operands and cause confusion, please be aware of the arrangement on the designated device numbers.

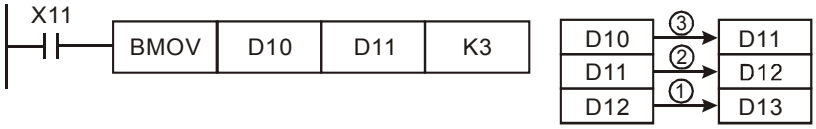
1. When **S** > **D**, the instruction is processed following the order ①→②→③



2. In EH/EH2/SV, when **S** < **D**, the instruction is processed following the order ①→②→③



3. In ESEX/SS/SA/SX/SC, when **S** < **D**, avoid the number difference of "1" and the instruction is processed following the order ③→②→①. If the devices have the number difference of "1", the contents in D11 ~ D13 will all be the content in D10.



API	Mnemonic			Operands	Function	Controllers
16	D	FMOV	P	(S) (D) (n)	Fill Move	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	FMOV, FMOVP: 7 steps DFMOV, DFMOVP: 13 steps
D								*	*	*	*	*	*			
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

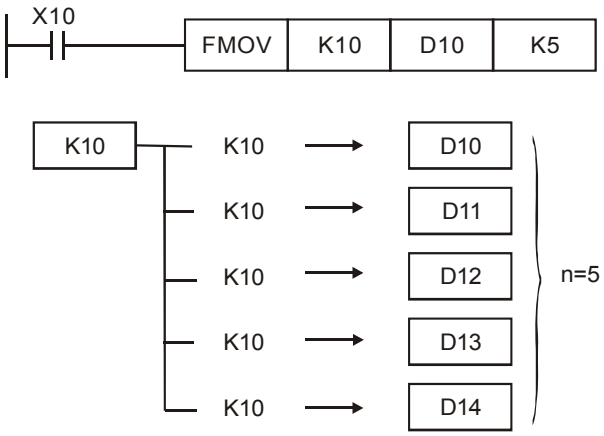
S: Source of data **D**: Destination of data **n**: Number of data to be moved

Explanations:

1. If **S** is used in device F, only 16-bit instruction is applicable.
2. Range of **n**: 1~ 512 (16-bit instruction); 1 ~ 256 (32-bit instruction).
3. See the specifications of each model for their range of use.
4. The contents in n registers starting from the device designated by **S** will be moved to n registers starting from the device designated by **D**. If n exceeds the actual number of available source devices, only the devices that fall within the valid range will be used.
5. ES/EX/SS do not support the use of KnX, KnY, KnM, KnS and E, F index register modification.

Program Example:

When X10 = On, K10 will be moved to the 5 consecutive registers starting from D10.



API	Mnemonic			Operands	Function	Controllers
17	D	XCH	P	D_1 D_2	Exchange	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D_1								*	*	*	*	*	*	*	*	XCH, XCHP: 5 steps DXCH, DXCHP: 9 steps
D_2								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

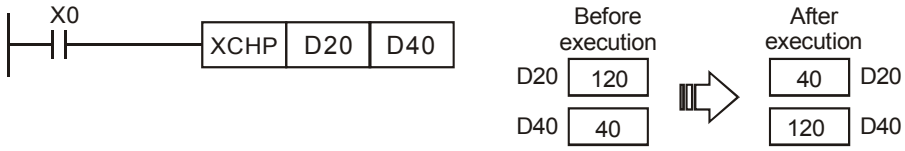
D_1: Data to be exchanged 1 **D_2**: Data to be exchanged 2

Explanations:

1. If **D_1** and **D_2** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. The contents in the devices designated by **D_1** and **D_2** will exchange.
4. Flag: M1303 (designated by XCH working mode).

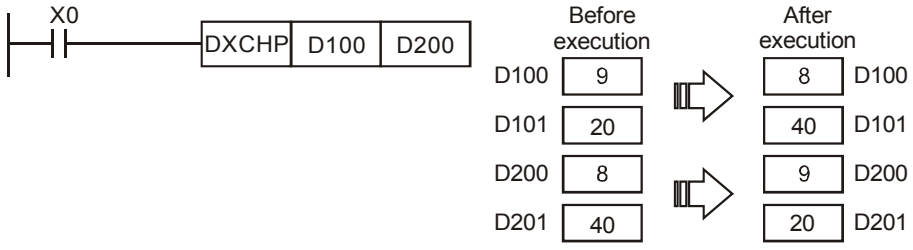
Program Example 1:

When X0 = Off→On, the contents in D20 and D40 exchange with each other.



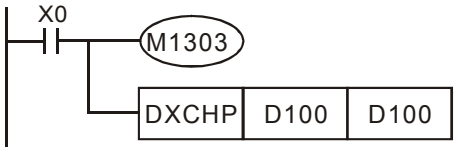
Program Example 2:

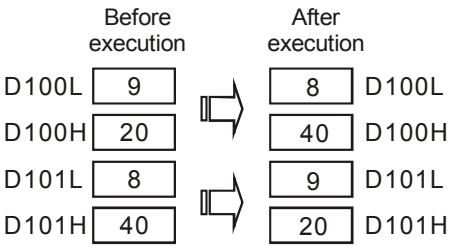
When X0 = Off → On, the contents in D100 and D200 exchange with each other.



Remarks:

1. ES/EX/SS do not support M1303.
2. As a 16-bit instruction, when the devices designated by **D_1** and **D_2** are the same and M1303 = On, the upper and lower 8 bits of the designated devices exchange with each other.
3. As a 32-bit instruction, when the devices designated by **D_1** and **D_2** are the same and M1303 = On, the upper and lower 16 bits in the individual designated device exchange with each other.
4. When X0 = On and M1303 = On, the 16-bit contents in D100 and those in D101 will exchange with each other.





API	Mnemonic	Operands	Function	Controllers
18	D BCD P	(S) (D)	Binary Coded Decimal	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*	*	*	BCD, BCDP: 5 steps DBCD, DBCDP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

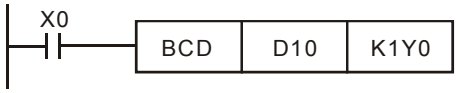
S: Source of data **D**: Conversion result

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. Flags: M1067 (operation error); M1068 (operation error); D1067 (error code)
4. The content in **S** (BIN value) is converted into BCD value and stored in **D**.
5. As a 16-bit (32-bit) instruction, when the conversion result exceeds the range of 0 ~ 9,999 (0 ~ 99,999,999), and M1067, M1068 = On, D1067 will record the error code 0E18 (hex).
6. The four arithmetic operations and applications in PLC and the execution of INC and DEC instructions are performed in BIN format. Therefore, if the user needs to see the decimal value display, simply use this instruction to convert the BIN value into BCD value.

Program Example:

1. When X0 = On, the binary value of D10 will be converted into BCD value, and the 1s digit of the conversion result will be stored in K1Y0 (Y0 ~ Y3, the 4 bit devices).



2. When D10 = 001E (hex) = 0030 (decimal), the execution result will be: Y0 ~ Y3 = 0000(BIN).

API	Mnemonic			Operands	Function	Controllers	
19	D	BIN	P	(S) (D)	Binary	ES/EX/SS	SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*	*	*	BIN, BINP: 5 steps DBIN, DBINP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

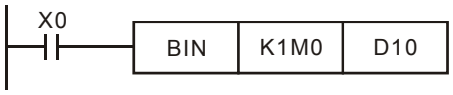
S: Source of data **D**: Conversion result

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. Flags: M1067 (operation error); M1068 (operation error); D1067 (error code)
4. The content in **S** (BCD value) is converted into BIN value and stored in **D**.
5. Valid range of **S** : BCD (0 ~ 9,999), DBCD (0 ~ 99,999,999)
6. Provided the content in S is not a BCD value (in hex and any one of its digits does not fall in the range of 0 ~ 9), an operation error will occur. M1067, M1068 = On and D1067 records the error code 0E18 (hex).
7. Constant K and H will automatically be converted into BIN format. Thus, they do not need to adopt this instruction.

Program Example:

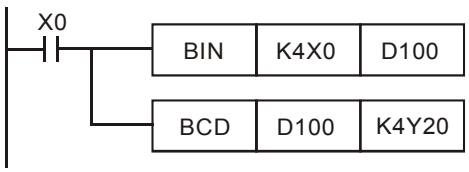
When X0 = On, the BCD value of K1M0 will be converted to BIN value and stored in D10.

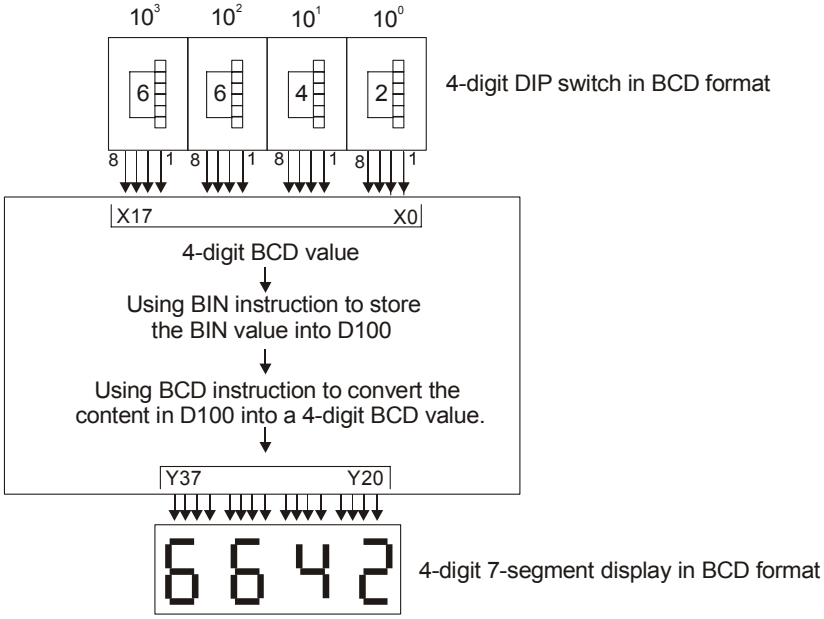


Remarks:

Explanations on BCD and BIN instructions:

1. When PLC needs to read an external DIP switch in BCD format, BIN instruction has to be first adopted to convert the read data into BIN value and store the data in PLC.
2. When PLC needs to display its stored data by a 7-segment display in BCD format, BCD instruction has to be first adopted to convert the data into BCD value and send the data to the 7-segment display.
3. When X0 = On, the BCD value of K4X0 is converted into BIN value and sent it to D100. The BIN value of D100 will then be converted into BCD value and sent to K4Y20.





API	Mnemonic			Operands	Function	Controllers
20	D	ADD	P	(S1) (S2) (D)	Addition	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	ADD, ADDP: 7 steps DADD, DADDP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Summand **S_2**: Addend **D**: Sum

Explanations:

1. If **S_1**, **S_2** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
4. This instruction adds **S_1** and **S_2** in BIN format and store the result in **D**.
5. The highest bit is symbolic bit 0 (+) and 1 (-), which is suitable for algebraic addition, e.g. 3 + (-9) = -6.
6. Flag changes in binary addition

   In 16-bit BIN addition,

   a) If the operation result = 0, zero flag M1020 = On.

   b) If the operation result < -32,768, borrow flag M1021 = On.

   c) If the operation result > 32,767, carry flag M1022 = On.

   In 32-bit BIN addition,

   a) If the operation result = 0, zero flag M1020 = On.

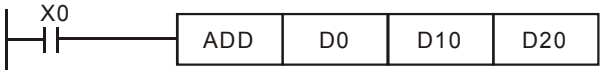
   b) If the operation result < -2,147,483,648, borrow flag M1021 = On.

   c) If the operation result > 2,147,483,647, carry flag M1022 = On.

Program Example 1:

In 16-bit BIN addition:

When X0 = On, the content in D0 will plus the content in D10 and the sum will be stored in D20.

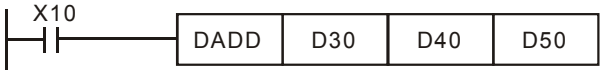


Program Example 2:

In 32-bit BIN addition:

When X0 = On, the content in (D31, D30) will plus the content in (D41, D40) and the sum will be stored in (D51, D50).

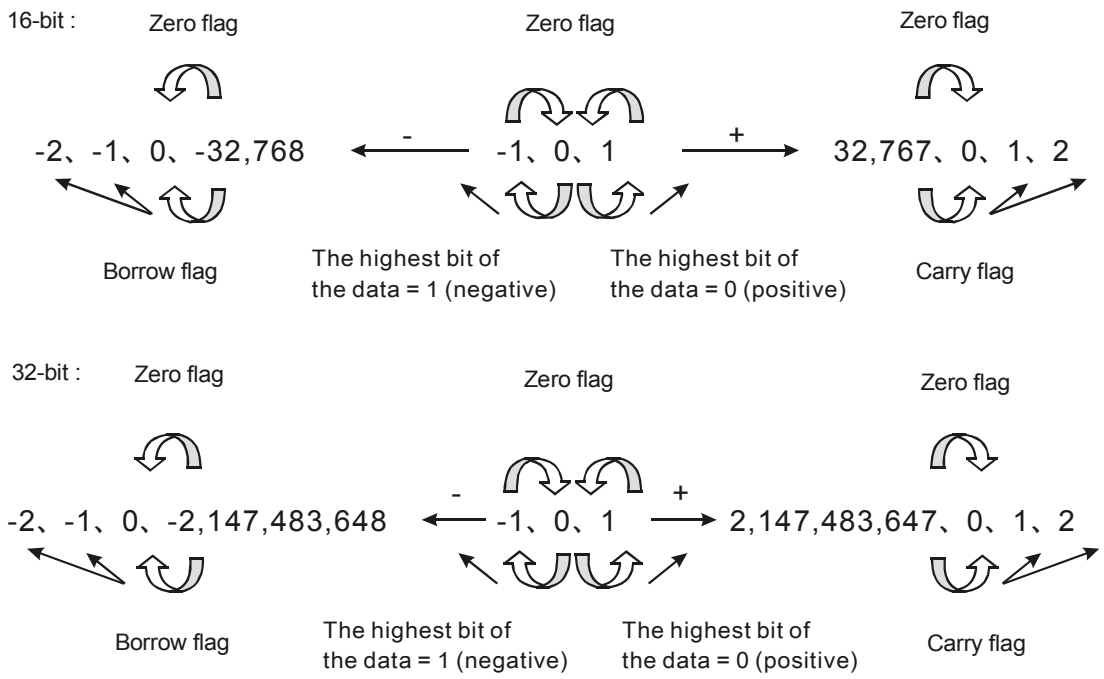
D30, D40 and D50 are low 16-bit data; D31, D41 and D51 are high 16-bit data.



(D31, D30) + (D41, D40) = (D51, D50)

Remarks:

Flags and the positive/negative sign of the values:



API	Mnemonic			Operands	Function	Controllers
21	D	SUB	P	(S1) (S2) (D)	Subtraction	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	SUB, SUBP: 7 steps DSUB, DSUBP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Minuend **S_2**: Subtrahend **D**: Remainder

Explanations:

1. If **S_1**, **S_2** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
4. This instruction subtracts **S_1** and **S_2** in BIN format and stores the result in **D**.
5. The highest bit is symbolic bit 0 (+) and 1 (-), which is suitable for algebraic subtraction.
6. Flag changes in binary subtraction

   In 16-bit instruction:

   a) If the operation result = 0, zero flag M1020 = On.

   b) If the operation result < –32,768, borrow flag M1021 = On.

   c) If the operation result > 32,767, carry flag M1022 = On.

   In 32-bit instruction:

   a) If the operation result = 0, zero flag M1020 = On.

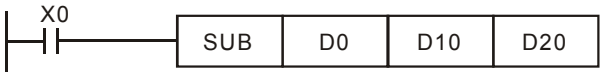
   b) If the operation result < –2,147,483,648, borrow flag M1021 = On.

   c) If the operation result > 2,147,483,647, carry flag M1022 = On.
7. For flag operations of SUB instruction and the positive/negative sign of the value, see the explanations in ADD instruction on the previous page.

Program Example 1:

In 16-bit BIN subtraction:

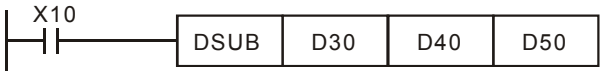
When X0 = On, the content in D0 will minus the content in D10 and the remainder will be stored in D20.



Program Example 2:

In 32-bit BIN subtraction:

When X10 = On, the content in (D31, D30) will minus the content in (D41, D40) and the remainder will be stored in (D51, D50). D30, D40 and D50 are low 16-bit data; D31, D41 and D51 are high 16-bit data.



(D31, D30) – (D41, D40) = (D51, D50)

API	Mnemonic			Operands	Function	Controllers
22	D	MUL	P	S1 S2 D	Multiplication	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	MUL, DMULP: 7 steps DMUL, DMULP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*		

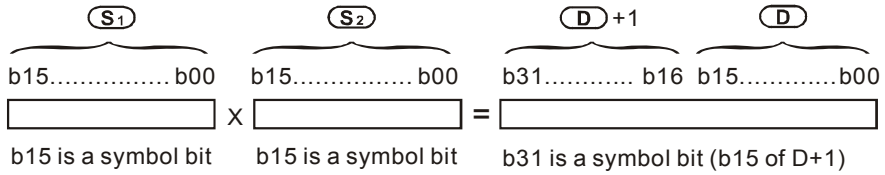
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Multiplicand **S_2**: Multiplicator **D**: Product

Explanations:

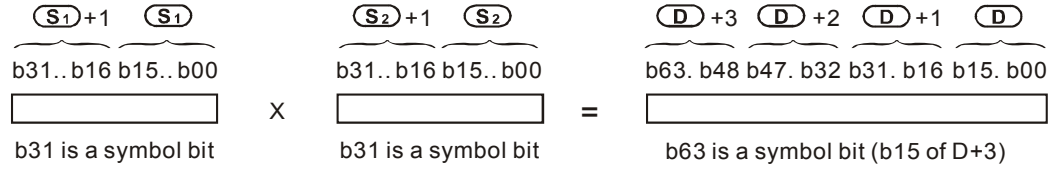
1. If **S_1** and **S_2** are used in device F, only 16-bit instruction is applicable.
2. If **D** is used in device E, only 16-bit instruction is applicable.
3. In 16-bit instruction, **D** occupies 2 consecutive devices.
4. In 32-bit instruction, **D** occupies 4 consecutive devices.
5. See the specifications of each model for their range of use.
6. This instruction multiplies **S_1** by **S_2** in BIN format and stores the result in **D**. Be careful with the positive/negative signs of **S_1**, **S_2** and **D** when doing 16-bit and 32-bit operations.
7. In 16-bit BIN multiplication,



Symbol bit = 0 refers to a positive value.
Symbol bit = 1 refers to a negative value.

When D serves as a bit device, it can designate K1 ~ K4 and construct a 16-bit result, occupying consecutive 2 groups of 16-bit data. ES/EX/SS only stores low 16-bit data.

8. 32-bit BIN multiplication,



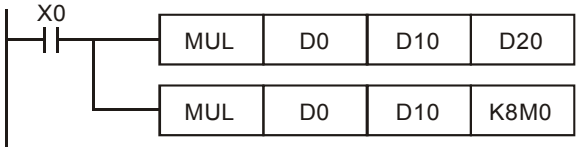
Symbol bit = 0 refers to a positive value.
Symbol bit = 1 refers to a negative value.

When D serves as a bit device, it can designate K1 ~ K8 and construct a 32-bit result, occupying consecutive 2 groups of 32-bit data.

Program Example:

The 16-bit D0 is multiplied by the 16-bit D10 and brings forth a 32-bit product. The higher 16 bits are stored in D21

and the lower 16-bit are stored in D20. On/Off of the most left bit indicates the positive/negative status of the result value.



API	Mnemonic			Operands	Function	Controllers
23	D	DIV	P	(S1) (S2) (D)	Division	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*		DIV, DIVP: 7 steps DDIV, DDIVP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*		
D								*	*	*	*	*	*	*		

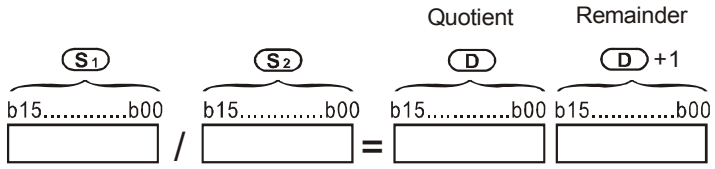
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Dividend **S_2**: Divisor **D**: Quotient and remainder

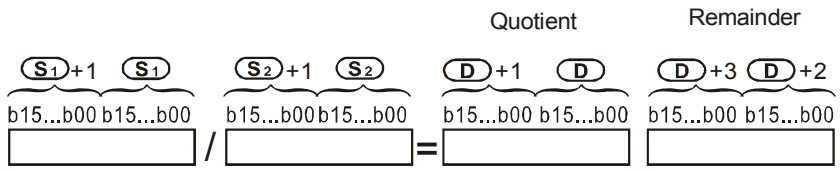
Explanations:

1. If **S_1** and **S_2** are used in device F, only 16-bit instruction is applicable.
2. If **D** is used in device E, only 16-bit instruction is applicable.
3. In 16-bit instruction, **D** occupies 2 consecutive devices.
4. In 32-bit instruction, **D** occupies 4 consecutive devices.
5. See the specifications of each model for their range of use.
6. This instruction divides **S_1** and **S_2** in BIN format and stores the result in **D**. Be careful with the positive/negative signs of **S_1**, **S_2** and **D** when doing 16-bit and 32-bit operations.
7. This instruction will not be executed when the divisor is 0. M1067 and M1068 will be On and D1067 records the error code 0E19 (hex).
8. In 16-bit BIN division,



When D serves as a bit device, it can designate K1 ~ K4 and construct a 16-bit result, occupying consecutive 2 groups of 16-bit data and bringing forth the quotient and remainder. ES/EX/SS is able to bring forth only quotient without the remainder.

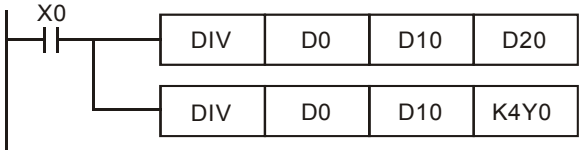
9. In 32-bit BIN division,



When D serves as a bit device, it can designate K1 ~ K8 and construct a 32-bit result, occupying consecutive 2 groups of 32-bit data and bringing forth the quotient and remainder.

Program Example:

When X0 = On, D0 will be divided by D10 and the quotient will be stored in D20 and remainder in D21. On/Off of the highest bit indicates the positive/negative status of the result value.



API	Mnemonic			Operands	Function	Controllers	
24	D	INC	P	(D)	Increment	ES/EX/SS	SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	INC, INCP: 3 steps DINC, DINCP: 5 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Destination device

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. If the instruction is not a pulse execution one, the content in the designated device D will plus “1” in every scan period whenever the instruction is executed.
4. This instruction adopts pulse execution instructions (INCP, DINCP).
5. In 16-bit operation, 32,767 pluses 1 and obtains -32,768. In 32-bit operation, 2,147,483,647 pluses 1 and obtains -2,147,483,648.
6. The operation results will not affect M1020 ~ M1022.

Program Example:

When X0 = Off→On, the content in D0 pluses 1 automatically.

API	Mnemonic			Operands	Function	Controllers	
25	D	DEC	P	(D)	Decrement	ES/EX/SS	SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	DEC, DECP: 3 steps DDEC, DDECP: 5 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Destination device

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. If the instruction is not a pulse execution one, the content in the designated device D will minus “1” in every scan period whenever the instruction is executed.
4. This instruction adopts pulse execution instructions (DECP, DDECP).
5. In 16-bit operation, -32,768 minuses 1 and obtains 32,767. In 32-bit operation, -2,147,483,648 minuses 1 and obtains 2,147,483,647.
6. The operation results will not affect M1020 ~ M1022.

Program Example:

When X0 = Off→On, the content in D0 minuses 1 automatically.

API	Mnemonic			Operands	Function	Controllers
26	W/D	AND	P	S_1 S_2 D	Logical Word AND	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	WAND, WANDP: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DAND, DANDP: 13 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

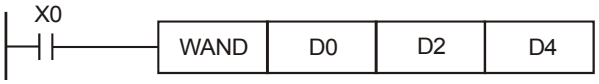
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

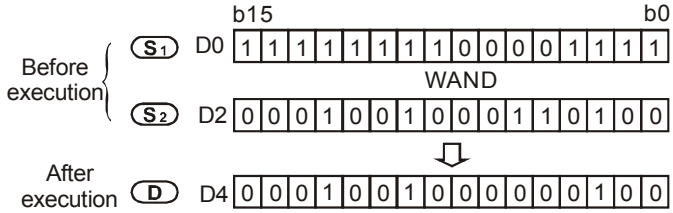
Explanations:

1. If **S_1**, **S_2** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. This instruction conducts logical AND operation of **S_1** and **S_2** and stores the result in **D**.
4. Operation rule: The corresponding bit of the operation result in **D** will be "0" if any of the bits in **S_1** or **S_2** is "0".

Program Example 1:

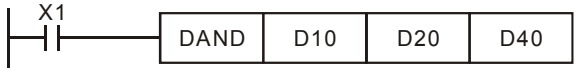
When X0 = On, the 16-bit D0 and D2 will perform WAND, logical AND operation, and the result will be stored in D4.

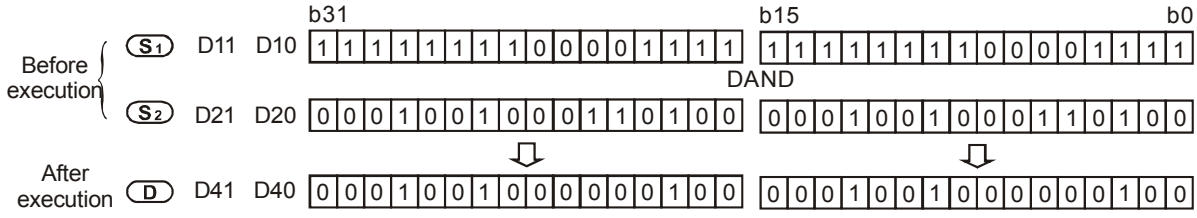




Program Example 2:

When X1 = On, the 32-bit (D11, D10) and (D21, D20) will perform DAND, logical AND operation, and the result will be stored in (D41, D40).





API	Mnemonic			Operands	Function	Controllers	
27	W D	OR	P	(S1) (S2) (D)	Logical Word OR	ES/EX/SS	SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	WOR, WORP: 7 steps DOR, DORP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

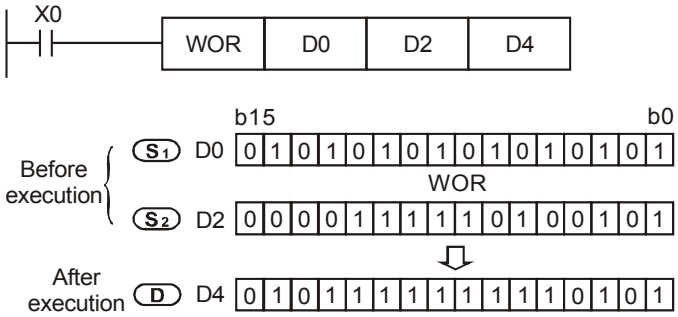
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

Explanations:

1. If **S_1**, **S_2** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. This instruction conducts logical OR operation of **S_1** and **S_2** and stores the result in **D**.
4. Operation rule: The corresponding bit of the operation result in **D** will be "1" if any of the bits in **S_1** or **S_2** is "1".

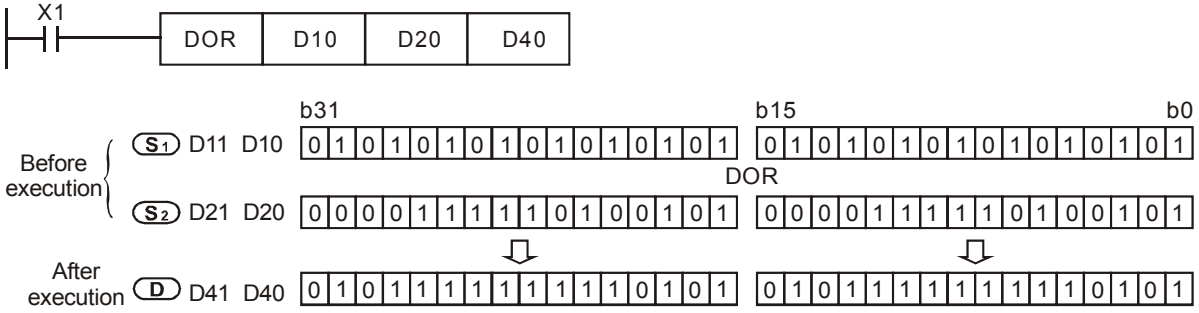
Program Example 1:

When X0 = On, the 16-bit D0 and D2 will perform WOR, logical OR operation, and the result will be stored in D4.



Program Example 2:

When X1 = On, the 32-bit (D11, D10) and (D21, D20) will perform DOR, logical OR operation, and the result will be stored in (D41, D40).



API	Mnemonic			Operands	Function	Controllers
28	W/D	XOR	P	S_1 S_2 D	Logical Exclusive OR	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	WXOR, WXORP: 7 steps DXOR, DXORP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

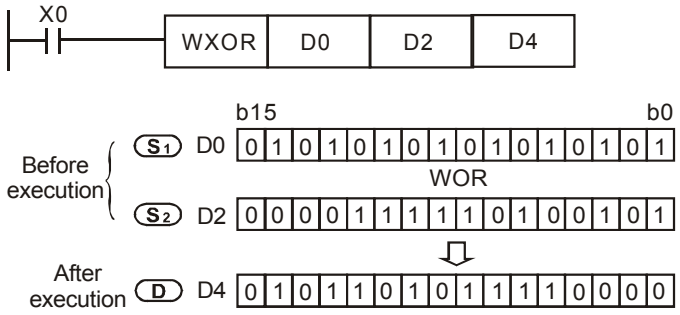
S_1: Source data device 1 **S_2**: Source data device 2 **D**: Operation result

Explanations:

1. If **S_1**, **S_2** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. This instruction conducts logical XOR operation of **S_1** and **S_2** and stores the result in **D**.
4. Operation rule: If the bits in **S_1** and **S_2** are the same, the corresponding bit of the operation result in **D** will be "0"; if the bits in **S_1** and **S_2** are different, the corresponding bit of the operation result in **D** will be "1".

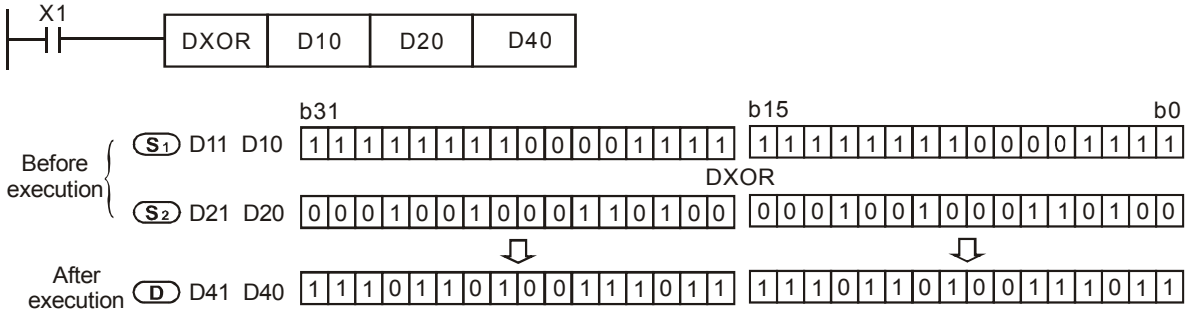
Program Example 1:

When X0 = On, the 16-bit D0 and D2 will perform WXOR, logical XOR operation, and the result will be stored in D4.



Program Example 2:

When X1 = On, the 32-bit (D11, D10) and (D21, D20) will perform DXOR, logical XOR operation, and the result will be stored in (D41, D40).



API	Mnemonic			Operands	Function	Controllers
29	D	NEG	P	(D)	2's Complement (Negative)	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	NEG, NEGP: 3 steps DNEG, DNEGP: 5 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

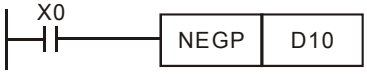
D: Device to store 2's complement

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. This instruction converts a negative BIN value into an absolute value.
4. This instruction adopts pulse execution instructions (NEGP, DNEGP).

Program Example 1:

When X0 = Off→On, the phase of every bit of the content in D10 will be reversed (0→1, 1→0) and pluses 1. The result will then be stored in D10.

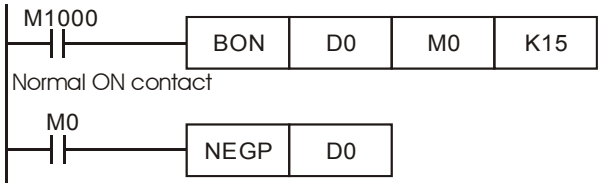


Program Example 2:

Obtaining the absolute value of a negative value:

a) When the 15th bit of D0 is "1", M0 = On. (D0 is a negative value).

b) When M0 = Off→On, NEG instruction will obtain 2's complement of D0 and further its absolute value.



Program Example 3:

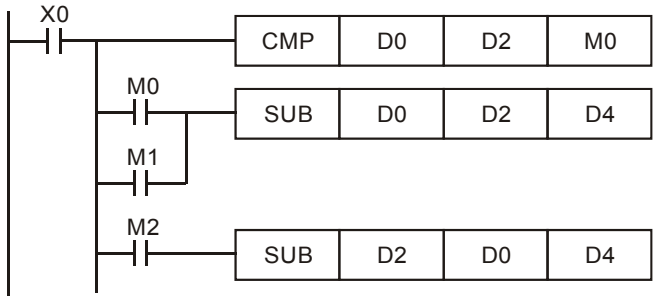
Obtaining the absolute value by the remainder of the subtraction. When X0 = On,

a) If D0 > D2, M0 = On.

b) If D0 = D2, M1 = On.

c) If D0 < D2, M2 = On.

d) D4 is then able to remain positive.

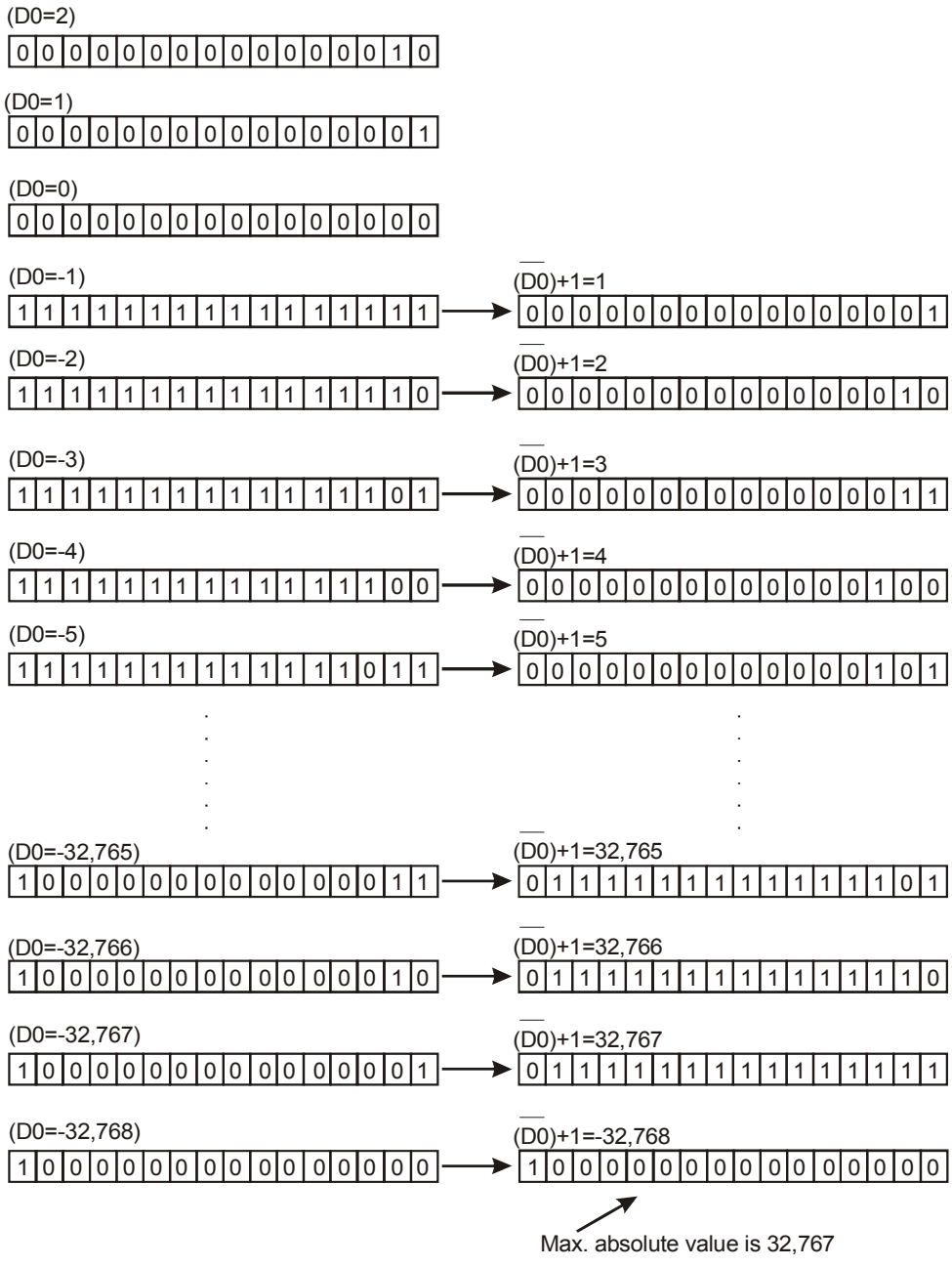


Remarks:

Negative value and its absolute value

a) The sign of a value is indicated by the highest (most left) bit in the register. 0 indicates that the value is a positive one and 1 indicates that the value is a negative one.

b) NEG instruction is able to convert a negative value into its absolute value.



API	Mnemonic			Operands	Function	Controllers
30	D	ROR	P	(D) (n)	Rotation Right	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	ROR, RORP: 5 steps DROR, DRORP: 9 steps
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

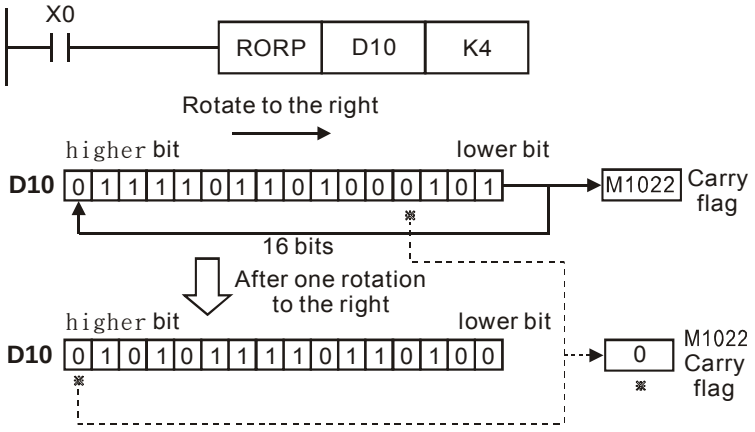
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.
3. Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)
4. See the specifications of each model for their range of use.
5. Flag: M1022 (carry flag)
6. This instruction rotates the device content designated by **D** to the right for **n** bits.
7. This instruction adopts pulse execution instructions (RORP, DRORP).

Program Example:

When X0 = Off→On, the 16 bits (4 bits as a group) in D10 will rotate to the right, as shown in the figure below. The bit marked with ※ will be sent to carry flag M1022.



API	Mnemonic			Operands	Function	Controllers
31	D	ROL	P	(D) (n)	Rotation Left	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	ROL, ROLP: 5 steps DROL, DROLP: 9 steps
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

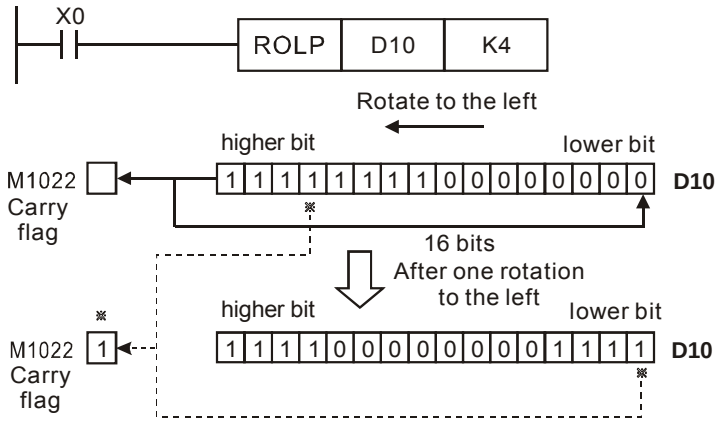
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.
3. Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)
4. See the specifications of each model for their range of use.
5. Flag: M1022 (carry flag)
6. This instruction rotates the device content designated by **D** to the left for **n** bits.
7. This instruction adopts pulse execution instructions (ROLP, DROLP).

Program Example:

When X0 = Off→On, the 16 bits (4 bits as a group) in D10 will rotate to the left, as shown in the figure below. The bit marked with ※ will be sent to carry flag M1022.



API	Mnemonic			Operands	Function	Controllers	
32	D	RCR	P	(D) (n)	Rotation Right with Carry	ES/EX/SS	SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	RCR, RCRP: 5 steps DRCR, DRCRP: 9 steps
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

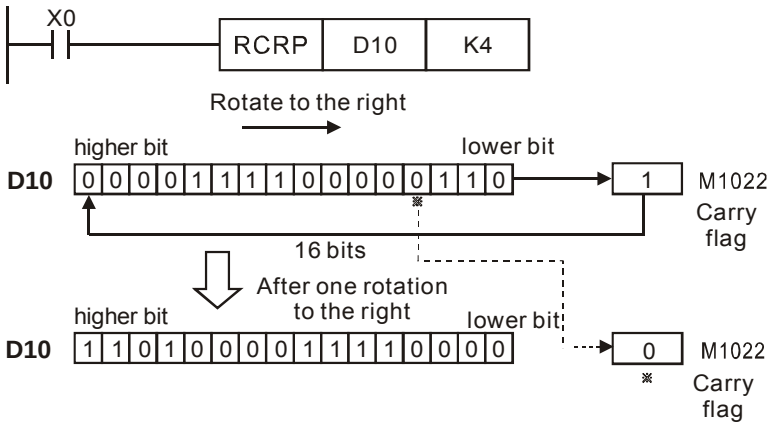
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.
3. Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)
4. See the specifications of each model for their range of use.
5. Flag: M1022 (carry flag)
6. This instruction rotates the device content designated by **D** together with carry flag M1022 to the right for **n** bits.
7. This instruction adopts pulse execution instructions (RCRP, DRCRP).

Program Example:

When X0 = Off→On, the 16 bits (4 bits as a group) in D10 together with carry flag M1022 (total 17 bits) will rotate to the right, as shown in the figure below. The bit marked with ※ will be sent to carry flag M1022.



API	Mnemonic			Operands	Function	Controllers
33	D	RCL	P	(D) (n)	Rotation Left with Carry	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	RCL, RCLP: 5 steps DRCL, DRCLP: 9 steps
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

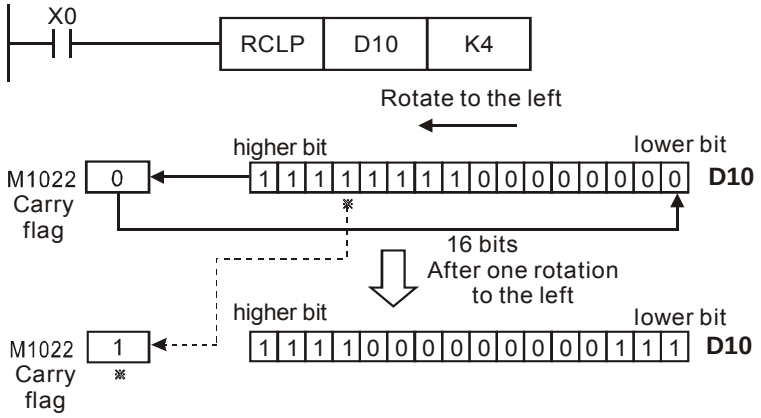
D: Device to be rotated **n**: Number of bits to be rotated in 1 rotation

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. If **D** is designated as KnY, KnM, and KnS, only K4 (16-bit) and K8 (32-bit) are valid.
3. Range of **n**: K1 ~ K16 (16-bit); K1 ~ K32 (32-bit)
4. See the specifications of each model for their range of use.
5. Flag: M1022 (carry flag)
6. This instruction rotates the device content designated by **D** together with carry flag M1022 to the left for **n** bits.
7. This instruction adopts pulse execution instructions (RCLP, DRCLP).

Program Example :

When X0 = Off→On, the 16 bits (4 bits as a group) in D10 together with carry flag M1022 (total 17 bits) will rotate to the left, as shown in the figure below. The bit marked with ※ will be sent to carry flag M1022.



API	Mnemonic	Operands	Function	Controllers
34	SFTR P	S D n_1 n_2	Bit Shift Right	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*	*												SFTR, SFTRP: 9 steps
D		*	*	*												
n_1					*	*										
n_2					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start No. of the shifted device **D**: Start No. of the device to be shifted n_1: Length of data to be shifted

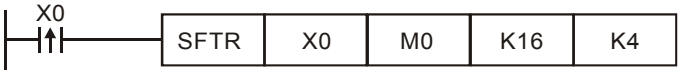
n_2: Number of bits to be shifted in 1 shift

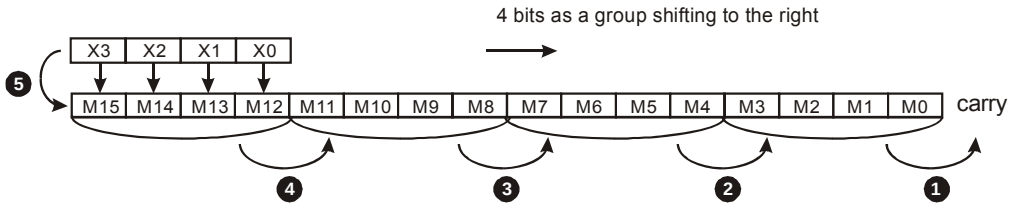
Explanations:

1. Range of n_1: 1~ 1,024
2. Range of n_2: 1 ~ n_1
3. In ES/EX/SS, $1 \le n_2 \le n_1 \le 512$
4. ES/EX/SS series MPU does not support E, F index register modification.
5. See the specifications of each model for their range of use.
6. This instruction shifts the bit device of n_1 bits (desired length for shifted register) starting from **D** to the right for n_2 bits. **S** is shifted into **D** for n_2 bits to supplement empty bits.
7. This instruction adopts pulse execution instructions (SFTRP).

Program Example:

1. When X0 = Off→On, M0 ~M15 will form 16 bits and shifts to the right (4 bits as a group).
2. The figure below illustrates the right shift of the bits in one scan.
   - ❶ M3 ~ M0 → carry
   - ❷ M7 ~ M4 → M3 ~ M0
   - ❸ M11 ~ M8 → M7 ~ M4
   - ❹ M15 ~ M12 → M11 ~ M8
   - ❺ X3 ~ X0 → M15 ~ M12 completed





API	Mnemonic		Operands	Function	Controllers
35	SFTL	P	(S) (D) (n1) (n2)	Bit Shift Left	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*	*												SFTL, SFTLP: 9 steps
D		*	*	*												
n_1					*	*										
n_2					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start No. of the shifted device **D**: Start No. of the device to be shifted $\mathbf{n_1}$: Length of data to be shifted

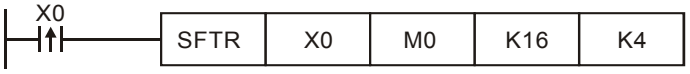
$\mathbf{n_2}$: Number of bits to be shifted in 1 shift

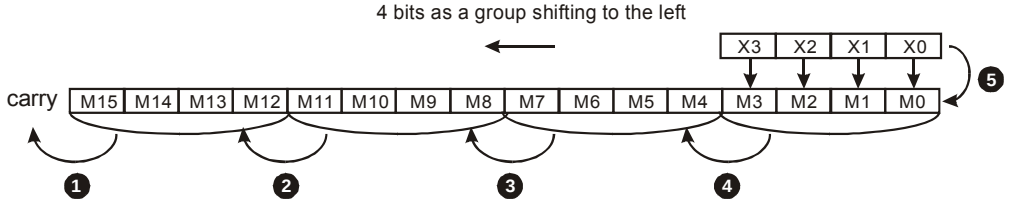
Explanations:

1. Range of $\mathbf{n_1}$: 1~ 1,024
2. Range of $\mathbf{n_2}$: 1 ~ $\mathbf{n_1}$
3. In ES/EX/SS, $1 \le \mathbf{n_2} \le \mathbf{n_1} \le 512$
4. ES/EX/SS series MPU does not support E, F index register modification.
5. See the specifications of each model for their range of use.
6. This instruction shifts the bit device of $\mathbf{n_1}$ bits (desired length for shifted register) starting from **D** to the left for $\mathbf{n_2}$ bits. **S** is shifted into **D** for $\mathbf{n_2}$ bits to supplement empty bits.
7. This instruction adopts pulse execution instructions (SFTLP).

Program Example:

1. When X0 = Off→On, M0 ~M15 will form 16 bits and shifts to the left (4 bits as a group).
2. The figure below illustrates the left shift of the bits in one scan.
   - ❶ M15 ~ M12 → carry
   - ❷ M11 ~ M8 → M15 ~ M12
   - ❸ M7 ~ M4 → M11 ~ M8
   - ❹ M3 ~ M0 → M7 ~ M4
   - ❺ X3 ~ X0 → M3 ~ M0 completed





API	Mnemonic	Operands	Function	Controllers
36	WSFR P	S D n1 n2	Word Shift Left	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			WSFR, WSFRP: 9 steps
D								*	*	*	*	*	*			
n_1					*	*										
n_2					*	*										

PULSE							16-bit							32-bit						
ES	EX	SS	SA	SX	SC	EH	ES	EX	SS	SA	SX	SC	EH	ES	EX	SS	SA	SX	SC	EH

Operands:

S: Start No. of the shifted device **D**: Start No. of the device to be shifted **n_1**: Length of data to be shifted

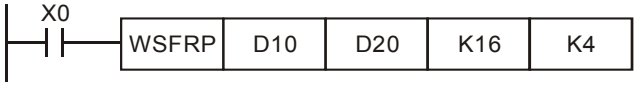
n_2: Number of words to be shifted in 1 shift

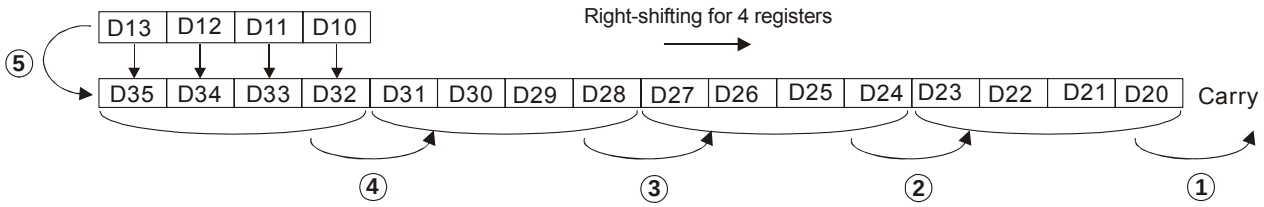
Explanations:

1. The type of devices designated by **S** and **D** has to be the same, e.g. K_nX, K_nY, K_nM, and K_nS as a category and T, C, and D as another category.
2. Provided the devices designated by **S** and **D** belong to K_n type, the number of digits of K_n has to be the same.
3. Range of **n_1**: 1~ 512
4. Range of **n_2**: 1 ~ **n_1**
5. See the specifications of each model for their range of use.
6. This instruction shifts the stack data of **n_1** words starting from **D** to the right for **n_2** words. **S** is shifted into **D** for **n_2** words to supplement empty words.
7. This instruction adopts pulse execution instructions (WSFRP)

Program Example 1:

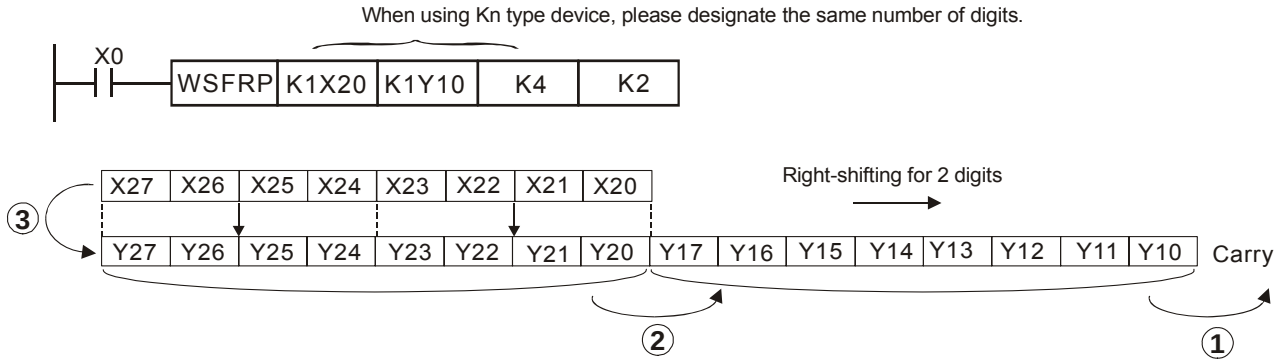
1. When X0 = Off→On, the 16 register stack data composed of D20 ~ D35 will shift to the right for 4 registers.
2. The figure below illustrates the right shift of the words in one scan.
   - ❶ D23 ~ D20 → carry
   - ❷ D27 ~ D24 → D23 ~ D20
   - ❸ D31 ~ D28 → D27 ~ D24
   - ❹ D35 ~ D32 → D31 ~ D28
   - ❺ D13 ~ D10 → D35 ~ D32 completed





Program Example 2:

1. When X0 = Off→On, the bit register stack data composed of Y10 ~ Y27 will shift to the right for 2 digits.
2. The figure below illustrates the right shift of the words in one scan.
   - ❶ Y17 ~ Y10 → carry
   - ❷ Y27 ~ Y20 → Y17 ~ Y10
   - ❸ X27 ~ X20 → Y27 ~ Y20 completed



API	Mnemonic	Operands	Function	Controllers
37	WSFL P	S D n1 n2	Word Shift Left	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			WSFL, WSFLP: 9 steps
D								*	*	*	*	*	*			
n_1					*	*										
n_2					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start No. of the shifted device **D**: Start No. of the device to be shifted **n_1**: Length of data to be shifted

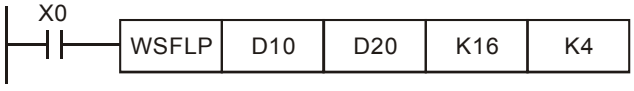
n_2: Number of words to be shifted in 1 shift

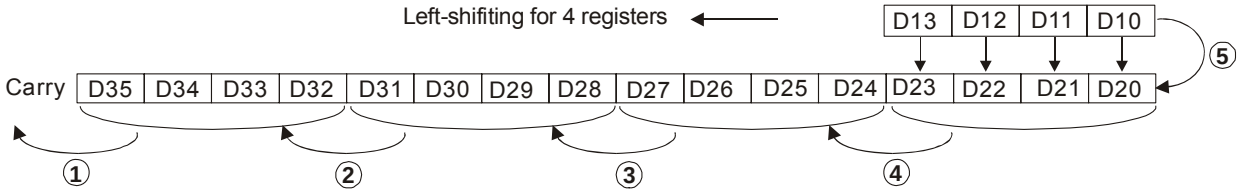
Explanations:

1. The type of devices designated by **S** and **D** has to be the same, e.g. K_nX, K_nY, K_nM, and K_nS as a category and T, C, and D as another category.
2. Provided the devices designated by **S** and **D** belong to K_n type, the number of digits of K_n has to be the same.
3. Range of **n_1**: 1~ 512
4. Range of **n_2**: 1 ~ **n_1**
5. See the specifications of each model for their range of use.
6. This instruction shifts the stack data of **n_1** words starting from **D** to the left for **n_2** words. **S** is shifted into **D** for **n_2** words to supplement empty words.
7. This instruction adopts pulse execution instructions (WSFLP)

Program Example:

1. When X0 = Off→On, the 16 register stack data composed of D20 ~ D35 will shift to the left for 4 registers.
2. The figure below illustrates the left shift of the words in one scan.
   - ❶ D35 ~ D32 → carry
   - ❷ D31 ~ D28 → D35 ~ D32
   - ❸ D27 ~ D24 → D31 ~ D28
   - ❹ D23 ~ D20 → D27 ~ D24
   - ❺ D13 ~ D10 → D23 ~ D20 completed





API	Mnemonic		Operands	Function	Controllers
38	SFWR	P	(S) (D) (n)	Shift Register Write	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	SFWR, SFWRP: 7 steps
D								*	*	*	*	*	*			
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

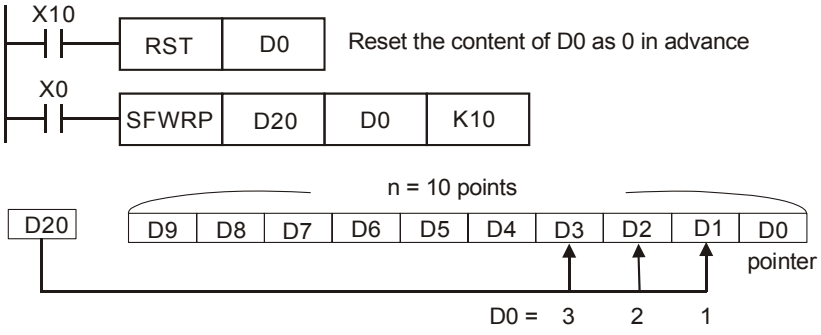
S: Device of stack data written in **D**: Start No. of stack data **n**: Length of stack data

Explanations:

1. Range of **n**: 2 ~ 512
2. See the specifications of each model for their range of use.
3. Flag: M1022 (carry flag)
4. The stack data of **n** words starting from **D** are defined as “first-in, first-out” stack data and designate the first device as the pointer. When the instruction is executed, the content in the pointer pluses 1, and the content in the device designated by **S** will be written into the designated location in the “first-in, first-out” stack data designated by the pointer. When the content in the pointer exceeds **n** - 1, this instruction will not process any new value written in and the carry flag M1022 = On.
5. This instruction adopts pulse execution instructions (SFWRP)

Program Example:

1. Pointer D0 is reset as 0. When X0 = Off→On, the content in D20 will be sent to D1 and the content in pointer D0 becomes 1. After the content in D20 is changed, make X0 = Off→On again, and the content in D2 will be sent to D2 and the content in D0 becomes 2.
2. The figure below illustrates the shift and writing in 1~2 execution of the instruction.
   - ❶ The content in D20 is sent to D1.
   - ❷ The content in pointer D0 becomes 1.



Remarks:

This instruction can be used together with API 39 SFRD for the reading/writing of “first-in, first-out” stack data.

API	Mnemonic		Operands	Function	Controllers
39	SFRD	P	(S) (D) (n)	Shift Register Read	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S								*	*	*	*	*	*			SFRD, SFRDP: 7 steps
D								*	*	*	*	*	*	*	*	
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

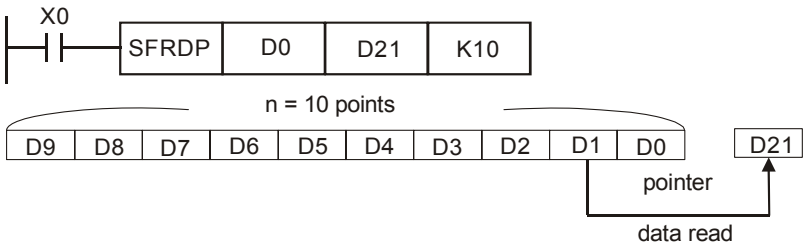
S: Start No. of stack data **D**: Device of stack data read out **n**: Length of stack data

Explanations:

1. Range of **n**: 2 ~ 512
2. See the specifications of each model for their range of use.
3. Flag: M1020 (zero flag)
4. The stack data of **n** words starting from **S** are defined as “first-in, first-out” stack data and designate the first device as the pointer. When the instruction is executed, the content in the pointer minuses 1, and the content in the device designated by **S** will be written into the designated location in the “first-in, first-out” stack data designated by the pointer. When the content in the pointer equals 0, this instruction will not process any new value written in and the zero flag M1020 = On.
5. This instruction adopts pulse execution instructions (SFRDP)

Program Example:

1. When X0 = Off→On, the content in D1 will be sent to D21 and D9~D2 will shift to the right for 1 register (content in D9 remains unchanged) and the content in D0 minus 1.
2. The figure below illustrates the shift and reading in 1~3 execution of the instruction.
   - ❶ The content in D1 is sent to D21.
   - ❷ D9 ~ D2 shift to the right for 1 register.
   - ❸ The content in D0 minuses 1.



Remarks:

This instruction can be used together with API 38 SFWR for the reading/writing of “first-in, first-out” stack data.

API	Mnemonic		Operands	Function	Controllers
40	ZRST	P	(D1) (D2)	Zero Reset	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D_1		*	*	*							*	*	*			ZRST, ZRSTP: 5 steps
D_2		*	*	*							*	*	*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

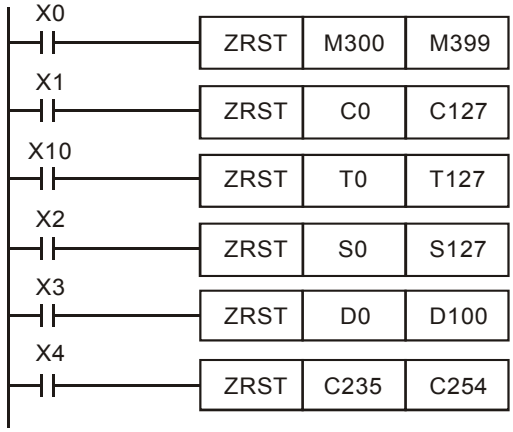
D_1: Start device of the range to be reset **D_2**: End device of the range to be reset

Explanations:

1. No. of operand D_1 ≤ No. of operand D_2.
2. D_1 and D_2 have to designate devices of the same type.
3. ES/EX/SS series MPU does not support E, F index register modification.
4. See the specifications of each model for their range of use.
5. When the instruction is executed, area from **D_1** to **D_2** will be cleared.
6. In ES/EX/SS, 16-bit counter and 32-bit counter cannot use ZRST instruction together.
7. In SA/EH, 16-bit counter and 32-bit counter can use ZRST instruction together.
8. When **D_1** > **D_2**, only operands designated by **D_2** will be reset.

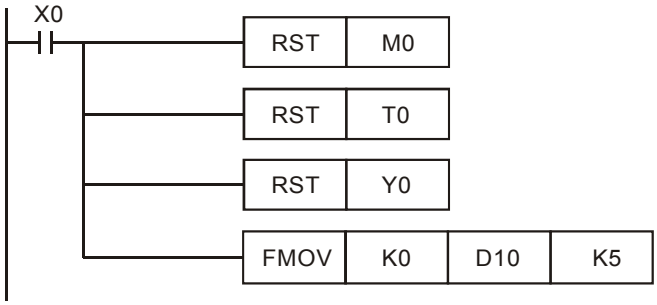
Program Example:

1. When X0 = On, auxiliary relays M300 ~ M399 will be reset to Off.
2. When X1 = On, 16 counters C0 ~ C127 will all be reset (writing in 0; contact and coil being reset to Off).
3. When X10 = On, timers T0 ~ T127 will all be reset (writing in 0; contact and coil being reset to Off).
4. When X2 = On, steps S0 ~ S127 will be reset to Off.
5. When X3 = On, data registers D0 ~ D100 will be reset to 0.
6. When X4 = On, 32-bit counters C235 ~ C254 will all be reset. (writing in 0; contact and coil being reset to Off)



Remarks:

1. Devices, e.g. bit devices Y, M, S and word devices T, C, D, can use RST instruction.
2. API 16 FMOV instruction is also to send K0 to word devices T, C, D or bit registers KnY, KnM, KnS for reset.



API	Mnemonic		Operands	Function	Controllers
41	DECO	P	(S) (D) (n)	Decode	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*	*	*	*					*	*	*	*	*	DECO, DECOP: 7 steps
D		*	*	*							*	*	*	*	*	
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

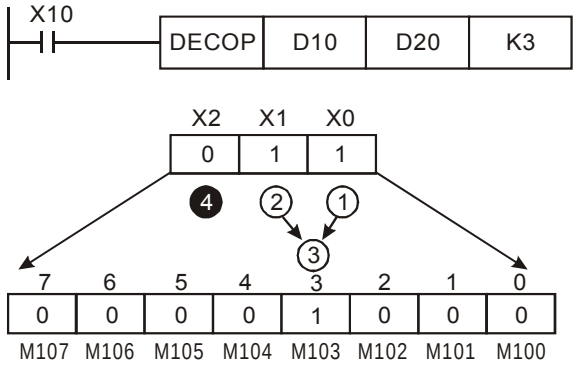
S: Source device to be decoded **D**: Device for storing the decoded result **n**: Length of decoded bits

Explanations:

1. Range of **n** when **D** is a bit device: 1 ~ 8
2. Range of **n** when **D** is a word device: 1 ~ 4
3. ES/EX/SS series MPU does not support E, F index register modification.
4. See the specifications of each model for their range of use.
5. The lower "**n**" bits of **S** are decoded and the results of "2^n" bits are stored in **D**.
6. This instruction adopts pulse execution instructions (DECOP)

Program Example 1:

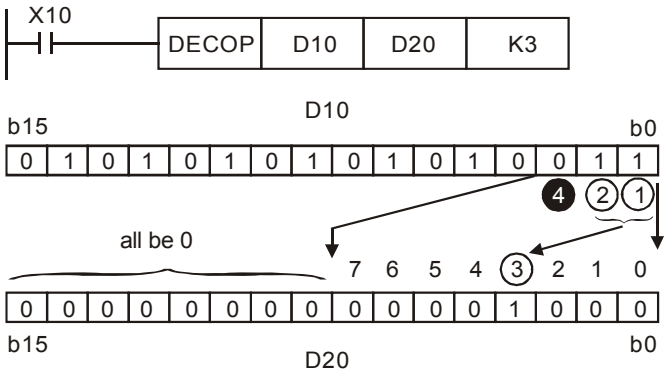
1. When **D** is used as a bit device, **n** = 1 ~ 8. Errors will occur if **n** = 0 or **n** > 8.
2. When **n** = 8, the maximum points to decode is 2^8 = 256 points. (Please be aware of the storage range of the devices after the decoding and do not use the devices repeatedly.)
3. When X10 = Off→On, this instruction will decode the content in X0 ~ X2 to M100 ~ M107.
4. When the source of data is 1 + 2 = 3, set M103, the 3rd bit starting from M100, as 1.
5. After the execution of this instruction is completed and X10 turns to Off, the content that has been decoded and output keeps acting.



Program Example 2:

1. When **D** is used as a word device, n = 1 ~ 4. Errors will occur if n = 0 or n > 4.
2. When **n** = 4, the maximum points to decode is 2^4 = 16 points.
3. When X10 = Off→On, this instruction will decode b2 ~ b0 in D10 to b7 ~ b0 in D20. b15 ~ b8 that have not been used in D20 will all become 0.

4. The lower 3 bits of D10 are decoded and stored in the lower 8 bits of D20. The higher 8 bits of D20 are all 0.
5. After the execution of this instruction is completed and X10 turns to Off, the content that has been decoded and output keeps acting.



API	Mnemonic		Operands	Function	Controllers
42	ENCO	P	(S) (D) (n)	Encode	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*	*							*	*	*	*	*	ENCO, ENCOP: 7 steps
D											*	*	*	*	*	
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

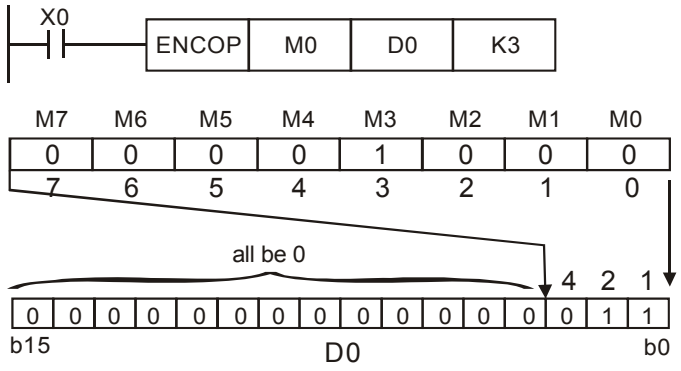
S: Source device to be encoded **D**: Device for storing the encoded result **n**: Length of encoded bits

Explanations:

1. Range of **n** when **S** is a bit device: 1 ~ 8
2. Range of **n** when **S** is a word device: 1 ~ 4
3. ES/EX/SS series MPU does not support E, F index register modification.
4. See the specifications of each model for their range of use.
5. The lower "2^n" bits of **S** are encoded and the result is stored in **D**.
6. If several bits of **S** are 1, the first bit that is 1 will be processed orderly from high bit to low bit.
7. If no bits of S is 1, M1067, M1068 = On and D1067 records the error code 0E1A (hex).
8. This instruction adopts pulse execution instructions (ENCOP)

Program Example 1:

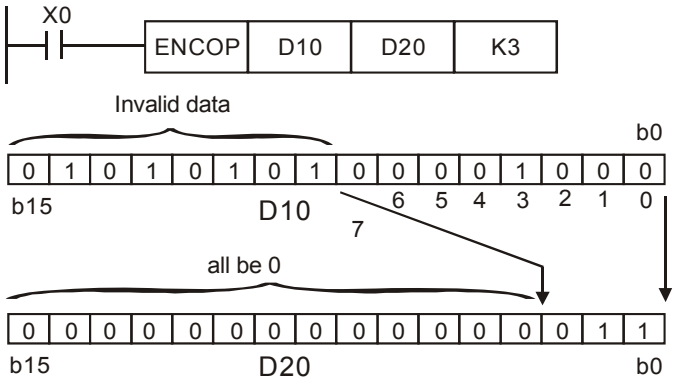
1. When **S** is used as a bit device, **n** = 1 ~ 8. Errors will occur if **n** = 0 or **n** > 8.
2. When **n** = 8, the maximum points to encode is 2^8 = 256 points.
3. When X10 = Off→On, this instruction will encode the 2^3 bits data (M0 ~ M7) and store the result in the lower 3 bits (b2 ~ b0) of D0. b15 ~ b3 that have not been used in D0 will all become 0.
4. After the execution of this instruction is completed and X10 turns to Off, the content in D remains unchanged.



Program Example 2:

1. When **S** is used as a word device, n = 1 ~ 4. Errors will occur if n = 0 or n > 4.
2. When **n** = 4, the maximum points to decode is 2^4 = 16 points.
3. When X10 = Off→On, this instruction will encode 2^3 bits (b0 ~ b7) in D10 and stores the result in the lower 3 bits (b2 ~ b0) of D20. b15 ~ b3 that have not been used in D20 will all become 0. b8 ~ b15 of D10 are invalid data.

4. After the execution of this instruction is completed and X10 turns to Off, the content in D remains unchanged.



API	Mnemonic			Operands	Function	Controllers		
43	D	SUM	P	(S) (D)	Sum of Active Bits	ES/EX/SS	SA/SX/SC	EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	SUM, SUMP: 5 steps DSUM, DSUMP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

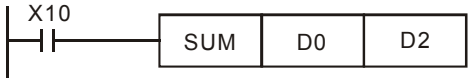
S: Source device **D**: Destination device for storing counted value

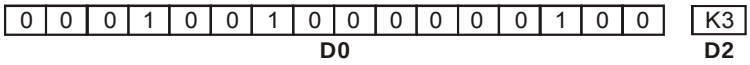
Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. Flag: M1020 (zero flag)
4. Among the bits of **S**, the total of bits whose content is “1” will be stored in **D.**
5. When all the 16 bits of **S** are “0”, zero flag M1020 = On.
6. When 32- instruction is in use, **D** will occupy 2 registers.

Program Example:

When X10 = On, among the 16 bits of D0, the total of bits whose content is “1” will be stored in D2.





API	Mnemonic			Operands	Function	Controllers
44	D	BON	P	(S) (D) (n)	Check Specified Bit Status	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	BON, BONP: 7 steps DBON, DBONP: 13 steps
D		*	*	*												
n					*	*					*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

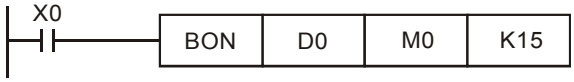
S: Source device **D**: Device for storing check result **n**: Bits specified for check

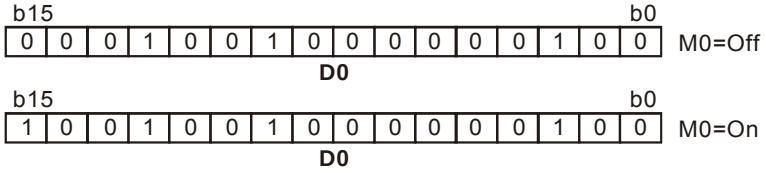
Explanations:

1. If **S** is used in device F, only 16-bit instruction is applicable.
2. Range of **n**: 0 ~ 15 (16-bit instruction); 0 ~ 31 (32-bit instruction)
3. See the specifications of each model for their range of use.
4. When the **n**th bit of **S** is “1”, D = On; when the **n**th bit of **S** is “0”, D = Off.

Program Example:

1. When X0 = On, assume the 15th bit of D0 is “1”, and M0 = On. Assume the 15th bit of D0 is “0”, and M0 = Off.
2. When X0 goes Off, M0 will remains in its previous status.





API	Mnemonic			Operands	Function	Controllers
45	D	MEAN	P	(S) (D) (n)	Mean	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			MEAN, MEANP: 7 steps DMEAN, DMEANP: 13 steps
D								*	*	*	*	*	*	*	*	
n					*	*	*	*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

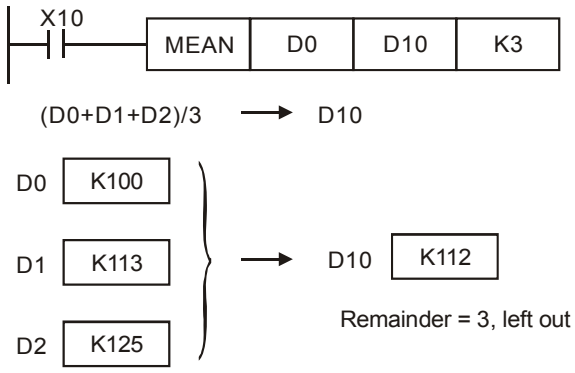
S: Start device to obtain mean value **D**: Destination device for storing mean value **n**: The number of consecutive source devices used

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. Range of **n**: 1 ~ 64
3. In ES/EX/SS series models: Operand S cannot designate KnX, KnY, KnM, KnS.
4. ES/EX/SS series MPU does not support E, F index register modification.
5. See the specifications of each model for their range of use.
6. After the content of **n** devices starting from **S** are added up, the mean value of the result will be stored in **D**.
7. Remainders in the operation will be left out.
8. Provided the No. of designated device exceeds its normal range, only the No. within the normal range can be processed.
9. If **n** falls without the range of 1 ~ 64, PLC will determine it as an “instruction operation error”.

Program Example:

When X10 = On, the contents in 3 (n = 3) registers starting from D0 will be summed and then divided by 3. The obtained mean value will be stored in D10 and the remainder will be left out.



API	Mnemonic	Operands	Function	Controllers
46	ANS	(S) (m) (D)	Timed Annunciator Set	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S											*					ANS: 7 steps
m					*											
D				*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

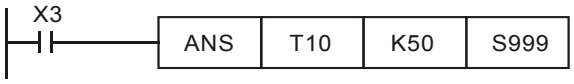
S: Timer for monitoring annunciator **m**: Time setting **D**: Annunciator device

Explanations:

1. Range of **S**: for SA/SX/SC T0 ~ T191; for EH/EH2/SV T0 ~ T199.
2. **m** can designate K1 ~ K32,767 (unit: 100ms)
3. Range of **D**: for SA/SX/SC S896 ~ S1023; for EH/EH2/SV S900 ~ S1023.
4. See the specifications of each model for their range of use.
5. Flags: M1048 (annunciator in action); M1049 (valid monitoring)
6. This instruction is used for enabling the annunciator.

Program Example:

If X3 = On for more than 5 seconds, annuniciator point S999 = On. Even X3 goes Off afterwards, S999 will still keep On. However, T10 will be reset to Off and the present value = 0.



API	Mnemonic		Operands	Function	Controllers
47	ANR	P		Annunciator Reset	ES/EX/SS | SA/SX/SC | EH/SV

OP	Descriptions	Program Steps
N/A		ANR, ANRP: 1 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

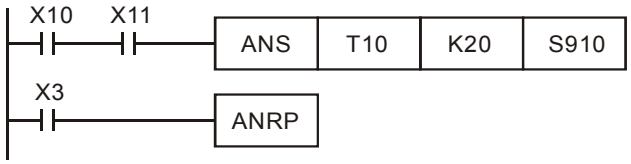
Explanations:

1. No operand.
2. This instruction is used for resetting the annunciator.
3. When more than one annuciators are On, the annunciator of smaller number will be reset.
4. This instruction adopts pulse execution instructions (ANRP)

Program Example:

1. If X10 and X11 = On at the same time for more than 2 seconds, annuniciator point S910 = On. Even X10 and X11 go Off afterwards, S910 will still keep On. However, T10 will be reset to Off and the present value = 0.

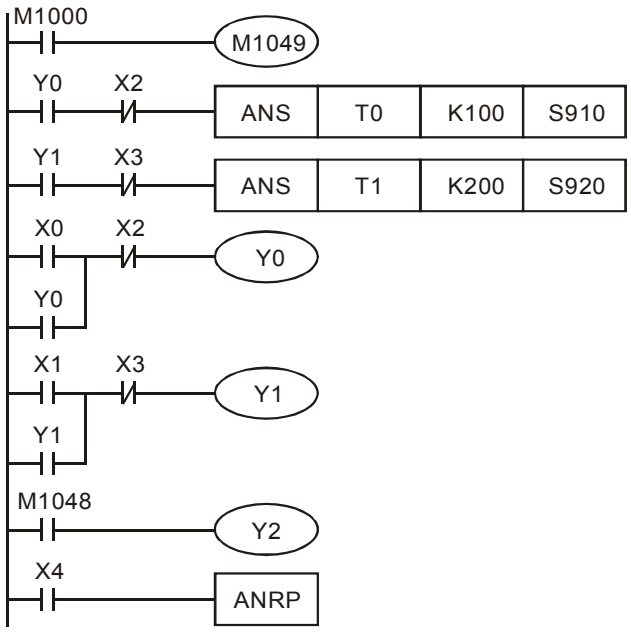
2. When X10 and X11 are On at the same time for less than 2 seconds, the present value of T10 will be reset to 0.
3. When X3 goes from Off to On,
   S896 ~ S1023 in SA/SX/SX are able to reset the annunciators in action.
   S900 ~ S1023 in EH/EH2/SV are able to reset the annunciators in action.
4. When X3 goes from Off to On again, the annuniator with secondary smaller No. will be reset.



Remarks:

1. Flag:
   a) M1048 (annunciator in action): When M1049 = On, any of the annunciators among S896 ~ S1023 in SA/SX/SC or S900 ~ S1023 in EH/EH2/SV starts output, M1048 will be On.
   b) M1049 (valid monitoring): When M1049 = On, D1049 will automatically display the annuciator of the smallest number in action.
2. Application of annunciators:
   I/O point configuration:

X0: Forward switch	Y0: Forward	S910: Forward annunciator
X1: Backward switch	Y1: Backward	S920: Backward annunciator
X2: Front position switch	Y2: Annunciator indicator	
X3: Back position switch		
X4: Annunciatro reset button		



1. M1048 and D1049 are valid only when M1049 = On.
2. When Y0 = On for more than 10 seconds and the device fails to reach the frong position X2, S910= On.
3. When Y1 = On for more than 10 seconds and the device fails to reach the back position X3, S920= On.

4. When backward switch X1 = On and backward device Y1 = On, Y1 will go Off only when the device reaches the back position switch X3.
5. Y2 will be On when any annunciator is enabled. Whenever X4 is on, 1 annunciator in action will be reset. The reset starts from the annunciator with the smallest No.

API	Mnemonic			Operands	Function	Controllers		
48	D	SQR	P	(S) (D)	Square Root	ES/EX/SS	SA/SX/SC	EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			SQR, SQRP: 5 steps DSQR, DSQRP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

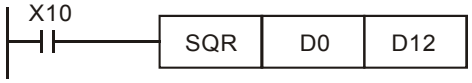
S: Source device **D**: Device for storing the result

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1067 (instruction operation error)
3. This instruction performs a square root operation on **S** and stores the result in **D**.
4. **S** can only be a positive value. If **S** is negative, PLC will regard it as an “instruction operation error” and will not execute this instruction. M1067 and M1068 = On and D1067 records the error code 0E1B (hex).
5. The operation result **D** should be integer only, and the decimal will be left out. Borrow flag M1021 = On.
6. When the operation result **D** = 0, zero flag M1020 = On.

Program Example:

When X10 = On, the instruction performs a square root on D0 and stores the result in D12.



$\sqrt{D0} \rightarrow D12$

API	Mnemonic			Operands	Function	Controllers
49	D	FLT	P	(S) (D)	Convert BIN integer to binary floating point	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			FLT, FLTP: 5 steps DFLT, DFLTP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

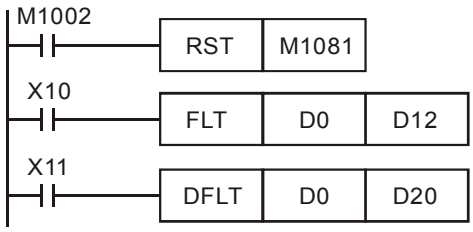
S: Source device for conversion **D**: Device for storing the conversion result

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1081 (FLT instruction function switch); M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. **D** will occupy 2 consecutive devices
4. When M1081 is Off, BIN integer is converted into binary floating point value. At this time, **S** of the 16-bit instruction, FLT, occupies 1 register and **D** occupies 2 registers.
   a) If the absolute value of the conversion result $>$ max. floating value, carry flag M1022 = On.
   b) If the absolute value of the conversion result $<$ min. floating value, carry flag M1021 = On.
   c) If the conversion result is 0, zero flag M1020 = On.
5. When M1081 is On, binary floating point value is converted into BIN integer (digits after decimal point are left out). At this time, **S** of the 16-bit instruction, FLT, occupies 2 registers and **D** occupies 1 register (action same as that of INT instruction).
   a) If the conversion result exceeds the range of BIN integer available in **D** (for 16-bit: -32,768 ~ 32,767; for 32-bit: -2,147,483,648 ~ 2,147,483,647), D will obtain the maximum or minimum value and carry flag M1022 = On.
   b) If any digits is left out during the conversion, borrow flag M1021 = On.
   c) If **S** = 0, zero flag M1020 = On.
   d) After the conversion, **D** stores the result in 16 bits.

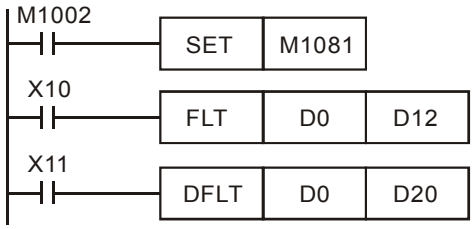
Program Example 1:

1. When M1081 = Off, the BIN integer is converted into binary floating point value.
2. When X10 = On, D0 (BIN integer) is converted into D13 and D12 (binary floating point value).
3. When X11 = On, D1 and D0 (BIN integer) are converted into D21 and D20 (binary floating point value).
4. If D0 = K10, X10 will be On. The 32-bit value of the converted floating point will be H41200000 and stored in 32-bit register D12 (D13).
5. If 32-bit register D0 (D1) = K100,000, X11 will be On. The 32-bit value of the converted floating point will be H47C35000 and stored in 32-bit register D20 (D21).



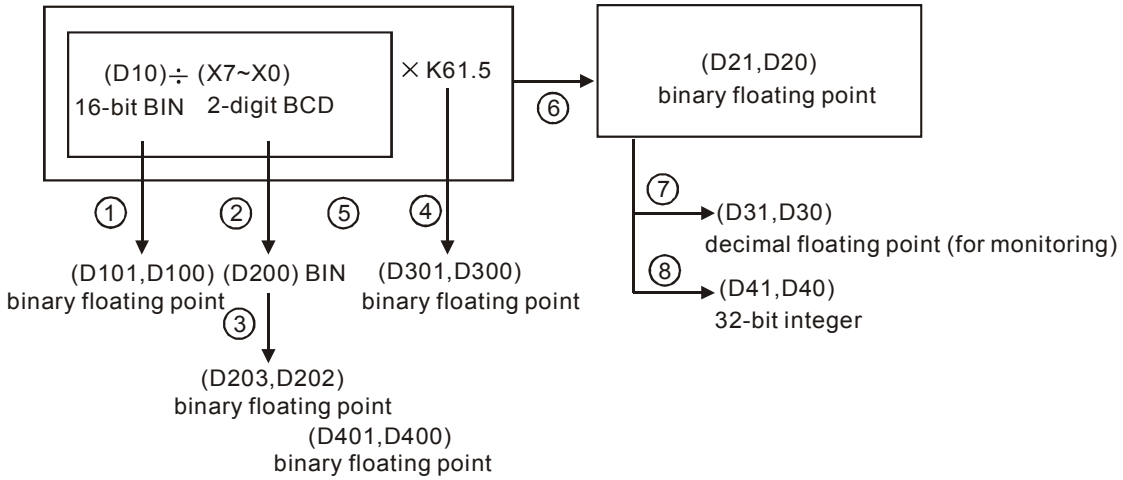
Program Example 2:

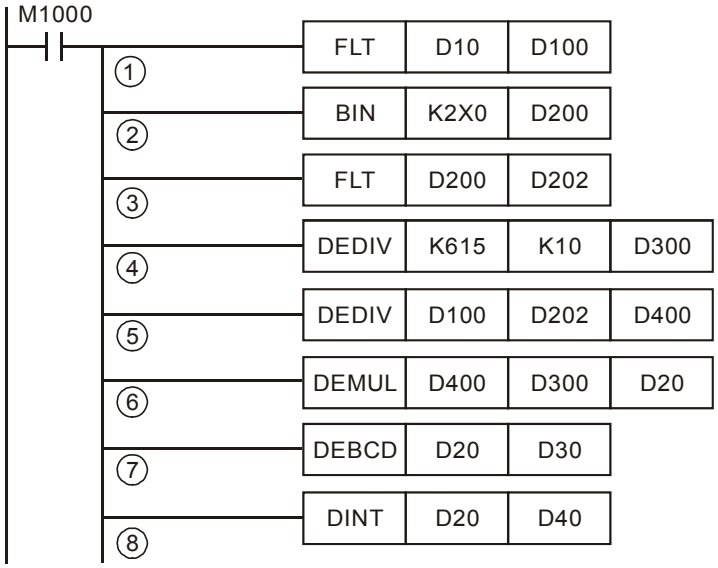
1. When M1081 = On, the binary floating point value is converted into BIN integer (the decimal is left out).
2. When X10 = On, D0 and D1 (binary floating point value) are converted into D12 (BIN integer). If D0 (D1) = H47C35000, the floating point value will be presented as 100,000. Due to that the value is larger than the value presentable by the 16-bit register D12, the result will be D12 = K32, 767 and M1022 = On.
3. When X11 = On, D1 and D0 (binary floating point value) are converted into D21 and D20 (BIN integer). If D0 (D1) = H47C35000, the floating point value will be presented as 100,000. The result will be stored in the 32-bit register D20 (D21).



Program Example 3:

Please use this instruction to complete the following operation.





① D10 (BIN integer) is converted to D101 and D102 (binary floating point value).

② X7 ~ X0 (BCD value) are converted to D200 (BIN value).

③ D200 (BIN integer) is converted to D203 and D202 (binary floating point value).

4The result of K615 ÷ K10 is stored in D301 and D300 (binary floating point value).

⑤ The result of binary decimal division (D101, D100) ÷ (D203, D202) is stored in D401 and D400 (binary floating point value).

⑥ The result of binary decimal multiplication (D401, D400) × (D301, D300) is stored in D21 and D20 (binary floating point value).

⑦ D21 and D20 (binary floating point value) are converted to D31 and D30 (decimal floating point value).

⑧ D21 and D20 (binary floating point value) are converted to D41 and D40 (BIN integer).

API	Mnemonic		Operands	Function	Controllers
50	REF	P	D n	Refresh	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D	*	*														REF, REFP: 5 steps
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Start device to be I/O refreshed **n**: Number of items to be I/O refreshed

Explanations:

1. **D** must designate X0, X10, Y0, Y10…the points whose 1s digit is "0". See remarks for more details.
2. Range of **n**: 8 ~ 256 (has to be the multiple of 8).
3. See the specifications of each model for their range of use.
4. The status of all PLC input/output terminals will be updated after the program scans to END. When the program starts to scan, the status of the external input terminal is read and stored into the memory of the input point. The output terminal will send the content in the output memory to the output device after END instruction is executed. Therefore, this instruction is applicable when the latest input/output data are needed for the operation.

Program Example 1:

When X0 = On, PLC will read the status of input points X0 ~ X17 immediately and refresh the input signals without any input delay.

Program Example 2:

When X0 = On, the 8 output signal from Y0 ~ Y7 will be sent to output terminals and refreshed without having to wait for the END instruction for output.

Remarks:

The instruction only process the I/O points X0 ~ X17 and Y0 ~ Y17 of ES/EX/SS/SA/SX/SC series MPU, namely **n** = K8 or **n** = K16.

API	Mnemonic		Operands	Function	Controllers
51	REFF	P	n	Refresh and Filter Adjust	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
n					*	*										REFF, REFFP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

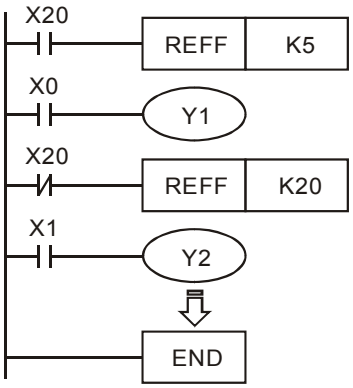
n: Response time (unit: ms)

Explanation:

1. Range of **n**: for SA/SX/SC, **n** = K0 ~ K20; for EH/EH2/SV, **n** = K0 ~ K60.
2. To avoid interferences, X0 ~ X17 of EH/EH2/SV series MPU and X0 ~ X7 of SA/SX/SC series MPU are equipped with digital filters on output terminals. Digital filters adjust the response time by REFF instruction. This instruction sets up **n** directly in D1020 (adjusting the response time of X0 ~ X7) and D1021 (adjusting the response time of X10 ~ X17).
3. Rules for adjusting the reponse time of the filter at X0 ~ X17:
   a) When the power of PLC turns from Off to On or the END instruction is being executed, the response time will be determined upon the contents in D1020 and D1021.
   b) You can use MOV instruction in the program to move the time values to D1020 and D1021 and make adjustments in the next scan.
   c) You can use REFF instruction to change the response time during the execution of the program. The changed response time will be move to D1020 and D1021 and you can make adjustments in the next scan.

Program Example:

1. When the power of PLC turns from Off to On, the response time of X0 ~ X17 will be determined by the contents in D1020 and D1021.
2. When X20 = On, REFF K5 will be executed and the response time will be changed to 5ms for the adjustment in the next scan.
3. When X20 = Off, the REFF K20 will be executed and the response time will be changed to 20ms for the adjustment in the next scan.



Remarks:

When inserting an interruption subroutine in the program or using the high speed counter or API 56 SPD instruction, the corresponding signals at the input terminals will not delay and has nothing to do with this instruction.

API	Mnemonic	Operands	Function	Controllers
52	MTR	S D1 D2 n	Input Matrix	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*															MTR: 9 steps
D_1		*														
D_2		*	*	*												
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start device of matrix input **D_1**: Start device of matrix output **D_2**: Corresponding start device for matrix scan

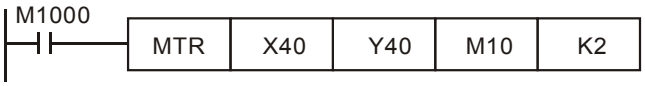
n: Number of arrays in matrix scan

Explanations:

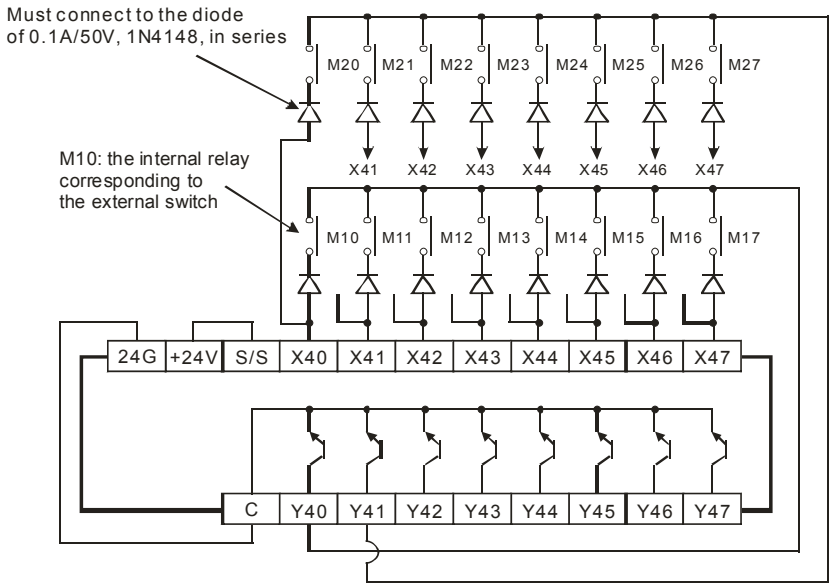
1. **S** must designate X0, X10…the X points whose 1st digit is "0" and occupies 8 consecutive points.
2. **D_1** must designate Y0, Y10…the Y points whose 1st digit is "0" and occupies n consecutive points.
3. **D_2** must designate Y0, M0. S0…the Y, M, S points whose 1st digit is "0".
4. Range of **n**: 2 ~ 8.
5. See the specifications of each model for their range of use.
6. Flag: M1029 (execution of the instruction is completed).
7. **S** is the start device No. of all input terminals connected to the matrix. Once **S** is designated, the 8 points following the No. will be the input terminals in the matrix.
8. **D_1** designate the start device No. of transistor output Y in the matrix scan.
9. This instruction occupies continuous 8 input devices starting from **S**. **n** external output terminals starting from **D_1** read the 8 switches of **n** arrays by matrix scan, obtaining 8 × **n** multiple-matrix input points. The status of scanned switches will be stored in the devices starting from **D_2**.
10. Maximum 8 input switches can be parallelly connected in 8 arrays and obtaining 64 input points (8 × 8 = 64).
11. When the 8-point 8-array matrix inputs are in use, the reading time of each array is approximately 25ms, totaling the reading of 8 arrays 200ms, i.e. the input signals with On/Off speed of over 200ms are not applicable in a matrix input.
12. The drive contact of this instruction uses normally On contact M1000.
13. Whenever this instruction finishes a matrix scan, M1029 will be On for one scan period.
14. There is no limitation on the number of times using the instruction, but only one instruction can be executed in a period of time.

Program Example:

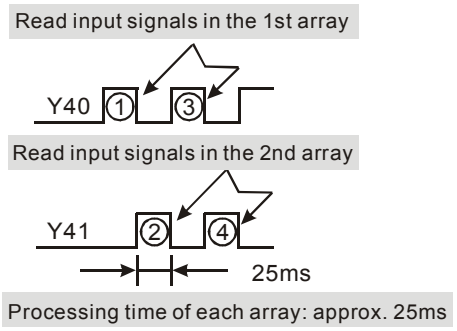
1. When PLC RUN, MRT instruction will start to be executed. The statuses of the external 2 arrays of 16 switches will be read in order and stored in the internal relays M10 ~ M17, M20 ~ M27.



2. The figure below illustrates the external wiring of the 2-array matrix input loop constructed by X40 ~ X47 and Y40 ~ Y41. The 16 switches correponds to the internal relays M10 ~ M17, M20 ~ M27. Should be used with MTR instruction.



3. See the figure above. The 8 points starting from X40 start to perform a matrix scan from Y40 ~ Y41 (n = 2). **D_2** designates that the start device No. of the read results is M10, indicating that the first array is read to M10 ~ M17 and the second array is read to M20 ~ M27.



API	Mnemonic	Operands	Function	Controllers
53	D HSCS	(S1) (S2) (D)	High Speed Counter Set	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DHSCS: 13 steps
S_1					*	*	*	*	*	*	*	*	*	*		
S_2												*				
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

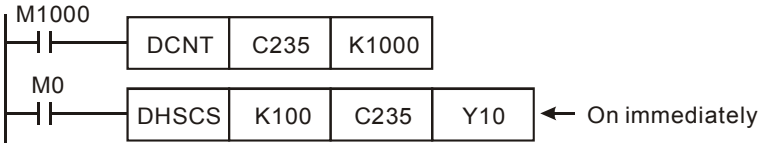
S_1: Comparative value **S_2**: No. of high speed counter **D**: Comparison result

Explanations:

1. **S_2** has to designate the No. of high speed counters C235 ~ C255. See remarks for more details.
2. **D** can designate I0□0; □ = 1 ~ 6. ES series MPU does not support this.
3. **D** of ES and SA series MPU does not support E, F index register modification.
4. See the specifications of each model for their range of use.
5. Flags: M1289 ~ M1294 are interruption disability of the high speed counters in EH/EH2/SV series MPU. See Program Example 3 for more details.
6. The high speed counter inputs counting pulses from the corresponding external input terminals X0 ~ X17 by inserting an interruption. When the high speed counter designated in **S_2** pluses 1 or minuses 1, DHSCS instruction will perform a comparison immediately. When the present value in the high speed counter equals the comparative value designated in **S_1**, device designated in **D** will turn On. Even the afterward comparison results are unequal, the device will still be On.
7. If the devices specified as the device **D** are Y0 ~ Y17, when the compare value and the present value of the high-speed counter are equal, the comparison result will immediately output to the external inputs Y0 ~ Y17, and other Y devices will be affected by the scan cycle. However, M, S devices are immediate output and will not be affected by the scan cycle.

Program Example 1:

After PLC RUN and M0 = On, DHSCS instruction will be executed. When the present value in C235 changes from 99 to 100 or 101 to 100, Y10 will be On constantly.

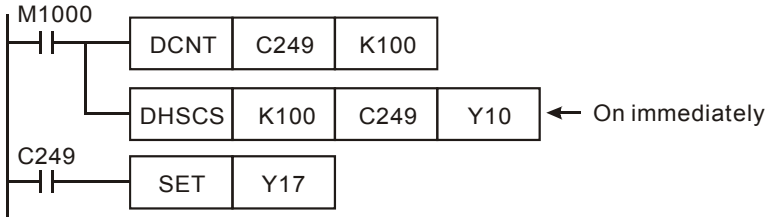


Program Example 2:

Differences between Y output of DHSCS instruction and general Y output:

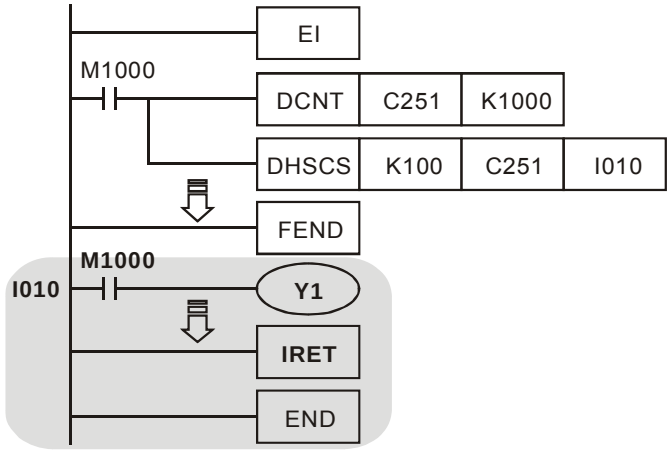
a) When the present value in C249 changes from 99 to 100 or 101 to 100, Y10 outputs immediately to the external output point by interruption and has nothing to do with the PLC scan time. However, the time will still be delayed by the relay (10ms) or transistor (10us) of the output module.

b) When the present value in C249 changes from 99 to 100, the drive contact of C249 will be On immediately. When the execution arrives at SET Y17, Y17 will still be affected by the scan time and will output after END instruction.



Program Example 3:

1. High speed counter interruption:
   a) Operand **D** of DHSCS instruction can designate I0□0, □ = 1 ~ 6, as the timing of interruption when the counting reaches its target.
   b) ES/EX/SS series MPU does not support high speed counter interruption.
   c) SA/SX/SC series MPU supports high speed counter interruption. However, when DHSCS instruction designates an I interruption, the designated high speed counter cannot be used in DHSCS, DHSCR, DHSZ instructions. Misuse of high speed counter will result in error.
   d) For SA/SX/SC series MPU, when the counting reaches the target, the interruption will occur. X0 is the counter for counting input and the interruption No. is I010 (1 phase 2 inputs and A-B phase counter No. C246 ~ C254 can only designate I010). X1 designates I020; X2 designates I030; X3 designates I040; X4 designates I050; X5 designates I060, totaling 6 points.
   e) When the present value in C251 changes from 99 to 100 or 101 to 100, the program will jump to I010 and execute the interruption service subroutine.



2. In SA/SX/SC series MPU, M1059 is “I010 ~ I060 high speed counter interruption forbidden” flag.
3. In EH/EH2/SV series MPU, M1289 ~ M1294 are the respectively for I010 ~ I060 “high speed counter interruption forbidden flags”, i.e. when M1294 = On, I060 interruption will be forbidden.

Interruption pointer I No.	Interruption forbidden flag	Interruption pointer I No.	Interruption forbidden flag
I010	M1289	I040	M1292
I020	M1290	I050	M1293
I030	M1291	I060	M1294

Remarks:

1. The output contact of the high speed counter and the comparative outputs of API 53 DHSCS, API 34 DHSCR and API 55 DHSZ instructions only perform comparison and contact outputs when there is a counting input. When using data operation instructions, e.g. DADD, DMOV, for changing the present value in the high speed counter or making the present value equals the set value, there will not be comparisons or comparative outputs because there is no counting inputs.
2. High speed counters supported by ES/EX/SS series MPU (total bandwidth: 20KHz):

Type / Input	1-phase 1 input							1-phase 2 inputs			2-phase 2 inputs		
	C235	C236	C237	C238	C241	C242	C244	C246	C247	C249	C251	C252	C254
X0	U/D				U/D		U/D	U	U	U	A	A	A
X1		U/D			R		R	D	D	D	B	B	B
X2			U/D			U/D			R	R		R	R
X3				U/D		R	S			S			S

U: Progressively increasing input A: A phase input S: Input started

D: Progressively decreasing input B: B phase input R: Input cleared

a) Input points X0 and X1 can be planned as counters of higher speed (1 phase input can reach 20KHz). However, the total counting frequency of the two input points has to be smaller or equal 20KHz. Provided the input is a 2-phas input signal, the counting frequency will be approximately 4KHz. The frequency of the input points X2 and X3 (1-phase) can reach 10KHz.

b) For ES/EX/SS series MPU, the uses of DHSCS instructio with DHSCR instruction cannot be more than 4 times.

3. High speed counters supported by SA/SX series MPU (total bandwidth: 40KHz):

Type / Input	1-phase 1 input									1-phase 2 inputs			2-phase 2 inputs			
	C235	C236	C237	C238	C239	C240	C241	C242	C244	C246	C247	C249	C251	C252	C253	C254
X0	U/D						U/D		U/D	U	U	U	A	A	B	A
X1		U/D					R		R	D	D	D	B	B	A	B
X2			U/D					U/D			R	R		R		R
X3				U/D				R	S			S				S
X4					U/D											
X5						U/D										

U: Progressively increasing input A: A phase input S: Input started

D: Progressively decreasing input B: B phase input R: Input cleared

a) Input points X0 and X1 for 1-phase input can reach a frequency of 20KHz and X2 ~ X5 can reach 10KHz. 2-phase input (X0, X1) C251, C252 and C254 can reach a frequency of 4KHz and C253 reach 25KHz (only supports 4 times frequency counting).

b) Functions of the input point X5:

   i) When M1260 = Off, C240 is the general U/D high speed counuter.

   ii) When M1260 = On and C240 is enabled by DCNT instruction, X5 will be the shared reset signal for C235 ~ C239. The counter C240 will still receive the counting input signals from X5.

4. High speed counters supported by SC series MPU (total bandwidth: 130KHz):

Type / Input	1-phase 1 input											1-phase 2 inputs				2-phase inputs			
	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245	C246	C247	C249	C250	C251	C252	C254	C255
X0	U/D						U/D			U/D		U	U	U		A	A	A	
X1		U/D					R			R		D	D	D		B	B	B	
X2			U/D					U/D					R	R			R	R	
X3				U/D				R		S				S				S	
X4					U/D														
X5						U/D													
X10									U/D						U				A
X11											U/D				D				B

U: Progressively increasing input A: A phase input S: Input started

D: Progressively decreasing input B: B phase input R: Input cleared

a) The functions of the high speed counters of input points X0 ~ X5 are the same of those in SA/SX series MPU.

b) The input points of 1-phase input X10 (C243), X11 (C245) and (X10, X11) C250 can reach a frequency of 100KHz. The total bandwidth of X10 ~ X11 is 130KHz. C255 of the 2-phase input (X10, X11) can reach a frequency of 50KHz.

c) For SA/SX/SC series MPU, the uses of DHSCS instruction with DHSCR instruction cannot be more than 6 times and the uses of DHSZ instruction cannot be more than 6 times as well. When DHSCR instruction designates I interruption, the designated high speed counter cannot be used in other DHSCS, DHSCR and DHSZ instructions.

d) The functions of X10 ~ X11 high speed counters in SC series MPU:

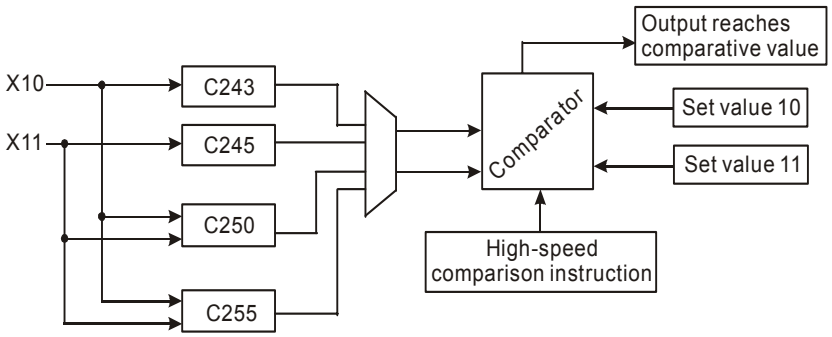
   i) When X10 and X11 are set as 1-phase 1 input or 1-phase 2 inputs. The maximum frequency of a single phase can reach 100KHz. When they are set as 2-phase 2 inputs, the maximum frequency can reach 50KHz.

   ii) X10 and X11 can select rising-edge counting mode or falling-edge counting mode. X10 is set by D1166 and X11 is set by D1167. K0: rising-edge counting. K1: falling-edge counting. K2: rising/falling edge counting (only supports X10).

   iii) The counting up and down of C243 are determined by the On/Off of M1243. The counting up and down of C245 are determined by the On/Off of M1245. Rising-edge and falling-edge counting are not able to take place at the same time. The rising-edge trigger and falling-edge trigger of C250 are determined by the content (K0 or K1) of D1166. C255 can only be used in a 4 times frequency counting and you can only select rising-edge trigger.

   iv) When C243 or C245 is in use, you will not be able to use C250 or C255, and vice versa.

   v) High speed counter and high speed comparator:



vi) Explanations on high speed counter and high speed comparator:

(1) When DHSCS and DHSCR instructions use the high speed counter (C243/C245/C250/C255), they can only use the set values of 2 groups of high speed comparative instructions. Assume that a group of comparative instruction DHSCS D0 C243 Y10 is already in use, you can only set another group DHSCR D2 C243 Y10 or DHSCS D4 C245 Y10.

(2) When DHSZ instruction use the high speed counter (C243/C245/C250/C255), it can only use the set value of a group of comparators.

(3) The number of set values in a high-speed comparative instruction offered in SA/SX series MPU will not decrease becasus of the addition of the new high speed counters.

(4) If the high-speed comparative instruction DHSCS requires a high-speed reponse output, it is suggested that you use Y10 or Y11 for the output. If you use other general devices for the output, there will be delay of 1 scan period. For example, when in I0x0 interruption, C234 will correspond to I020, C245 to I040 and C250/C255 to I060.

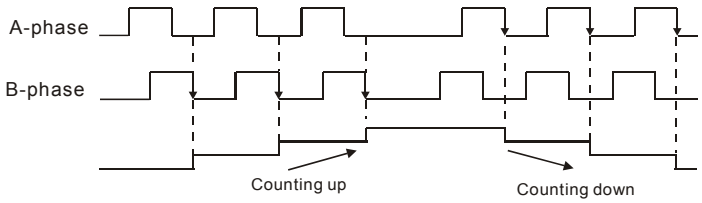
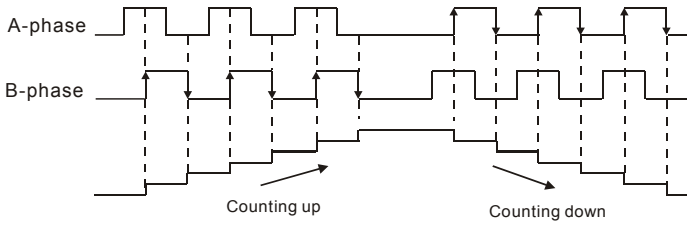
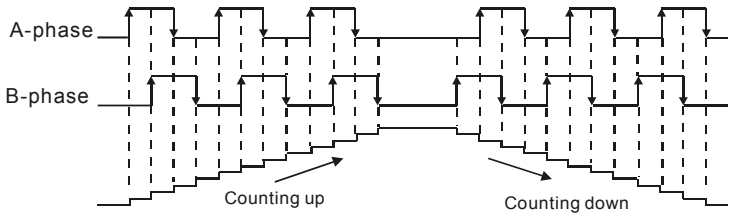
(5) The high-speed comparative instruction DHSCR can clear output devices and counter devices, but only the counters used by the same instruction, e.g. DHSCR K10 C243 C243. This function can only applied in the four special high speed counters C243, C245, C250 and C255.

e) Counting modes:

i) The 2-phase 2 inputs counting mode of the high speed counters in ES/EX/SS (V5.5 and above) and SA/SX/SC series MPU is set by special D1022 with normal frequency, double frequency and 4 times frequency modes. The contents in D1022 will be loaded in in the first scan when PLC is switched from STOP to RUN.

Device No.	Function
D1022	Setting up the multiplied frequency of the counter
D1022 = K1	Normal frequency mode selected
D1022 = K2 or 0	Double frequency mode selected (default)
D1022 = K4	4 times frequency mode selected

ii) Multiplied frequency mode (↑↓ indicates the occurrence of counting)

Counting mode		Counting wave pattern
2-phase 2 inputs	1 (Normal frequency)	
	2 (Double frequency)	
	4 (4 times frequency)	

5. EH/EH2/SV series MPU supports high speed counters. C235 ~ C240 are program-interruption 1-phase high speed counter with a total bandwidth of 20KHz, can be used alone with a counting frequency of up to 10KHz. C241 ~ C254 are hardware high speed counter (HHSC). There are four HHSC in EH/EH2/SV series MPU, HHSC0 ~ 3. The pulse input frequency of HHSC0 and HHSC1 can reach 200KHz and that of HHSC2 and HHSC3 can reach 20KHz (1 phase or A-B phase). The pulse input frequency of HHSC0 ~ 3 of 40EH2 series MPU can reach 200KHz, among which:

   C241, C246 and C251 share HHSC0

   C242, C247 and C252 share HHSC1

   C243, C248 and C253 share HHSC2

   C244, C249 and C254 share HHSC3

   a) Every HHSC can only be designated to one counter by DCNT instruction.

   b) There are three counting modes in every HHSC (see the table below):

      i) 1-phase 1 input refers to “pulse/direction” mode.

      ii) 1-phase 2 inputs refers to “clockwise/counterclockwise (CW/CCW)” mode.

      iii) 2-phase 2 inputs refers to “A-B phase” mode.

Counter type	Program-interruption high speed counter						Hardware high speed counter											
Type / Input	1-phase 1 input						1-phase 1 input				1-phase 2 inputs				2-phase 2 inputs			
	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C246	C247	C248	C249	C251	C252	C253	C254
X0	U/D						U/D				U				A			
X1		U/D									D				B			
X2			U/D				R				R				R			
X3				U/D			S				S				S			
X4					U/D			U/D				U				A		
X5						U/D						D				B		
X6								R				R				R		
X7								S				S				S		
X10									U/D				U				A	
X11													D				B	
X12									R				R				R	
X13									S				S				S	
X14										U/D				U				A
X15														D				B
X16										R				R				R
X17										S				S				S

U: Progressively increasing input A: A phase input S: Input started
B: Progressively decreasing input B: B phase input R: Input cleared

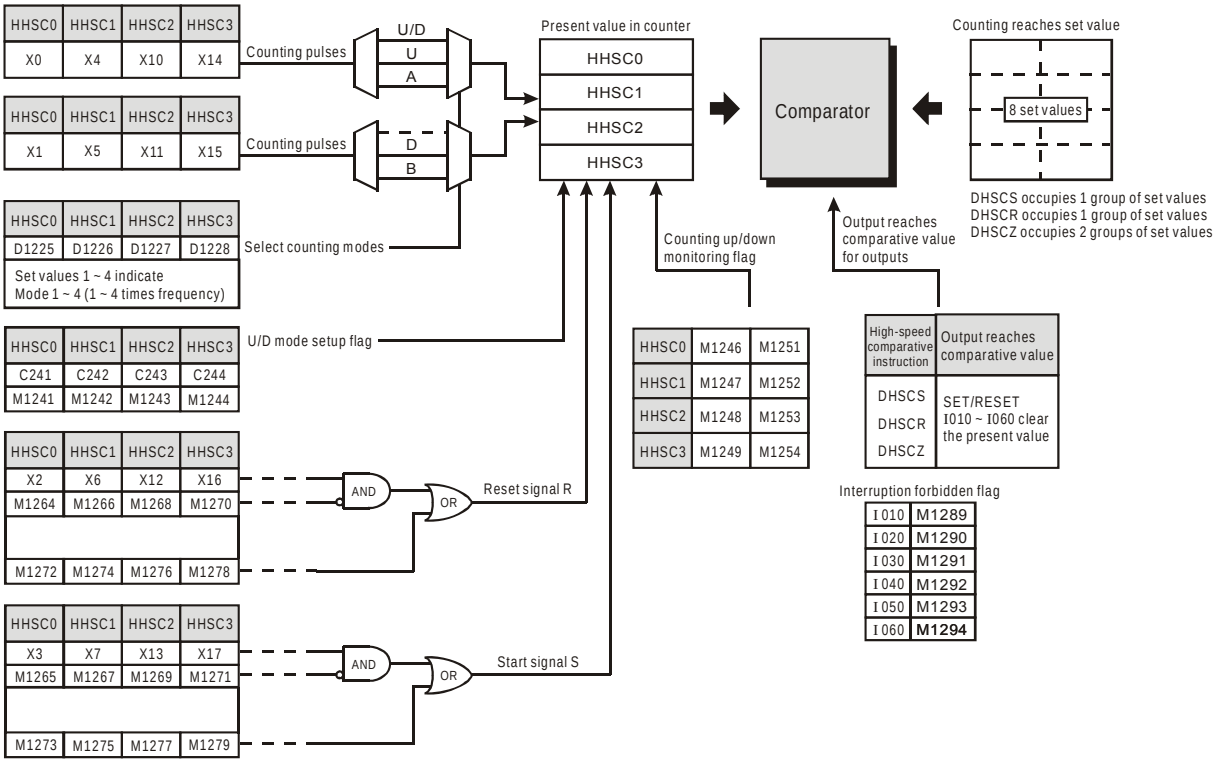
c) In EH/EH2/SV series MPU, there is no limitation on the times of using the hardware high speed counter related instructions, DHSCS, DHSCR and DHSZ. However, when these instructions are enabled at the same time, there will be some limitations. DHSCS instruction will occupy 1 group of settings, DHSCR 1 group of settings and DHSZ 2 groups of settings. There three instructions cannot occupy 8 groups of settings in total; otherwise the system will ignore the instructions which are not the first scanned and enabled.

d) System structure of the hardware high speed counters:

i) HHSC0 ~ 3 have reset signals and start signals from external inputs. Settings in M1272, M1274, M1276 and M1278 are reset signals of HHSC0, HHSC1, HHSC2 and HHSC3. Settings in M1273, M1275, M1277 and M1279 are start signals of HHSC0, HHSC1, HHSC2 and HHSC3.

ii) If the external control signal inputs of R and S are not in use, you can set M1264/M1266/M1268/M1270 and M1265/M1267/M1269/M1271 as True and disable the input signals. The corresponding external inputs can be used again as general input points (see the figure below).

iii) When special M is used as a high speed counter, the inputs controlled by START and RESET will be affected by the scan time.



e) Counting modes:

Special D1225 ~ D1228 are for setting up different counting modes of the hardware high speed counters (HHSC0 ~ 3) in EH/EH2/SV series MPU. There are normal ~ 4 times frequency for the counting and the default setting is double frequency.

Counting modes		Wave pattern	
Type	Set value in special D	Counting up(+1)	Counting down(-1)
1-phase 1 input	1 (Normal frequency)	U/D U/D FLAG	
	2 (Double frequency)	U/D U/D FLAG	
1-phase 2 inputs	1 (Normal frequency)	U D	
	2 (Double frequency)	U D	
2-phase 2 inputs	1 (Normal frequency)	A B	
	2 (Double frequency)	A B	
	3 (Triple frequency)	A B	
	4 (4 times frequency)	A B	

f) Special registers for relevant flags and settings of high speed counters:

Flag	Function
M1150	DHSZ instruction in multiple set values comparison mode
M1151	The execution of DHSZ multiple set values comparison mode is completed.
M1152	Set DHSZ instruction as frequency control mode
M1153	DHSZ frequency control mode has been executed.
M1235 ~ M1245	Designating the counting direction of high speed counters C235 ~ C245 When M12□□ = Off, C2□□ will perform a counting up. When M12□□ = On, C2□□ will perform a counting down.
M1246 ~ M1255	Monitor the counting direction of high speed counters C246 ~ C255 When M12□□ = Off, C2□□ will perform a counting up. When M12□□ = On, C2□□ will perform a counting down.
M1260	X5 as the reset input signal of all high speed counters
M1261	High-speed comparison flag for DHSCR instruction
M1264	Disable the external control signal input point of HHSC0 reset signal point (R)
M1265	Disable the external control signal input point of HHSC0 start signal point (S)
M1266	Disable the external control signal input point of HHSC1 reset signal point (R)
M1267	Disable the external control signal input point of HHSC1 start signal point (S)
M1268	Disable the external control signal input point of HHSC2 reset signal point (R)
M1269	Disable the external control signal input point of HHSC2 start signal point (S)
M1270	Disable the external control signal input point of HHSC3 reset signal point (R)
M1271	Disable the external control signal input point of HHSC3 start signal point (S)
M1272	Internal control signal input point of HHSC0 reset signal point (R)
M1273	Internal control signal input point of HHSC0 start signal point (S)
M1274	Internal control signal input point of HHSC1 reset signal point (R)
M1275	Internal control signal input point of HHSC1 start signal point (S)
M1276	Internal control signal input point of HHSC2 reset signal point (R)
M1277	Internal control signal input point of HHSC2 start signal point (S)
M1278	Internal control signal input point of HHSC3 reset signal point (R)
M1279	Internal control signal input point of HHSC3 start signal point (S)
M1289	High speed counter I010 interruption forbidden
M1290	High speed counter I020 interruption forbidden
M1291	High speed counter I030 interruption forbidden
M1292	High speed counter I040 interruption forbidden
M1293	High speed counter I050 interruption forbidden
M1294	High speed counter I060 interruption forbidden
M1312	C235 Start input point control
M1313	C236 Start input point control

Flag	Function
M1314	C237 Start input point control
M1315	C238 Start input point control
M1316	C239 Start input point control
M1317	C240 Start input point control
M1320	C235 Reset input point control
M1321	C236 Reset input point control
M1322	C237 Reset input point control
M1323	C238 Reset input point control
M1324	C239 Reset input point control
M1325	C240 Reset input point control
M1328	Enable Start/Reset of C235
M1329	Enable Start/Reset of C236
M1330	Enable Start/Reset of C237
M1331	Enable Start/Reset of C238
M1332	Enable Start/Reset of C239
M1333	Enable Start/Reset of C240

Special D	Function
D1022	Multiplied frequency of A-B phase counters for ES/SA series MPU
D1150	Table counting register for DHSZ multiple set values comparison mode
D1151	Register for DHSZ instruction frequency control mode (counting by table)
D1152 (low word) D1153 (high word)	In frequency control mode, DHSZ reads the upper and lower limits in the table counting register D1153 and D1152.
D1166	Switching between rising/falling edge counting modes of X10 (for SC series MPU only)
D1167	Switching between rising/falling edge counting modes of X11 (for SC series MPU only)
D1225	The counting mode of the 1st group counters (C241, C246, C251)
D1226	The counting mode of the 2nd group counters (C242, C247, C252)
D1227	The counting mode of the 3rd group counters (C243, C248, C253)
D1228	The counting mode of the 4th group counters (C244, C249, C254)
D1225 ~ D1228	Counting modes of HHSC0 ~ HHSC3 in EH/EH2/SV series MPU (default = 2) 1: Normal frequency counting mode 2: Double frequency counting mode 3: Triple frequency counting mode 4: 4 times frequency counting mode

API	Mnemonic		Operands	Function	Controllers		
54	D	HSCR	(S1) (S2) (D)	High Speed Counter Reset	ES/EX/SS	SA/SX/SC	EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*		DHSCR: 13 steps
S_2												*				
D		*	*	*								*				

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

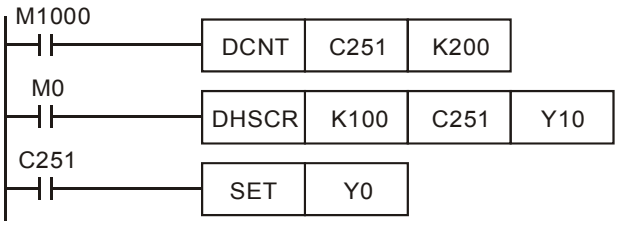
S_1: Comparative value **S_2**: No. of high speed counter **D**: Comparison result

Explanations:

1. **S_2** has to designate the No. of high speed counters C235 ~ C255. See remarks of API 53 DHSCS for more details.
2. **D** of EH/EH2/SV series MPU can designate the No. of high speed counters C241 ~ C254 that are the same as the counters designated by **S_2**.
3. **D** of SC series MPU can designate the No. of high speed counters C243, C245, C250 and C255 that are the same as the counters designated by **S_2**.
4. **D** of ES/EX/SS/SA/SX series MPU does not support device C.
5. See the specifications of each model for their range of use.
6. Flags: M1150 ~ M1333; see remarks of API 53 DHSCS for more details. ES/EX/SS/SA/SX/SC series MPU does not support M1261 (high speed counter external reset mode designation); see remarks for more details.
7. The high speed counter inputs counting pulses from the corresponding external input terminals X0 ~ X17 by inserting an interruption. When the No. of high-speed counter designated in **S_2** "+1" or "-1", DHSCR will perform a comparison immediately. When the present value in the high speed counter equals the comparative value designated in **S_1**, the device designated in **D** will turn Off and even the afterward comparison results are unequal, the device will still be Off.
8. If the devices designated in **D** are Y0 ~ Y17, when the comparative value equals the present value in the high speed counter, the comparison result will immediately output to the external output terminals Y0 ~ Y17 (and clear the designated Y output) and the rest of Y devices will be affected by the scan cycle. Devices M and S act immediately without being affected by the scan cycle.

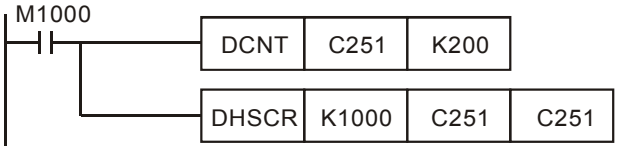
Program Example 1:

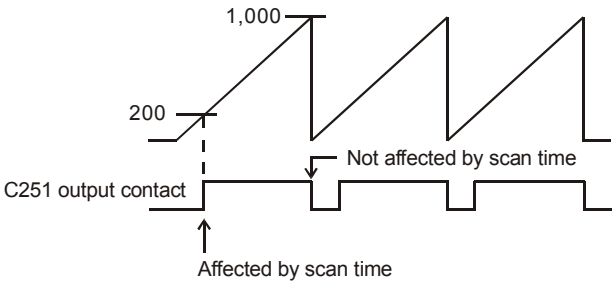
1. When M0 = On and the present value in the high speed counter C251 changes from 99 to 100 or 101 to 100, Y10 will be cleared and Off.
2. When the present value in the high speed counter C251 changes from 199 to 200, the contact of C251 will be On and make Y0 = On. However, the program scan time will delay the output.
3. Y10 will immediately reset the status when the counting reaches its target. **D** is also able to designate high speed counters of the same No. See Program Example 2.



Program Example 2:

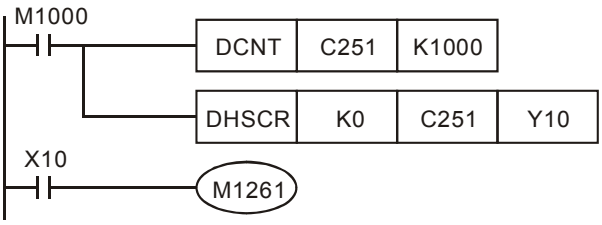
When DHSCR instruction designates the same high speed counter, and the present value in the high speed counter C251 changes from 999 to 1,000 or 1,001 to 1,000, C251 will be reset to Off.





Remarks:

1. DVP all series MPU support high speec counters. For the limitation on the use of instructions, see remarks of API 53 DHSCS for more details.
2. M1261 of EH/EH2/SV series MPU designates the external reset modes of the high speed counter. Some high speec counters have input points for external reset; therefore, when the input point is On, the present value in the corresponding high speed counter will be cleared to 0 and the output contact will be Off. If you wish the reset to be executed immediately by the external output, you have to set M1261 to be On.
3. M1261 can only be used in the hardware high speed counter C241 ~ C255.
4. Example:
   a) X2 is the input point for external reset of C251.
   b) Assume Y10 = On.
   c) When M1261 = Off and X2 = On, the present value in C251 will be cleared to 0 and the contact of C251 will be Off. When DHSCR instruction is executed, there will be no counting input and the comparison result will not output. The external output will not execute the reset; therefore Y10 = On will remain unchanged.
   d) When M1261 = On and X2 = On, the present value in C251 will be cleared to 0 and the contact of C251 will be Off. When DHSCR instruction is executed, there will be no counting input but the comparison result will output. Therefore, Y10 will be reset.



API	Mnemonic	Operands	Function	Controllers
55	D HSZ	(S1) (S2) (S) (D)	High Speed Zone Compare	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*		DHSZ: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*		
S												*				
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

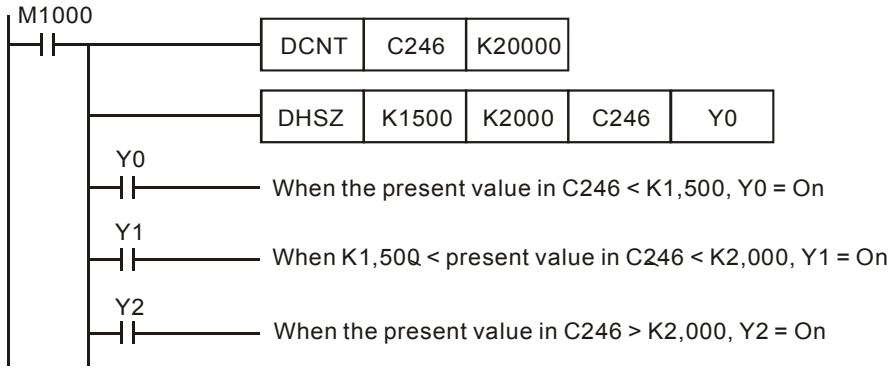
S_1: Lower bound of the comparison zone **S_2**: Upper bound of the comparison zone **S**: No. of high speed counter **D**: Comparison result

Explanations:

1. **S_1** has to be euqal to or smaller than **S_2**. (**$S_1 \leq S_2$**)
2. When **S_1** > **S_2**, the instruction will perform a comparison by using **S_1** as the upper bound and **S_2** as the lower bound.
3. **S** has to designate high speed counters C235 ~ C255, See remarks of API 53 DHSCS for more details.
4. **D** will occupy 3 consecutive devices.
5. Flags: M1150 ~ M1333; see remarks of API 53 DHSCS for more details. M1150, M1151 DHSZ executing multiple points comparison mode; see Program Example 3 for more details; SA/SX/SC series MPU does not support. M1152, M1153 DHSZ as frequency control mode; see Program Example 4 for more details; SA/SX/SC series MPU does not support.
6. The output will not be affected by the scan time.
7. The zone comparisons and outputs are all processed by inserting interruptions.

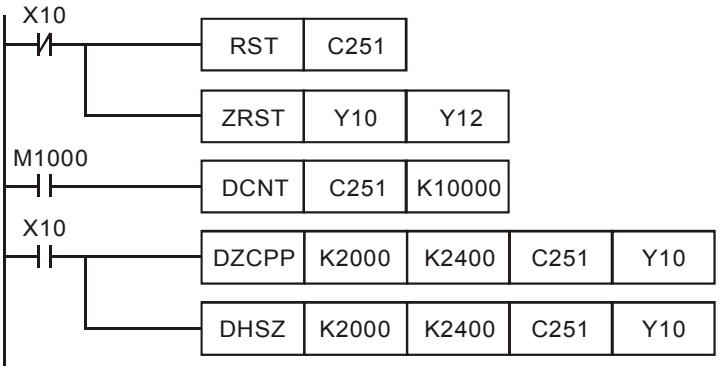
Program Example 1:

1. Designate device Y0 and Y0 ~ Y2 will be automatically occupied.
2. When DHSZ instruction is being executed and the counting of the high speed counter C246 reaches upper and lower bounds, one of Y0 ~ Y2 will be On

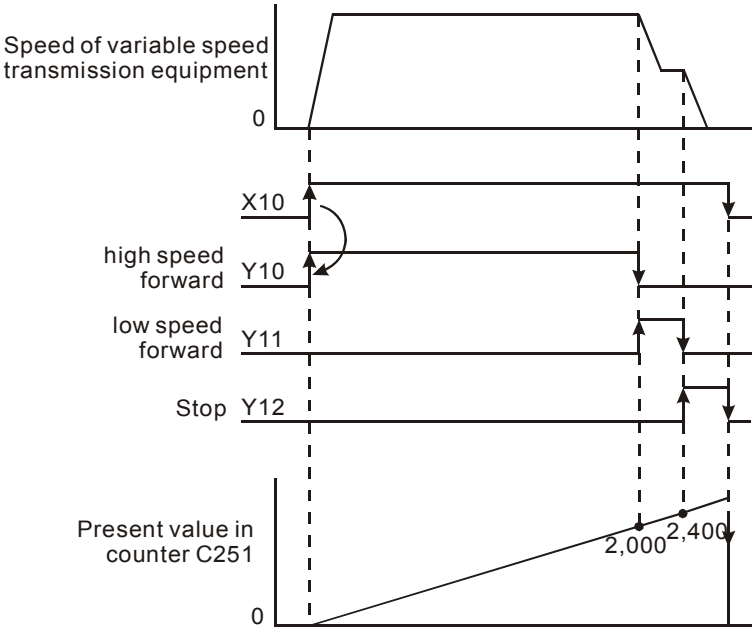


Program Example 2:

1. Use DHSZ instruction for high/low speed stop control. C251 is an A-B phase high speed counter and DHSZ only performs comparison output when there is a C251 counting pulse input. Therefore, even when the present value in the counter is 0, Y10 will not be On.
2. When X10 = On, DHSZ will require that Y10 has to be On when the present value in the counter ≤ K2,000. To solve this requirement, you can execute DZCPP instruction when the program was first RUN and compare C251 with K2,000. When the present value in the counter ≤ K2,000, Y10 will be On. DZCPP instruction is a pulse execution instruction and will only be executed once with Y10 being kept On.
3. When the drive contact X10 = Off, Y10 ~ Y12 will be reset to Off.



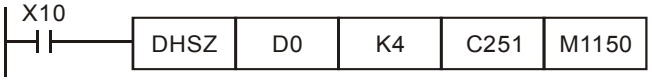
4. The timing diagram



Program Example 3:

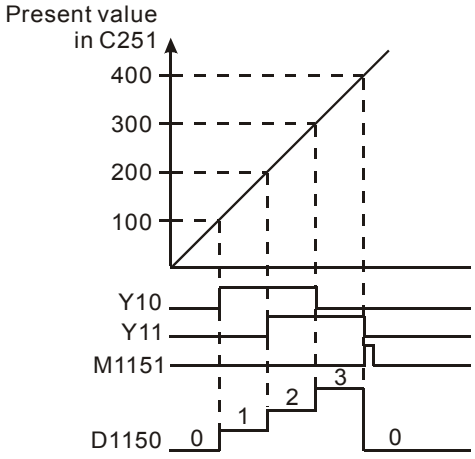
1. Program Example 3 is only applicable to EH/EH2/SV series MPU.
2. The multiple set values comparison mode: If **D** of DHSZ instruction designates a special auxiliary relay M1150, the instruction will be able to compare (output) the present value in the high speed counter with many set values.

3. In this mode,
   - **S_1**: start device in the comparison table. **S_1** can only designate data register D and can be modified by E and F. Once this mode is enabled, **S_1** will not be changed even the E and F has been changed.
   - **S_2**: number of group data to be compared. **S_2** can only designate K1 ~ K255 or H1 ~ HFF and can be modified by E and F. Once this mode is enabled, **S_2** cannot be changed. If **S_2** is not within its range, error code 01EA (hex) will display and the instruction will not be executed.
   - **S**: No. of high speed counter (designated as C241 ~ C254).
   - **D**: Designated mode (can only be M1150)
4. The No. of start register designated in **S_1** and the number of rows (groups) designated in **S_2** construct a comparison table. Please enter the set values in every register in the table before executing the instruction.
5. When the present value in the counter C251 designated in **S** equals the set values in D1 and D0, the Y output designated by D2 will be reset to Off (D3 = K0) or On (D3 = K1) and be kept. Output Y will be processed as an interruption. No. of Y output pointss are in decimal (range: 0 ~ 255). If the No. falls without the range, SET/RESET will not be enabled when the comparison reaches its target.
6. When this mode is enabled, PLC will first acquire the set values in D0 and D1 as the target value for the first comparison section. At the same time, the index value displayed in D1150 will be 0, indicating that PLC performs the comparison based on the group 0 data.
7. When the group 0 data in the table have been compared, PLC will first execute the Y output set in group 0 data and determine if the comparison reaches the target number of groups. If the comparison reaches the target, M1151 will be On; if the comparison has not reached the final group, the content in D1150 will plus 1 and continue the comprison for the next group.
8. M1151 is the flag for the completion of one execution of the table, can be Off by the user. Or when the next comparion cycle takes place and the group 0 data has been compared, PLC will automatically reset the flag.
9. When the drive contact of the instruction X10 goes Off, the execution of the instruction will be interrupted and the content in D1150 (table counting register) will be reset to 0. However, the On/Off status of all outputs will be remained.
10. When the instruction is being executed, all set values in the comparison table will be regarded as valid values only when the scan arrives at END instruction for the first time.
11. This mode can only be used once in the program.
12. This mode can only be used on the hardware high speed counters C241 ~ C254.
13. When in this mode, the frequency of the input counting pulses cannot exceed 50KHz or the neighboring two groups of comparative values cannot differ by 1; otherwise there will not be enough time for the PLC to react and result in errors.



The comparison table:

32-bit data for comparison		No. of Y output	On/Off indication	Table counting register D1150
High word	Low word			
D1 (K0)	D0 (K100)	D2 (K10)	D3 (K1)	0
D5 (K0)	D4 (K200)	D6 (K11)	D7 (K1)	1
D9 (K0)	D8 (K300)	D10 (K10)	D11 (K0)	2
D13 (K0)	D12 (K400)	D14 (K11)	D15 (K0)	3
		K10: Y10 K11: Y11	K0: Off K1: On	0→1→2→3→0 Cyclic scan



14. Special registers for flags and relevant settings:

Flag	Function
M1150	DHSZ instruction in multiple set values comparison mode
M1151	The execution of DHSZ multiple set values comparison mode is completed.

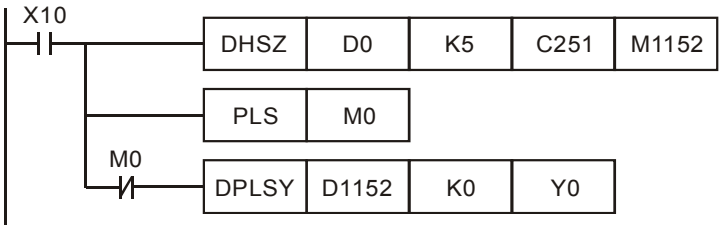
Special D	Function
D1150	Table counting register for DHSZ multiple set values comparison mode

Program Example 4:

1. Program Example 4 is only applicable to EH/EH2/SV series MPU.
2. DHSZ and DPLSY instructions are combined for frequency control. If **D** of DHSZ instruction is a special auxiliary relay M1152, the present value in the counter will be able to control the pulse output frequency of DPLSY instruction.
3. In this mode,
   - **S_1**: start device in the comparison table. **S_1** can only designate data register D and can be modified by E and F. Once this mode is enabled, **S_1** will not be changed even the E and F has been changed.
   - **S_2**: number of group data to be compared. **S_2** can only designate K1 ~ K255 or H1 ~ HFF and can be modified by E and F. Once this mode is enabled, **S_2** cannot be changed. If **S_2** is not within its range, error code 01EA (hex) will display and the instruction will not be executed.
   - **S**: No. of high speed counter (designated as C241 ~ C254).

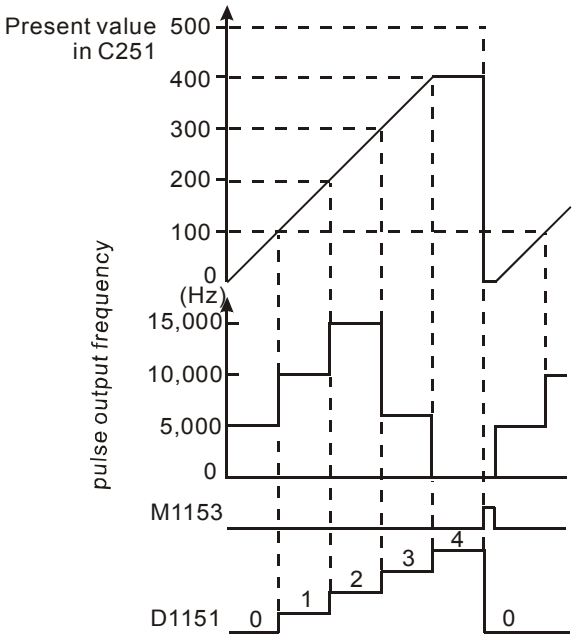
- **D**: Designated mode (can only be M1152)

4. This mode can only be used once. For EH/EH2/SV series MPU, this mode can only be used in the hardware high speed counter C241 ~ C254. Please enter the set values in every register in the table before executing the instruction.
5. When this mode is enabled, PLC will first acquire the set values in D0 and D1 as the target value for the first comparison section. At the same time, the index value displayed in D1152 will be 0, indicating that PLC performs the comparison based on the group 0 data.
6. When the group 0 data in the table have been compared, PLC will first execute at the frequency set in group 0 data (D2, D3) and copy the data to D1152 and D1153, determining if the comparison reaches the target number of groups. If the comparison reaches the target, M1153 will be On; if the comparison has not reached the final group, the content in D1151 will plus 1 and continue the comprison for the next group.
7. M1153 is the flag for the completion of one execution of the table, can be Off by the user. Or when the next comparion cycle takes place and the group 0 data has been compared, PLC will automatically reset the flag.
8. If you wish to use this mode with PLSY instruction, please preset the value in D1152.
9. If you wish to stop the execution at the last row, please set the value in the last row K0.
10. When the drive contact of the instruction X10 goes Off, the execution of the instruction will be interrupted and the content in D1151 (table counting register) will be reset to 0.
11. When in this mode, the frequency of the input counting pulses cannot exceed 50KHz or the neighboring two groups of comparative values cannot differ by 1; otherwise there will not be enough time for the PLC to react and result in errors.



The comparison table:

32-bit data for comparison		Pulse output frequency 0 ~ 200KHz	Table counting register D1151
High word	Low word		
D1 (K0)	D0 (K0)	D3, D2 (K5,000)	0
D5 (K0)	D4 (K100)	D7, D6 (K10,000)	1
D9 (K0)	D8 (K200)	D11, D10 (K15,000)	2
D13 (K0)	D12 (K300)	D15, D14 (K6,000)	3
D17 (K0)	D16 (K400)	D19, D18 (K0)	4
			0→1→2→3→4 Cyclic scan

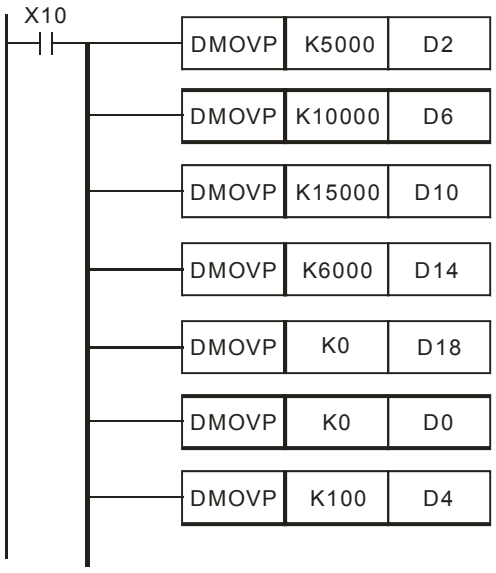


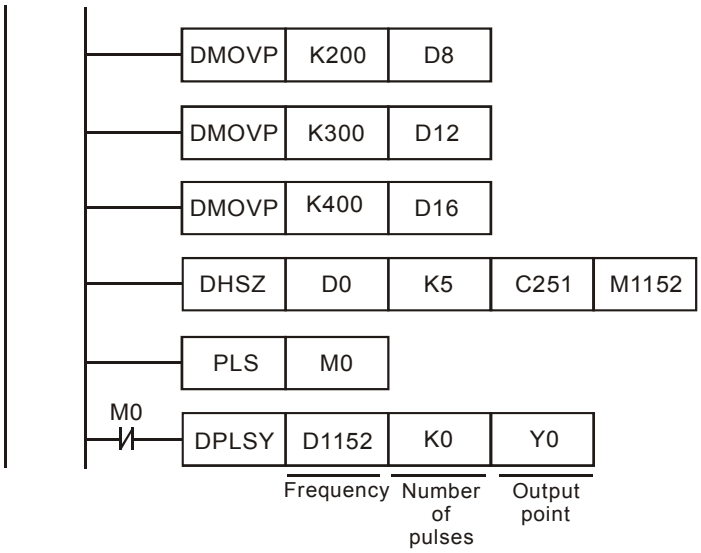
12. Special registers for flags and relevant settings:

Flag	Function
M1152	DHSZ instruction in frequency control mode
M1153	The execution of DHSZ frequency control mode is completed.

Special D	Function
D1151	Table counting register for DHSZ multiple set values comparison mode
D1152 (low word) D1153 (high word)	In frequency control mode, DHSZ reads the upper and lower limits in the table counting register D1153 and D1152.
D1336 (low word) D1337 (high word)	Current number of pulses output by DPLSY instruction

13. The complete program:





14. During the execution of DHSZ instruction, do not modify the set values in the comparison table.
15. The designated data will be arranged into the the above program diagram when the program executes to END instruction. Therefore, PLSY instruction has to be executed after DHSZ instruction has been executed once.

API	Mnemonic	Operands	Function	Controllers
56	SPD	(S1) (S2) (D)	Speed Detection	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1	*															SPD: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D											*	*	*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: External pulse input terminal **S_2**: Pulse receiving time (ms) **D**: Detected result

Explanations:

1. See the specifications of each model for their range of use.
2. Flag: M1100 (SPD instruction performs sampling for one time)
3. External pulse input terminals designated in **S_1** for all series MPU:

Input \ MPU	ES/EX/SS (V5.7and above)	SA/SX/SC	EH/EH2/SV
Available input points	X1, X2	X0, X1, X2	X0 ~ X3

4. For SA/SX (V1.4 and above) series MPU and SC (V1.2 and above) series MPU, the new X0 and X1 can be used together with A-B phase input points. When “A ahead of B” detection result is a positive value and “B ahead of A” detection result is a negative value, the multiplied frequency of the counter can be set by D1022.
5. The received number of pulses of the input terminal designated in **S_1** is calculated within the time (in ms) designated in **S_2**. The result is stored in the register designated in **D**.
6. **D** will occupy 5 consecutive devices. **D** + 1 and **D** are the detected value obtained from the previous pulses; **D** +3 and **D** + 2 are the current accumulated number of values; **D** + 4 is the counting time remaining (max. 32,767ms).
7. Pulse frequency detection for all series:

MPU	ES/EX/SS (V5.7 and above)	SA/SX/SC	EH/EH2/SV
Max. frequency	X1 (20KHz), X2 (10KHz)	X0/X1 A-B phase input (4KHz) X1 (30KHz), X2 (10KHz)	X0/X1 (100KHz) X2/X3 (10KHz)

8. This instruction is mainly used for obtaining a proportional value of rotation speed. The result **D** and rotation speed will be in proportion. The following equation is for obtaining the rotation speed of motor.

$$N=\frac{60(D0)}{nt}\times 10^3(rpm)$$

N: Rotation speed
n: The number of pulses produced per rotation
t: Detecting time designated in **S_2** (ms)

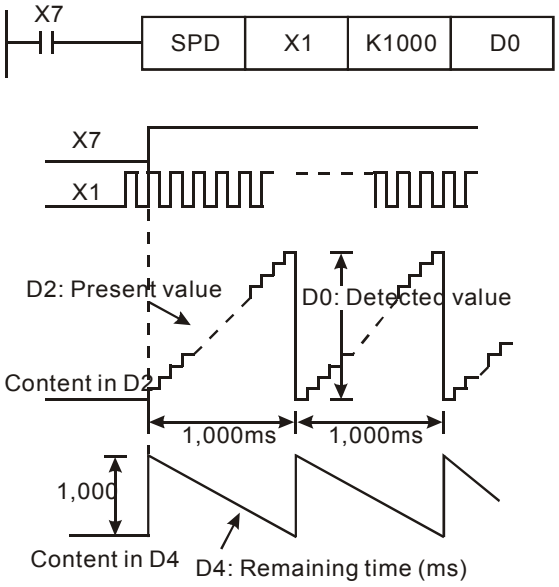
9. The X input point designated by this instruction cannot be used again as the pulse input terminal of the high speed counter or as an external interruption signal.
10. When M1036 in SC (V1.4 and above) series MPU is enabled, SPD instruction can detect the speeds at X0 ~ X5

at the same time with a total bandwidth of 40KHz. See 2.11 for more details for how to use M1036.

11. There is no limitation on the times of using this instruction in the program, but only one instruction will be executed at a time.
12. When SPD instruction is enabled and M1100 = On, SPD instruction will perform a sampling at the moment when M1100 goes from Off to On and stop the sampling. If you wish to resume the sampling, you have to turn Off M1100 and re-enable SPD instruction.

Program Example:

1. When X7 = On, D2 will calculate the high-speed pulses input by X1 and stop the calculation automatically after 1,000ms. The result will be stored in D0.
2. When the 1,000ms counting is completed, D2 will be cleared to 0. When X7 is On again, D2 will start the calculation again.



Remarks:

1. When ES/EX/SS (V5.7 and above) and SA/SX/SC series MPU use X1 or X2, the relevant high speed counters or external interruptions I101 and I201 cannot be used.
2. For SC (V1.4 and above) series MPU, when M1036 is enabled, the speed of X0 ~ X5 can be detected at the same time.

API	Mnemonic	Operands	Function	Controllers
57	D PLSY	S_1 S_2 D	Pulse Y Output	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	PLSY: 7 steps DPLSY: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Pulse output frequency **S_2**: Number of output pulses **D**: Pulse output device (please use transistor output module)

Explanations:

1. The program of ES/EX/SS series MPU can use PLSY instruction two times but cannot designate the same Y device.
2. Flags: M1010 ~ M1345. See remarks for more details.
3. **S_1** designates the pulse output frequency. With M1133 ~ M1135 and D1133, Y0 of SA/SX series MPU is able to output pulses at 50KHz. See 2.11 for M1133 ~ M1135 and D1133.

   Range of output frequency for all series:

MPU	ES/EX/SS	SA/SX	SC	EH	EH2/SV
Frequency range	Y0: 0 ~10KHz Y1: 0 ~10KHz	Y0: 0 ~ 32KHz Y1: 0 ~10KHz	Y0: 0 ~ 30KHz Y1: 0 ~ 30KHz Y10: 77 ~ 100KHz Y11: 77 ~ 100KHz	Y0: 1 ~ 200KHz Y2: 1 ~ 200KHz	Y0: 0 ~ 200KHz Y2: 0 ~ 200KHz Y4: 0 ~ 200KHz Y6: 0 ~ 200KHz

4. **S_2** designates the number of output pulses. The 16-bit instruction can designate 1 ~ 32,767 pulses and the 32-bit instruction can designate 1 ~ 2,147,483,647 pulses.

   Number of continuous pulses for all series:

MPU	ES/EX/SS/SA/SX/SC	SC	EH/EH2/SV
How to designate continuous pulses	M1010 (Y0) On M1023 (Y1) On	M1010 (Y0) On M1023 (Y1) On The number of output pulses designated for Y10 and Y11 is set to K0.	The number of output pulses designated for Y0, Y2, Y4 and Y6 is set to K0

5. For EH/EH2/SV series MPU, when the number of output pulses is set to 0, there will be continuous pulse output with no limitation on the number of pulses. For ES/EX/SS/SA/SX/SC series MPU, you have to make M1010 (Y0) or M1023 (Y1) On to allow a continuous pulse output with no limitation on the number of pulses.
6. For the pulse output device designated in D, EH series MPU can designate Y0 and Y2, EH2/SV series MPU can designate Y0, Y2, Y4 and Y6, ES/EX/SS/SA/SX series MPU can designate Y0 and Y1, SC series MPU can designate Y0, Y1, Y10 and Y11. (SC V1.2 and above series MPU supports Y10 and Y11).

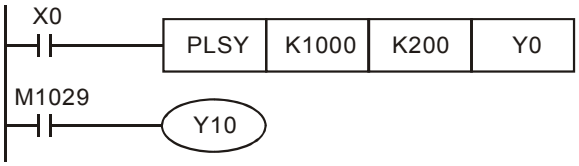
7. EH series MPU has two groups of A-B phase pulse output from CH0 (Y0, Y1) and CH1 (Y2, Y3); EH2/SV series MPU has four groups of A-B phase pulse output from CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See 2.3 and remarks for how to set up.
8. When PLSY instruction is executed, it will designate the number of output pulses ($\mathbf{S_2}$) output from the output device (**D**) at a pulse output frequency ($\mathbf{S_1}$).
9. When PLSY instruction is used in the program, its outputs cannot be the same as those in API 58 PWM and API 59 PLSR.
10. Pulse output completed flags for all series:

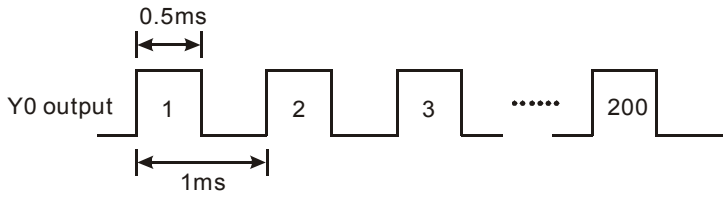
MPU	ES/EX/SS SA/SX/SC		SC		EH/EH2/SV		EH2/SV	
Output device	Y0	Y1	Y10	Y11	Y0	Y2	Y4	Y6
Flag	M1029	M1030	M1102	M1103	M1029	M1030	M1036	M1037

11. For ES/EX/SS/SA/SX/SC/EH series MPU, when PLSY and DPLSY instruction is disabled, the pulse output completed flags will all be Off automatically.
12. For EH2/SV series MPU, when PLSY and DPLSY instruction is disabled, the user will have to reset the pulse output completed flags.
13. The user has to reset the pulse output completed flags after the pulse output is completed.
14. After PLSY instruction starts to be executed, Y will start a pulse output. Modifying $\mathbf{S_2}$ at this moment will not affect the current output. If you wish to modify the number of output pulses, you have to first stop the execution of PLSY instruction and modify the number.
15. $\mathbf{S_1}$ can be modified when the program executes to PLSY instruction.
16. Off time : On time of the pulse output = 1 : 1.
17. When the program executes to PLSY instruction, the current number of output pulses will be stored in the special data registers D1336 ~ D1339. See remarks for more details.
18. For SA/EH series MPU, there is no limitation on the times using this instruction. For SA/SX/SC/EH series MPU, the program allows two instructions being executed at the same time. For EH2/SV series MPU, the program allows four instructions being executed at the same time.

Program Example:

1. When X0 = On, there will be 200 pulses output from Y0 at 1KHz. When the pulse output is completed, M1029 will be On and Y10 will be On.
2. When X0 = Off, the pulse output from Y0 will stop immediately. When X0 is On again, the output will start again ffrom the first pulse.





Remarks:

1. Flags and special registers for ES/EX/SS series MPU:
   - M1010: When On, Y0 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y0 will be decided by S_2.
   - M1023: When On, Y1 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y1 will be decided by S_2.
   - M1029: On when Y0 pulse output is completed.
   - M1030: On when Y1 pulse output is completed.
   - M1078: Y0 output pauses.
   - M1079: Y1 output pauses.
   - D1030: Low word of the current number of output pulses from Y0
   - D1031: High word of the current number of output pulses from Y0
   - D1032: Low word of the current number of output pulses from Y1
   - D1033: High word of the current number of output pulses from Y1
2. Flags and special registers for SA/SX/SC series MPU:
   - M1010: (SA/SX/SC) When On, Y0 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y0 will be decided by S_2.
   - M1023: (SA/SX/SC) When On, Y1 output will be continuous with no limitation on the number of pulses. When Off, the number of output pulses from Y1 will be decided by S_2.
   - M1029: (SA/SX/SC) On when Y0 pulse output is completed.
   - M1030: (SA/SX/SC) On when Y1 pulse output is completed.
   - M1078: (SA/SX/SC) Y0 output pauses.
   - M1079: (SA/SX/SC) Y1 output pauses.
   - M1102: (SC) On when Y10 pulse output is completed.
   - M1103: (SC) On when Y11 pulse output is completed.
   - D1030: (SA/SX/SC) Low word of the current number of output pulses from Y0
   - D1031: (SA/SX/SC) High word of the current number of output pulses from Y0
   - D1032: (SA/SX/SC) Low word of the current number of output pulses from Y1
   - D1033: (SA/SX/SC) High word of the current number of output pulses from Y1
   - D1348: (SC) Low word of the current number of output pulses from Y10
   - D1349: (SC) High word of the current number of output pulses from Y10
   - D1350: (SC) Low word of the current number of output pulses from Y11
   - D1351: (SC) High word of the current number of output pulses from Y11

3. Flags and special registers for EH/EH2/SV series MPU:

M1010: (EH/EH2/SV) When On, CH0, CH1, CH2 and CH3 will output pulses at END instruction. Off when the output starts.

M1029: (EH/EH2/SV) On when CH0 pulse output is completed.

M1030: (EH/EH2/SV) On when CH1 pulse output is completed.

M1036: (EH2/SV) On when CH2 pulse output is completed.

M1037: (EH2/SV) On when CH3 pulse output is completed.

M1334: (EH/EH2/SV) CH0 pulse output pauses.

M1335: (EH/EH2/SV) CH1 pulse output pauses.

M1520: (EH2/SV) CH2 pulse output pauses.

M1521: (EH2/SV) CH3 pulse output pauses.

M1336: (EH/EH2/SV) CH0 pulse output has been sent.

M1337: (EH/EH2/SV) CH1 pulse output has been sent.

M1522: (EH2/SV) CH2 pulse output has been sent.

M1523: (EH2/SV) CH3 pulse output has been sent.

M1338: (EH/EH2/SV) CH0 offset pulses enabled.

M1339: (EH/EH2/SV) CH1 offset pulses enabled.

M1340: (EH/EH2/SV) I110 interruption occurs after CH0 pulse output is completed.

M1341: (EH/EH2/SV) I120 interruption after occurs CH1 pulse output is completed.

M1342: (EH/EH2/SV) I130 interruption occurs when CH0 pulse output is sending.

M1343: (EH/EH2/SV) I140 interruption occurs when CH0 pulse output is sending.

M1344: (EH/EH2/SV) CH0 pulse compensation enabled.

M1345: (EH/EH2/SV) CH1 pulse compensation enabled.

M1347: (EH/EH2/SV) CH0 pulse output reset flag

M1348: (EH/EH2/SV) CH1 pulse output reset flag

M1524: (EH2/SV) CH2 pulse output reset flag

M1525: (EH2/SV) CH3 pulse output reset flag

D1220: (EH/EH2/SV) Phase setting of CH0 (Y0, Y1): D1220 determines the phase by the last two bits; other bits are invalid.

1. K0: Y0 output
2. K1: Y0, Y1 AB-phase output; A ahead of B.
3. K2: Y0, Y1 AB-phase output; B ahead of A.
4. K3: Y1 output

D1221: (EH/EH2/SV) Phase setting of CH1 (Y2, Y3): D1221 determines the phase by the last two bits; other bits are invalid.

1. K0: Y2 output
2. K1: Y2, Y3 AB-phase output; A ahead of B.
3. K2: Y2, Y3 AB-phase output; B ahead of A.
4. K3: Y3 output

D1229: (EH2/SV) Phase setting of CH2 (Y4, Y5): D1229 determines the phase by the last two bits; other bits are invalid.

1. K0: Y4 output
2. K1: Y4, Y5 AB-phase output; A ahead of B.
3. K2: Y4, Y5 AB-phase output; B ahead of A.
4. K3: Y5 output

D1230: (EH2/SV) Phase setting of CH3 (Y6, Y7): D1230 determines the phase by the last two bits; other bits are invalid.

1. K0: Y6 output
2. K1: Y6, Y7 AB-phase output; A ahead of B.
3. K2: Y6, Y7 AB-phase output; B ahead of A.
4. K3: Y7 output

D1328: (EH/EH2/SV) Low word of the number of CH0 offset pulses

D1329: (EH/EH2/SV) High word of the number of CH0 offset pulses

D1330: (EH/EH2/SV) Low word of the number of CH1 offset pulses

D1331: (EH/EH2/SV) High word of the number of CH1 offset pulses

D1332: (EH/EH2/SV) Low word of the number of remaining pulses at CH0

D1333: (EH/EH2/SV) High word of the number of remaining pulses at CH0

D1334: (EH/EH2/SV) Low word of the number of remaining pulses at CH1

D1335: (EH/EH2/SV) High word of the number of remaining pulses at CH1

D1336: (EH/EH2/SV) Low word of the current number of output pulses at CH0

D1337: (EH/EH2/SV) High word of the current number of output pulses at CH0

D1338: (EH/EH2/SV) Low word of the current number of output pulses at CH1

D1339: (EH/EH2/SV) High word of the current number of output pulses at CH1

D1375: (EH2/SV) Low word of the current number of output pulses at CH2

D1376: (EH2/SV) High word of the current number of output pulses at CH2

D1377: (EH2/SV) Low word of the current number of output pulses at CH3

D1378: (EH2/SV) High word of the current number of output pulses at CH3

D1344: (EH/EH2/SV) Low word of the number of compensation pulses at CH0

D1345: (EH/EH2/SV) High word of the number of compensation pulses at CH0

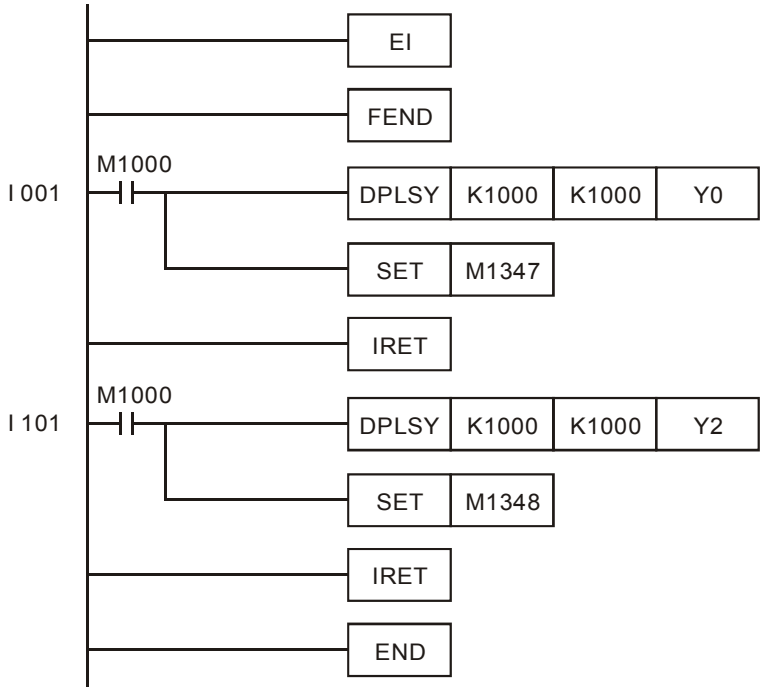
D1346: (EH/EH2/SV) Low word of the number of compensation pulses at CH1

D1347: (EH/EH2/SV) High word of the number of compensation pulses at CH1

4. When there are many high speed output instructions (PLSY, PWM, PLSR) for Y0 output in a program, PLC will only execute the settings and outputs of the instruction that is first enabled.
5. More explanations on M1347 and M1348:

   If M1347 and M1348 is enabled, and when the execution of PLSY instruction has been completed, M1347/M1348 will be reset automatically, i.e. you do not have to turn the status of the drive contact from Off to On before PLSY instruction and when PLC scans to the instruction (assume the drive contact of the instruction is True), there will still be pulse output. PLC detects the status of M1347 and M1348 when END instruction is being executed. Therefore, when the pulse output is completed and if PLSY instruction is a continuous execution one, there will be a scan time of delay in the next string of pulse output.

Program Example 1:

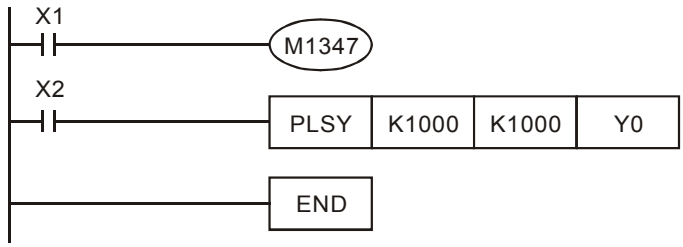


Explanations:

a) Whenever X0 is triggered, Y0 will output 1,000 pulses; whenever X1 is triggered, Y2 will output 1,000 pulses.

b) When X triggers Y pulse output, there should be an interval of at least one scan time between the end of Y pulse output and the next X-triggered output.

Program Example 2:



Explanations:

When both X1 and X2 are On, Y0 pulse output will keep operating. However, there will be a short pause (approx. 1 scan time) every 1,000 pulses before the output of the next 1,000 pulses.

API	Mnemonic	Operands	Function	Controllers
58	PWM	(S1) (S2) (D)	Pulse Width Modulation	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	PWM: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Pulse output width **S_2**: Pulse output period **D**: Pulse output device (please use transistor output module)

Explanations:

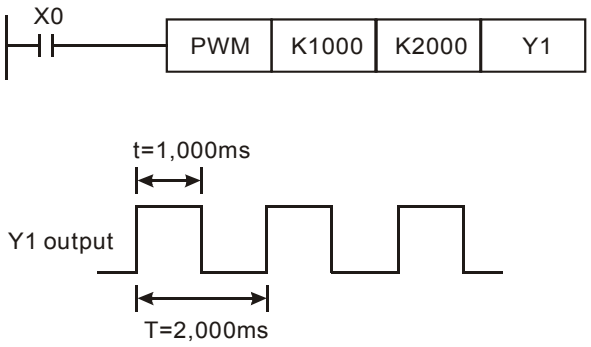
1. **S_1 ≤ S_2**.
2. See the specifications of each model for their range of use.
3. In ES/EX/SS series MPU, PWM instruction can only be used once in the program.
4. Flags: See remarks for more details.
5. Range of **S_1**: (t) 0 ~ 32,767ms.
6. Range of **S_2**: (T) 1 ~ 32,767ms (but **S_1 ≤ S_2**).
7. **D** for all series MPU:

MPU	ES/EX/SS/SA/SX/SC	EH	EH2/SV
Output point	Y1	Y0, Y2	Y0, Y2, Y4, Y6

8. When PWM instruction is used in the program, its outputs cannot be the same as those of API 57 PLSY and API 59 PLSR.
9. PWM instruction designates the pulse output width in **S_1** and pulse output period in **S_2** and outputs from output device **D**.
10. For SA/SX/SC series MPU, When, **S_1** ≤ 0 or **S_2** ≤ 0 or **S_1** > **S_2**, there will be operational errors (M1067 and M1068 will not be On), and there will be no output from the pulse output device. When **S_1** = **S_2**, the pulse output device will keep being On.
11. For EH/EH2/SV series MPU, When, **S_1** < 0 or **S_2** ≤ 0 or **S_1** > **S_2**, there will be operational errors (M1067 and M1068 will be On), and there will be no output from the pulse output device. When **S_1** = 0, M1067 and M1068 will not be On and there will be no output from the pulse output device. When **S_1** = **S_2**, the the pulse output device will keep being On.
12. **S_1** and **S_2** can be changed when PWM instruction is being executed.
13. For SA/EH series MPU, there is no limitation on the times using this instruction in the program. However, for SA/SX/SC/EH series MPU, two instructions are allowed to be executed at the same time; for EH2/SV series MPU, four instructions are allowed to be executed at the same time.

Program Example:

When X0 = On, Y1 will output the pulses as below. When X0 = Off, Y1 output will also be Off.



Remarks:

1. Flags for ES/EX/SS/SA/SX/SC series MPU:
   M1070: Y1 pulse output time unit switch. When Off: 1ms; when On: 100us
2. Flags and special registers for EH/EH2/SV series MPU:
   M1010: (EH/EH2/SV) When On, CH0, CH1, CH2 and CH3 will output pulses when END instruction is executed. Off when the output starts.
   M1070: (EH/EH2/SV) The setting of time unit of CH0 has to work with D1371.
   M1071: (EH/EH2/SV) The setting of time unit of CH1 has to work with D1372.
   M1258: (EH/EH2/SV) CH0 pulse output signals reverse.
   M1259: (EH/EH2/SV) CH1 pulse output signals reverse.
   M1334: (EH/EH2/SV) CH0 pulse output pauses.
   M1335: (EH/EH2/SV) CH1 pulse output pauses.
   M1336: (EH/EH2/SV) CH0 pulse output has been sent.
   M1337: (EH/EH2/SV) CH1 pulse output has been sent.
   M1520: (EH2/SV) CH2 pulse output pauses.
   M1521: (EH2/SV) CH3 pulse output pauses.
   M1522: (EH2/SV) CH2 pulse output has been sent.
   M1523: (EH2/SV) CH3 pulse output has been sent.
   M1526: (EH2/SV) CH2 pulse output signals reverse.
   M1527: (EH2/SV) CH3 pulse output signals reverse.
   M1530: (EH2/SV) The setting of time unit of CH2 has to work with D1373.
   M1531: (EH2/SV) The setting of time unit of CH3 has to work with D1374.
   D1336: (EH/EH2/SV) Low word of the current number of output pulses from CH0.
   D1337: (EH/EH2/SV) High word of the current number of output pulses from CH0.
   D1338: (EH/EH2/SV) Low word of the current number of output pulses from CH1.
   D1339: (EH/EH2/SV) High word of the current number of output pulses from CH1.
   D1371: (EH/EH2/SV) Time unit of CH0 output pulses when M1070 = On.
   D1372: (EH/EH2/SV) Time unit of CH1 output pulses when M1071 = On.

D1373: (EH2/SV) Time unit of CH2 output pulses when M1530 = On.

D1374: (EH2/SV) Time unit of CH3 output pulses when M1531 = On.

D1375: (EH2/SV) Low word of the current number of output pulses from CH2.

D1376: (EH2/SV) High word of the current number of output pulses from CH2.

D1377: (EH2/SV) Low word of the current number of output pulses from CH3.

D1378: (EH2/SV) High word of the current number of output pulses from CH3.

3. Time unit settings for EH/EH2/SV series MPU:

   You cannot modify M1070 in the program.

   D1371, D1372, D1373 and D1374 determine the time unit of the output pulses from CH0, CH1, CH2 and CH3 and the default setting is K1. If your set value is not within the range, the default value will be adopted.

D1371, D1372, D1373, D1374	K0	K1	K2	K3
Time unit	10us	100us	1ms	10ms

4. When there are many high speed pulse output instructions (PLSY, PWM, PLSR) in a program for Y0 output, and provided these instructions are being executed in the same scan period, PLC will set up and output the instructions with the fewest steps.

API	Mnemonic	Operands	Function	Controllers
59	D PLSR	(S1) (S2) (S3) (D)	Pulse Ramp	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	PLSR: 9 steps DPLSR: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
S_3					*	*	*	*	*	*	*	*	*	*	*	
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Maximum speed of pulse output **S_2**: Total number of output pulses **S_3**: Acceleration/deceleration time (ms)

D: Pulse output device (please use transistor output module)

Explanations:

1. See the specifications of each model for their range of use.
2. For ES/EX/SS series MPU, PLSR instruction can be used twice in the program but the outputs cannot be overlapped.
3. Flags: See remarks of API 57 PLSY.
4. Range of **S_1**: 10 ~ 32,767Hz (16-bit); 10 ~ 200,000Hz (32-bit). The maximum speed has to be 10's multiple; if not, the 1s digit will be left out. 1/10 of the maximum speed is the variation of one acceleration or deleration. Please be aware if the variation reponds to the acceleration/deceleration demand from the step motor, in case the step motor may crash.
5. Range of **S_2**: 110 ~ 32,767 (16-bit); 110 ~ 2,147,483,647 (32-bit). If **S_2** is less than 110, the pulet output will be abnormal.
6. Range of **S_3**: below 5,000ms. The acceleration time and deceleration time have to be the same.
   a) The acceleration/deceleration time has to be 10 times longer than the maximum scan time (D1012). If not, the slope of accleration and deceleration will be incorrect.
   b) The minimum set value of acceleration/deceleration time can be obtained from the following equation:

$$S_3 \geqslant \frac{90{,}000}{S_1}$$

   If the set value is less than the result obtained from the equation, the acceleration/deceleration time will be longer. If the set value is less than 90,000/**S_1**, use the result of 90,000/**S_1** as the set value.
   c) The maximum set value of acceleration/deceleration time can be obtained from the following equation:

$$S_3 \leqslant \frac{S_2}{S_1} \times 818$$

   d) The speed variation is fixed to 10 steps. If the input acceleration/deceleration time is longer than the maximum set value, the acceleration/deceleration time will follow the maximum set time. If shorter than the minimum set value, the accleration/deceleration time will follow the minimum set time.

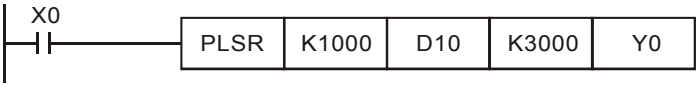
7. **D** for all series MPU:

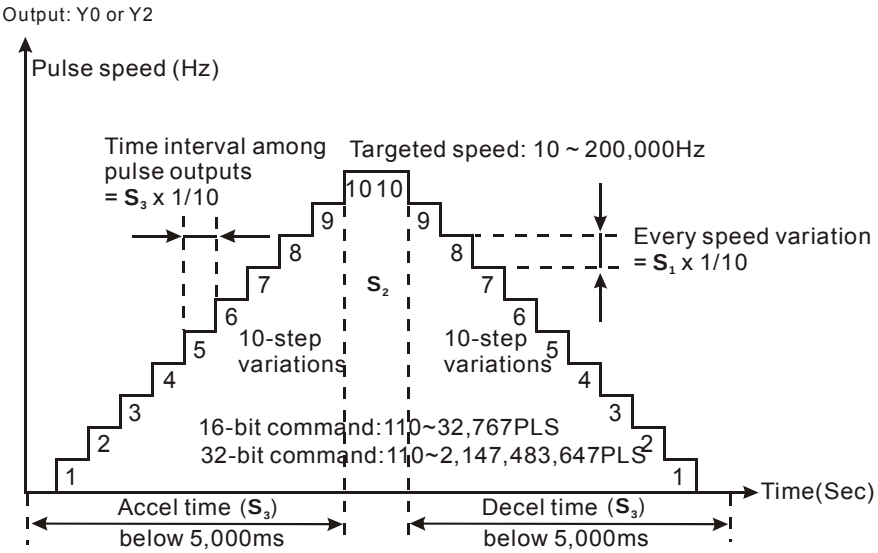
MPU	ES/EX/SS/SA/SX/SC	EH	EH2/SV
Output point	Y0, Y1	Y0, Y2	Y0, Y2, Y4, Y6

8. EH series MPU has two groups pf A-B phase pulse output CH0 (Y0, Y1) and CH1 (Y2, Y3). EH2/SV series MPU has four groups pf A-B phase pulse output CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks of API 57 PLSY for how to set up.
9. PLSR instruction is a pulse output instruction with acclerating and decelerating functions. The pulses accelerate from the static status to target speed and decelerates when the target distance is nearly reached. The pulse output will stop when the target distance is reached.
10. When PLSR instruction is executed, after $\mathbf{S_1}$, $\mathbf{S_2}$ and $\mathbf{S_3}$ are set, the pulses will output from **D**. The output starts at the frequency of increasing $\mathbf{S_1}$/10 at a time. The time forf every frequency is fixed at $\mathbf{S_3}$/9.
11. $\mathbf{S_1}$, $\mathbf{S_2}$ and $\mathbf{S_3}$ can be changed when PLSR instruction is being executed.
12. For ES/EX/SS/SA/SX/SC series MPU, when all the Y0 pulses have been sent, M1029 will be On; when all the Y1 pulses have been sent, M1030 will be On. Next time when PLSR instruction is enabled, M1029 or M1030 will be 0 again and after the pulse output is completed, it will become 1 again.
13. For EH/EH2/SV series MPU, when all the CH0 (Y0, Y1) pulses have been sent, M1029 will be On; when all the CH1 (Y2, Y3) pulses have been sent, M1030 will be On; when CH2 (Y4, Y5) pulses have been sent, M1036 will be On; when CH3 (Y6, Y7) pulses have been sent, M1037 will be On. Next time when PLSR instruction is enabled, M1029, M1030, M1036 or M1037 will be 0 again and after the pulse output is completed, they will become 1 again.
14. During every acceleration section, the number of pulses (frequency × time) may not all be integers. PLC will round up the number to an integer before the output. Therefore, the acceleration time of every section may not be exactly the same. The offset is determined upon the frequency and the decimal after rounding up. In order to ensure the correct number of output pulses, PLC will supplement insufficient pulses in the last section.
15. For SA/EH series MPU, there is no limitation on the times of using this instruction in the program. However, for SA/SX/SC/EH series MPU, two instructions can be exeucted at the same time; for EH2/SV series MPU, four instructions can be executed at the same time.

Program Example:

1. When X0 = On, the pulses will output at the maximum frequency 1,000Hz with the total number D10 at 3,000ms from Y0. The frequency will increase by 1,000/10Hz at a time and every frequency will last for 3,000/9ms.
2. When X10 is Off, the output will be interrupted. When X0 is On again, the counting of pulses will start from 0.





Remarks:

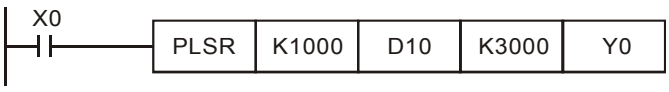
1. The outputs cannot be the same as those of API 57 PLSY and API 58 PWM.
2. When there are many high speed pulse output instructions (PLSY, PWM, PLSR) in a program for Y0 output, and provided these instructions are being executed in the same scan period, PLC will set up and output the instructions with the fewest steps.
3. With M1133 ~ M1135 and D1133, Y0 of SA/SX/SC series MPU can output pulses at up to 50KHz. See 2.11 for more details of special D and special M.

   Range of output frequecies for all series:

MPU	ES/EX/SS	SA/SX/SC	EH	EH2/SV
Range	Y0: 10 ~ 10,000Hz Y1: 10 ~ 10,000Hz	Y0: 10 ~ 30,000Hz Y1: 10 ~ 30,000Hz	Y0: 10 ~ 200,000Hz Y2: 10 ~ 200,000Hz	Y0: 10 ~ 200,000Hz Y2: 10 ~ 200,000Hz Y4: 10 ~ 200,000Hz Y6: 10 ~ 200,000Hz

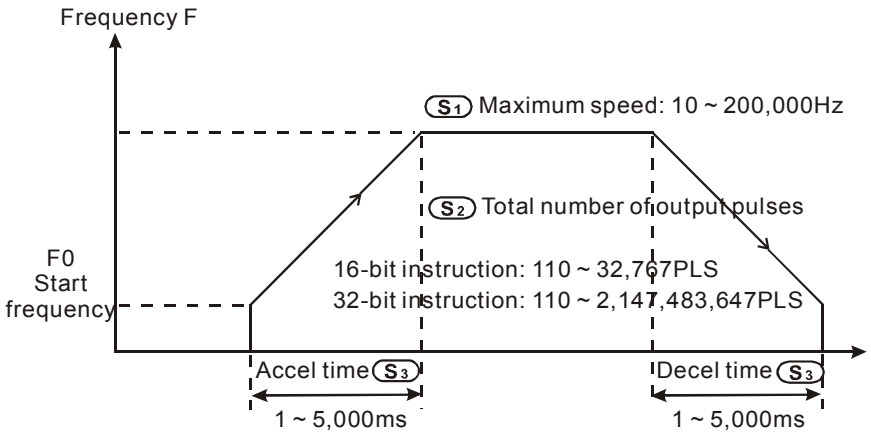
Functions in EH series MPU:

1. Relevant devices for EH/EH2/SV series MPU:

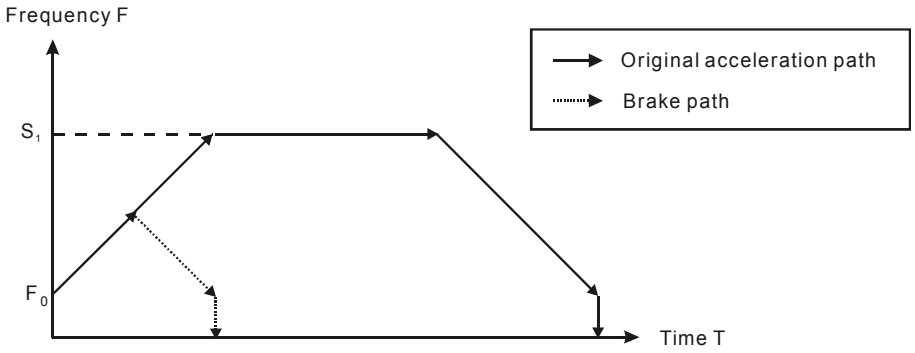


2. The range of pulse speed for this instruction is 10 ~ 200,000Hz. If the set values of maximum speed and acceleration/deceleration time exceed the range, PLC will operate by the default value that is within the range.

Operand		S_1	S_2	S_3	D
Explanation		Max. frequency	Total number of pulses	Accel/Decel time	Output point
Range	16-bit	10 ~ 32,767Hz	110 ~ 32,767	1 ~ 5,000ms	Y0 ~ Y7
	32-bit	10 ~ 200KHz	110 ~ 2,147,483,647		
Definition		K0: No output Kn: Designated frequency	Kn: Designated number	Flag: M1067, M1068	See settings of D1220, D1221



3. The acceleration/deceleration of EH/EH2/SV series MPU is based on the number of pulses. If the output cannot reach the maximum acceleration frequency within the acceleration/deceleration time offered, the instruction will automatically adjust the acceleration/deceleration time and the maximum frequency.
4. The operands have to be set before the execution of the instruction.
5. All acceleration/deceleration instructions are included with the brake function. The brake function will be enabled when PLC is performing acceleration and the switch contact is suddenly Off. The deceleration will operate at the slope of the acceleration.



API	Mnemonic	Operands	Function	Controllers
60	IST	(S) (D1) (D2)	Initial State	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*													IST: 7 steps
D_1				*												
D_2				*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start device in the designated operation mode **D_1**: The smallest No. of designated step in auto mode

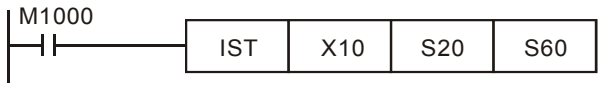
D_2: The biggest No. of designated step in auto mode

Explanations:

1. **S** will occupy 8 consecutive points.
2. Range of **D_1** and **D_2**: for SA/SX/SC/EH/EH2/SV S20 ~ S899; for ES/EX/SS S20 ~ S127; **D_2** > **D_1**.
3. See the specifications of each model for their range of use.
4. ES/SA series MPU does not support E, F index register modification.
5. IST instruction can only be used once in the program.
6. Flags: M1040 ~ M1047. See remarks for more details.
7. IST instruction is a handy instruction specifically for the initial status of step ladder control procedure to accommodate special auxiliary relay.

Program Example 1:

1. Use of IST instruction

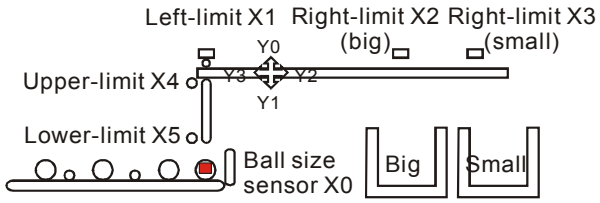


S X10: Individual operation
X11: Zero return
X12: Step operation
X13: One cycle operation
X14: Continuous operation
X15: Zero return enabled switch
X16: Start switch
X17: Stop switch

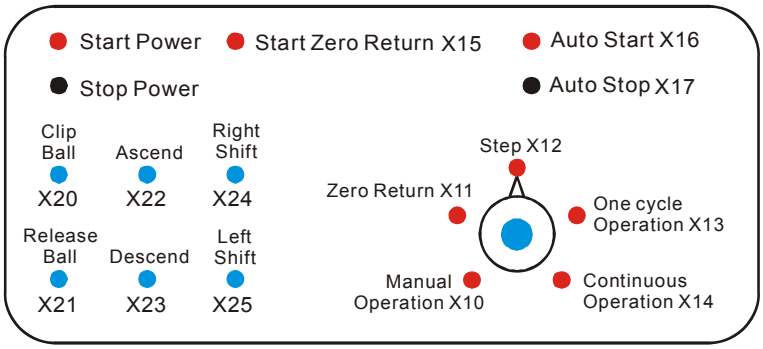
2. When IST instruction is being executed, the following special auxiliary relays will switch automatically.
   M1040: Operation forbidden
   M1041: Operation starts
   M1042: Pulse output enabled
   M1047: STL monitor enabled
   S0: Initiates manual operation
   S1: Initiates zero return
   S2: Initiates auto operation
3. S10 ~ S19 are for zero return and cannot be used as general steps. When S0 ~ S9 are in use, S0 ~ S2 represent manual operation mode, zero return mode and auto operation mode. Therefore, in the program , you have to write the circuit of the three steps in advance.
4. When switched to S1 (zero return) mode, any On in S10 ~ S19 will result in no zero return.
5. When switched to S2 (auto operation) mode, any On of the S in **D_1** ~ **D_2** or M1043 = On will result in no auto operation.

Program Example 2:

1. Robot arm control (by IST instruction):
   a) Motion request: Separate the big ball and small ball and move them to different boxes. Configure the control panel for the control.
   b) Motions of the robot arm: descending, clipping ball, ascending, right shifting, releasing ball, ascending, left shifting.
   c) I/O devices:

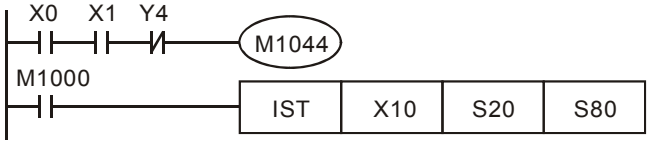


2. Operation modes:
   Manual operation: Turn On/Off of the load by a single button.
   Zero return: Press the zero return button to automatically zero-return the machine.
   Auto operation:
   a) Single step operation: Press “auto start” button for every one step forward.
   b) One cycle operation: Press “auto start” button at the zero point. After a cycle of auto operation, the operation will stops at the zero point. Press “auto stop” button in the middle of the operation to stop the operation and press “auto start” to restart the operation. The operation will resume until it meets the zero point.
   c) Continuous operation: Press “auto start” button at the zero point to resume the operation. Press “auto stop” to operate until it meets the zero point.
3. The control panel:

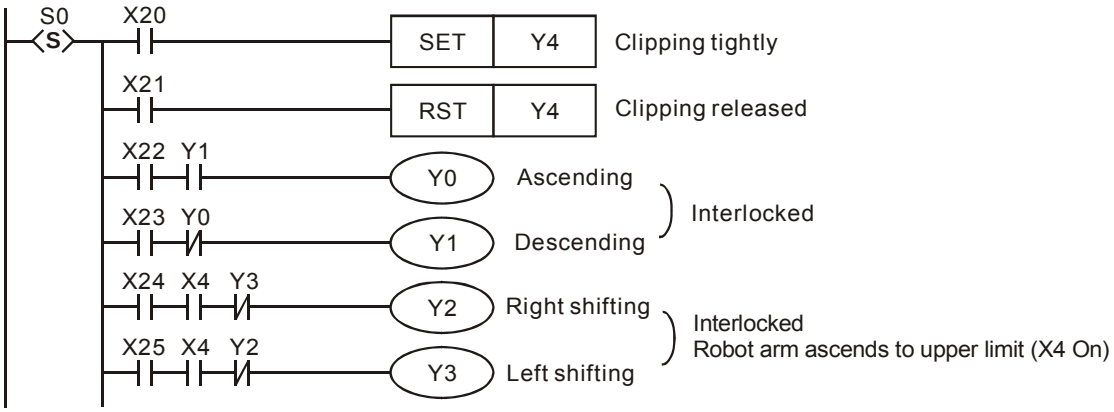


   a) Ball size sensor X0.
   b) Robot arm: left limit X1, big ball right limit X2, small ball right limit X3, upper limit X4, lower limit X5.
   c) Robot arm: ascending Y0, descending Y1, right shifting Y2, left shifting Y3, clipping Y4.

Start Circuit

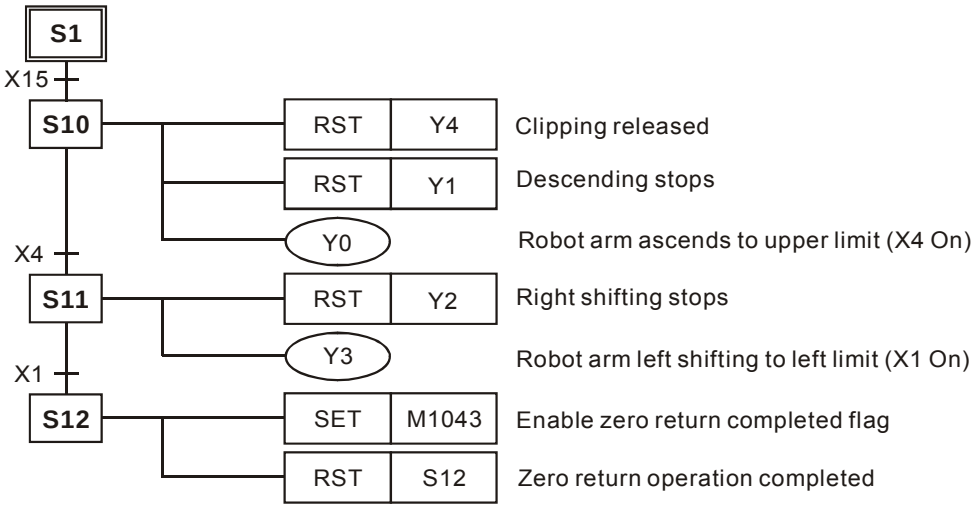


Manual Operation Mode

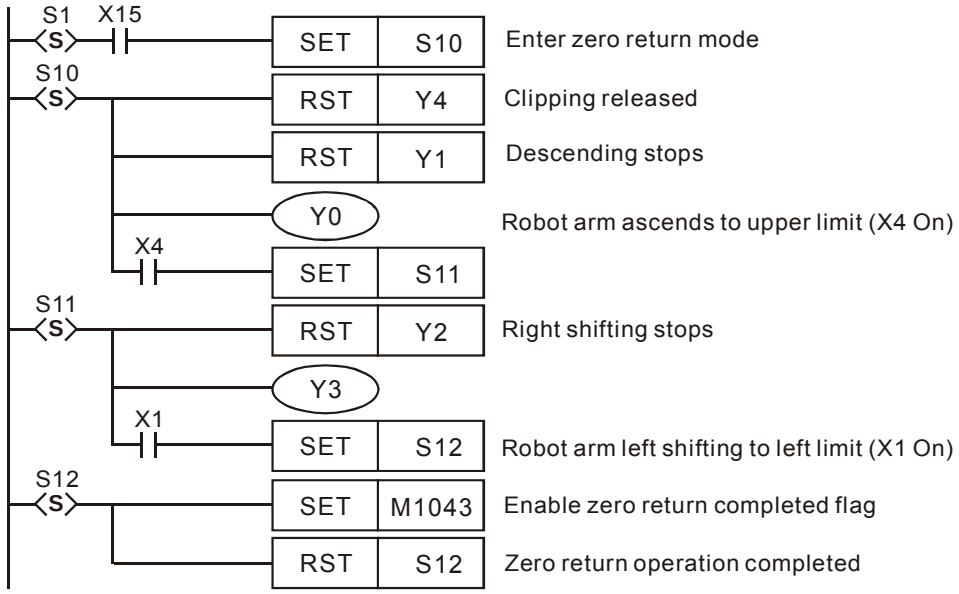


Zero Return Mode

SFC:

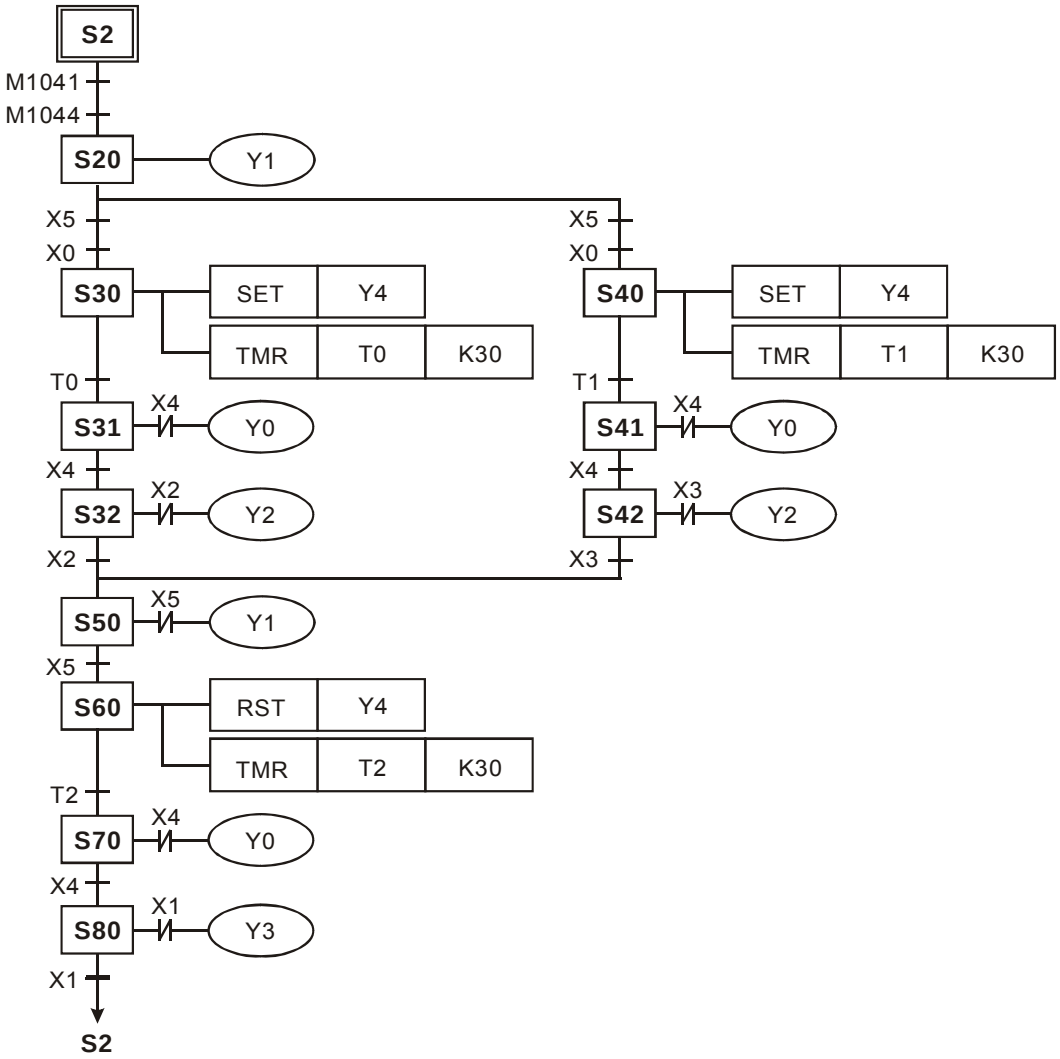


Ladder Diagram:

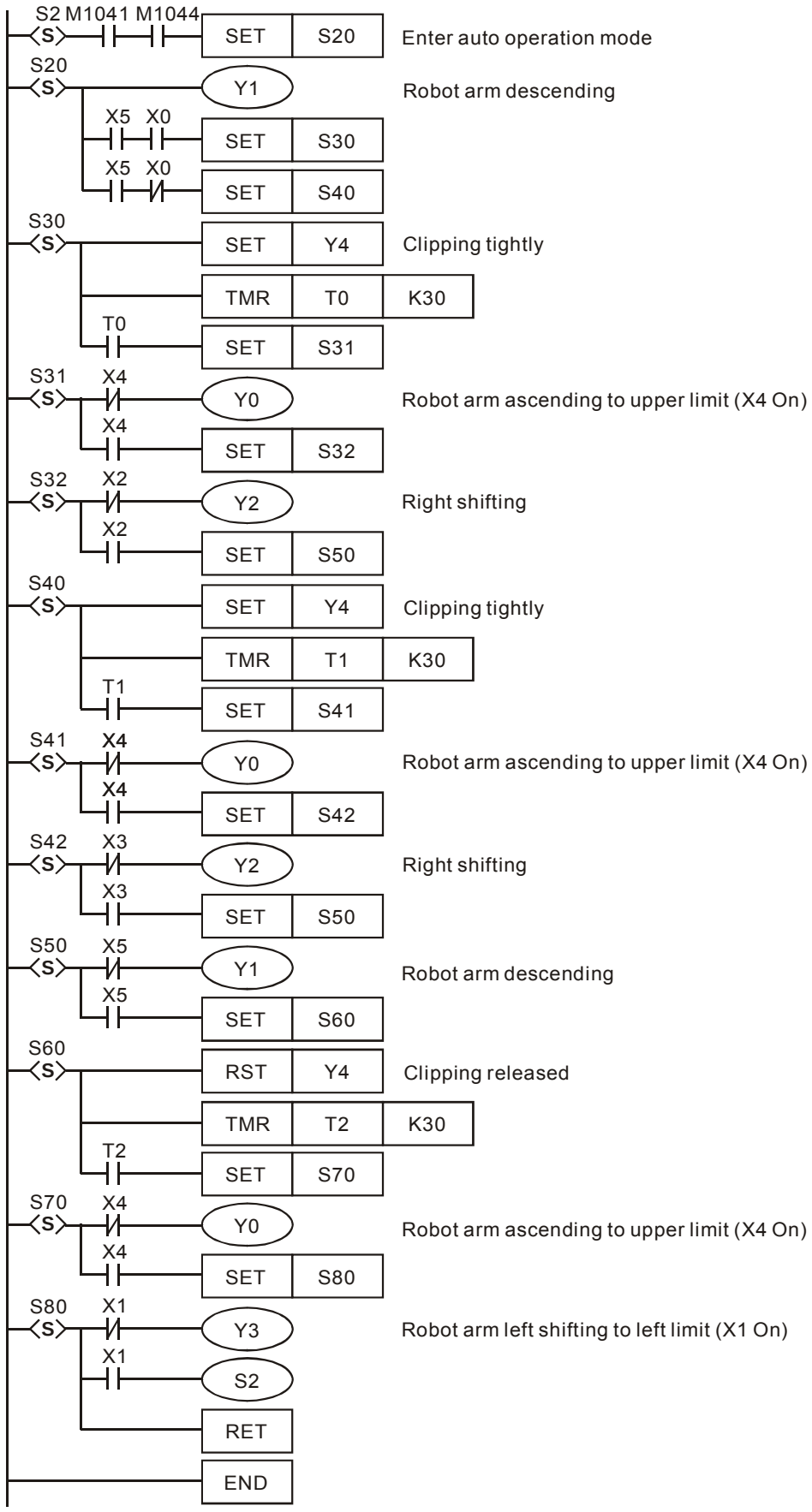


Auto Operation Modes

SFC:



Ladder Diagram:



Remarks:

Flag explanations:

M1040: When On, all step operations are forbidden.

1. Manual mode: M1040 keeps being On
2. Zero return/one cycle operation mode: Between the timing of pressing “auto stop” and “auto start” buttons, M1040 will keep being On.
3. Step mode: M1040 keeps being On until “auto start” button is pressed.
4. Continuous operation mode: When PLC goes from STOP to RUN, M1040 will keep being On and turn Off when “auto start” button is pressed.

M1041: Step operation starts. Special M for initial S2 to move to the next step.

1. Manual/zero return mode: M1041 keeps being Off.
2. Step/one cycle operation mode: M1041 will only be On when “auto start” button is pressed.
3. Continuous operation mode: M1041 keeps On when “auto start” button is pressed; Off when “auto stop” button is pressed.

M1042: Enabling pulse output. Sending pulses once when “auto start” button is pressed.

M1043: On when zero return is completed.

M1044: In continuous operation mode, M1044 has to be On to more S2 to the next step.

M1045: All output resets are forbidden.

If the machine (not at the zero point) goes

- from manual (S0) to zero return (S1)
- from auto (S2) to manual (S0)
- from auto (S2) to zero return (S1)

1. When M1045 is Off, and any of the S among $\mathbf{D_1}$ ~ $\mathbf{D_2}$ is On, SET Y output and the step in action will be reset to Off.
2. When M1045 is On, SET Y output will be remained but the step in action will be reset to Off

If the machine executes zero return (at the zero point) and goes from zero return (S1) to manual (S0), no matter M1045 is On or Off, SET Y output will be remained but the step in action will be reset to Off.

M1046: STL state setting. On when any of the steps is On. When M1047 is forced On, On of any S will result in On of M1046. D1040 ~ D1047 will record the No. of the previous 8 points before On of S.

M1047: On for enabling STL monitor. When IST instruction starts to be executed, M1047 will be forced On. In every scan time, as long as IST instruction is still On, M1047 will be forced On. M1047 monitors all the S.

D1040 ~ D1047: On status of step No. 1 ~ 8

API	Mnemonic			Operands	Function	Controllers		
61	D	SER	P	(S1) (S2) (D) (n)	Search a Data Stack	ES/EX/SS	SA/SX/SC	EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1							*	*	*	*	*	*	*			SER, SERP: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	DSER, DSERP: 17 steps
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

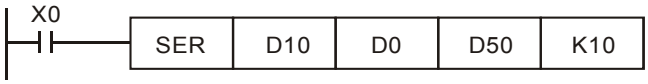
S_1: Start device for data stack comparison **S_2**: Data to be compared **D**: Start device for storing comparison result **n**: Length of data to be compared

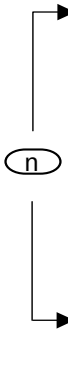
Explanations:

1. When **S_2** are used in device F, only 16-bit instruction is applicable.
2. **D** will occupy 5 consecutive points.
3. Range of **n**: for 16-bit instruction 1 ~ 256; for 32-bit instruction 1 ~ 128.
4. See the specifications of each model for their range of use.
5. The **n** data in the registers starting from **S_1** are compared with **S_2** and the results are stored in the registers starting from **D**.
6. In the 32-bit instruction, **S_1**, **S_2**, **D** and **n** will designate 32-bit registers.
7. For **D**, the 16-bit counters and 32-bit counters in SA/SX/SC series MPU cannot be mixed when being used.

Program Example:

1. When X0 = On, the data stack consist of D10 ~ D19 will be compared against D0 and the result will be stored in D50 ~ D52. If there are equivalent values appearing during the comparison, D50 ~ D52 will all be 0.
2. The data are compared algebraically. (-10 < 2).
3. The No. of the register with the smallest value among the compared data will be recorded in D53; the biggest will be recorded in D54. When there are more than one smallest value or biggest value, device D will record the No. of the register with bigger value.





S_1	Content	Data to be compared	Data No.	Result
D10	88	S_2	0	
D11	100		1	Equal
D12	110		2	
D13	150		3	
D14	100	D0 = K100	4	Equal
D15	300		5	
D16	100		6	Equal
D17	5		7	Smallest
D18	100		8	Equal
D19	500		9	Biggest

D	Content	Description
D50	4	Total number of data with equivalent values
D51	1	No. of the first equivalent value
D52	8	No. of the last equivalent value
D53	7	No. of the smallest value
D54	9	No. of the biggest value

API	Mnemonic		Operands	Function	Controllers
62	D	ABSD	(S1) (S2) (D) (n)	Absolute Drum Sequencer	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1							*	*	*	*	*	*	*			ABSD: 9 steps DABSD: 17 steps
S_2											*	*	*			
D		*	*	*												
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Start device in the data table **S_2**: No. of counter **D**: Start No. of the devices for the comparison results

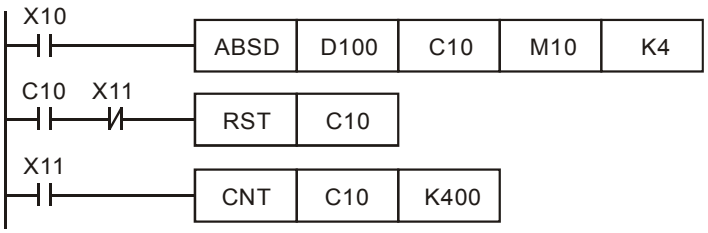
n: Number of data for comparison

Explanations:

1. When **S_1** designates KnX, KnY, KnM and KnS, the 16-bit instruction has to designate K4 and 32-bit instruction has to designate K8.
2. Range of **n**: 1 ~ 64
3. See the specifications of each model for their range of use.
4. ABSD instruction is for the absolute control of the multiple output pulses generated by the present value in the counter.
5. **S_2** of DABSD instruction can designate high speed counters. However, when the present value in the high speed counter is compared with the target value, the result cannot output immediately owing to the scan time. If an immediate output is required, please use DHSZ instruction that is exclusively for high speed counters.

Program Example:

1. Before the execution of ABSD instruction, use MOV instruction to write all the set values into D100 ~ D107 in advance. The even-number D is for lower bound value and the odd-number D is for upper bound value.
2. When X10 = On, the present value in counter C10 will be compared with the four groups of lower and upper bound values in D100 ~ D107. The comprison results will be stored in M10 ~ M13.
3. When X10 = Off, the original On/Off status of M10 ~ M13 will be remained.



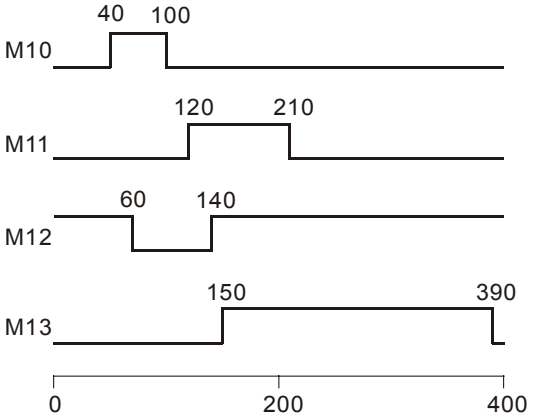
4. M10~ M13 will be On when the present value in C10 ≦ upper bound value or ≧ lower bound value.

Lower bound value	Upper bound value	Present value in C10	Output
D100 = 40	D101 = 100	40 ≦ C10 ≦ 100	M10 = On
D102 = 120	D103 = 210	120 ≦ C10 ≦ 210	M11 = On

Lower bound value	Upper bound value	Present value in C10	Output
D104 = 140	D105 = 170	140 ≦ C10 ≦ 170	M12 = On
D106 = 150	D107 = 390	150 ≦ C10 ≦ 390	M13 = On

5. If the lower bound value > upper bound value, when C10 < upper bound value (60) or > upper bound value (140), M12 will be On.

Lower bound value	Upper bound value	Present value in C10	Output
D100 = 40	D101 = 100	40 ≦ C10 ≦ 100	M10 = On
D102 = 120	D103 = 210	120 ≦ C10 ≦ 210	M11 = On
D104 = 140	D105 = 60	60 ≦ C10 ≦ 140	M12 = On
D106 = 150	D107 = 390	150 ≦ C10 ≦ 390	M13 = On



API	Mnemonic	Operands	Function	Controllers
63	INCD	S_1 S_2 D n	Incremental Drum Sequencer	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	INCD: 9 steps
S_1							*	*	*	*	*	*	*			
S_2												*				
D		*	*	*												
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Start device in the data table **S_2**: No. of counter **D**: Start No. of the devices for the comparison results

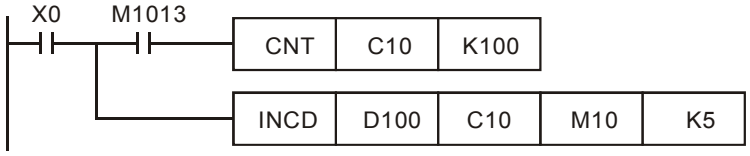
n: Number of data for comparison

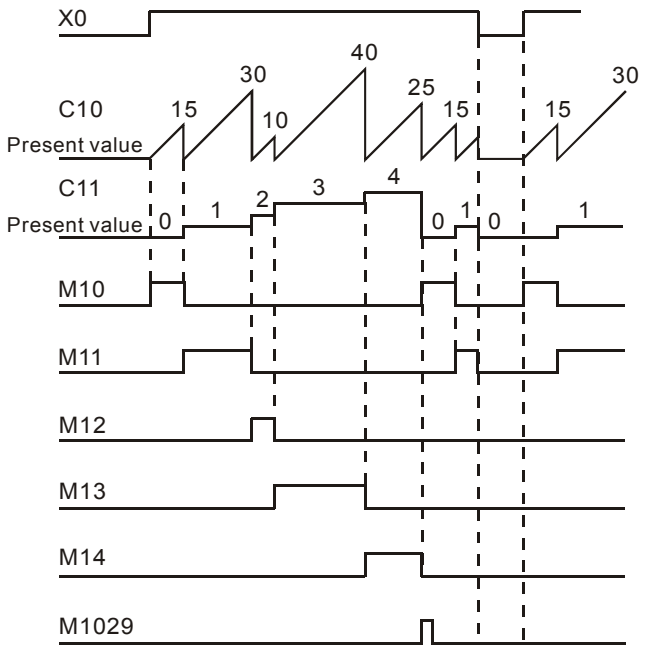
Explanations:

1. When **S_1** designates KnX, KnY, KnM and KnS, it has to designate K4.
2. In the 16-bit instruction, **S_2** has to designate C0 ~ C198 and will occupy 2 consecutive No. of counters.
3. Range of **n**: 1 ~ 64
4. See the specifications of each model for their range of use.
5. Flag: M1029 (instruciton execution completed)
6. INCD instruction is for the relative control of the multiple output pulses generated by the present value in the counter.
7. The present value in **S_2** is compared with **S_1**. **S_2** will be reset to 0 whenever a comparison is completed. The current number of data processed in temporarily stored in **S_2** + 1.
8. When n data have been processed, M1029 will be On for one scan period.

Program Example:

1. Before the execution of INCD instruction, use MOV instruction to write all the set values into D100 ~ D104 in advance. D100 = 15, D101 = 30, D102 = 10, D103 = 40, D104 = 25.
2. The present value in C10 is compared against the set values in D100 ~ D104. The present value will be reset to 0 whenever a comparison is completed.
3. The current number of data having been processed is temporarily stored in C11.
4. The number of times of reset is temporarily stored in C11.
5. Whenever the content in C11 pluses 1, M10 ~ M14 will also correspondingly change. See the timing diagram below.
6. After the 5 groups of data have been compared, M1029 will be On for one scan period.
7. When X0 goes from On to Off, C10 and C11 will both be reset to 0 and M10 ~ M14 will all be Off. When X0 is On again, the instruction will start its execution again from the beginning.





API	Mnemonic	Operands	Function	Controllers
64	TTMR	D n	Teaching Timer	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D													*			TTMR: 5 steps
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Device No. for storing the "On" time of button switch **n**: Multiple setting

Explanations:

1. **D** will occupy 2 consecutive devices.
2. Range of **n**: 0 ~ 2
3. See the specifications of each model for their range of use.
4. For SA series MPU, TTMR instruction can be used 8 times in the program.
5. The "On" time (unit: 100ms) of the external button switch is stored in device No. **D** + 1. The "On" time (unit: second) of the switch is multiplied by **n** and stored in **D**.
6. Multiple setting:

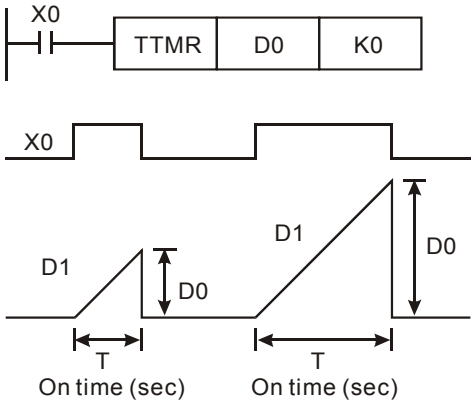
   When n = 0, unit of **D** = second

   When n = 1, unit of **D** = 100ms (D × 10)

   When n = 2, unit of **D** = 10ms (D × 100)

Program Example 1:

1. The "On" (being pressed) time of button switch X0 is stored in D1. The setting of n is stored in D0. Therefore, the button switch will be able to adjust the set value in the timer.
2. When X0 goes Off, the content in D1 will be cleared to 0, but the content in D0 will remain.

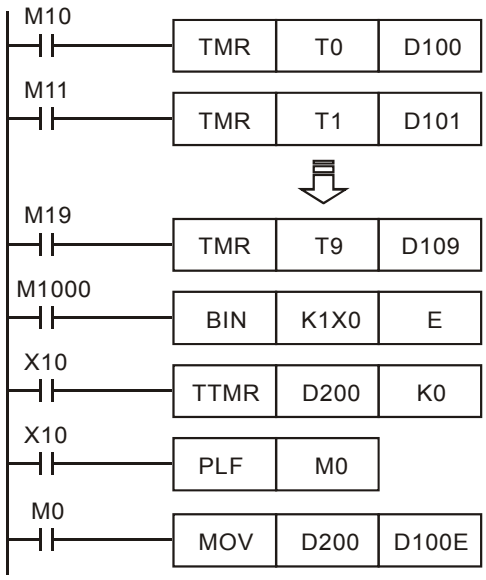


3. Assume the "On" time of X0 is T (sec.), see the relation between D0, D1 and n in the table below.

n	D0	D1 (unit: 100ms)
K0 (unit: s)	1 × T	D1 = D0 × 10
K1 (unit: 100 ms)	10 × T	D1 = D0
K2 (unit: 10 ms)	100 × T	D1 = D0/10

Program Example 2:

1. Use TMR instruction to write in 10 groups of set time.
2. Write the set values into D100 ~ D109 in advance.
3. The timing unit for timer T0 ~ T9 is 0.1 sec. The timing unit for the teaching timer is 1 sec.
4. Connect the 1-bit DIP switch to X0 ~ X3 and use BIN instruction to convert the set value of the switch into a bin value and store it in E.
5. Store the “On” time (sec.) of X10 in D200.
6. M0 refers to the pulses generated from one scan period after the button switch of the teaching timer X10 is released.
7. Use the set number of the DIP switch as the indirectly designated pointer and send the content in D200 to D100E (D100 ~ D109).



Remarks:

1. For SA series MPU, TTMR instruction can be used 8 times in the program. But in a subroutine or interruption subroutine, the instruction can only be used once.
2. For EH series MPU, there is no limitation on the times using this instruction in the program and 8 instructions can be executed at the same time.

API	Mnemonic	Operands	Function	Controllers
65	STMR	(S) (m) (D)	Special Timer	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S											*					STMR: 7 steps
m					*	*										
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

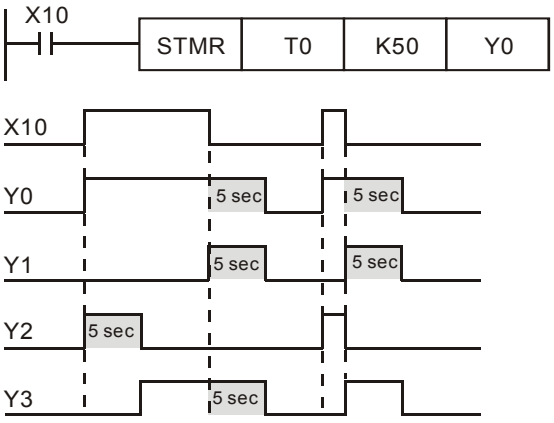
S: No. of timer **m**: Set value in timer (unit: 100ms) **D**: No. of start output device

Explanations:

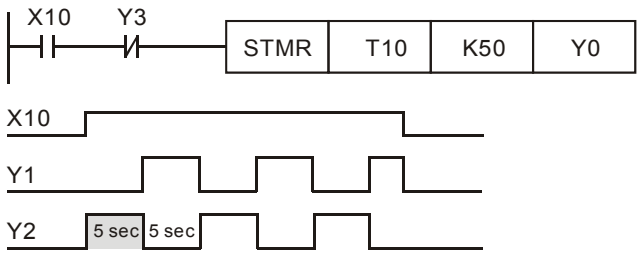
1. Range of **S**: for SA/SX/SC T0 ~ T191; for EH/EH2/SV T0 ~ T199
2. Range of **m**: 1 ~ 32,767
3. **D** will occupy 4 consecutive devices.
4. See the specifications of each model for their range of use.
5. STMR instruction is used for Off-delay, one shot timer and flashing sequence.
6. The No. of timers designated by STMR instructions can be used only once.

Program Example:

1. When X10 = On, STMR instruction will designate timer T0 and set the set value in T0 as 5 seconds.
2. Y0 is the contact of Off-delay. When X10 goes from Off to On, Y0 will be On. When X10 goes from On to Off, Y0 will be Off after a five seconds of delay.
3. When X10 goes from On to Off, there will be a five seconds of Y1 = On output.
4. When X10 goes from Off to On, there will be a five seconds of Y2 = On output.
5. When X10 goes from Off to On, Y3 will be On after a five seconds of delay. When X10 goes from On to Off, Y3 will be Off after a five seconds of delay.



6. Add a b contact of Y3 after X10, and Y1 and Y2 can operate for flashing sequence output. When X10 goes Off, Y0, Y1 and Y3 will be Off and the content in T10 will be reset to 0.



API	Mnemonic			Operands	Function	Controllers	
66		ALT	P	(D)	Alternate State	ES/EX/SS	SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D		*	*	*												ALT, ALTP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

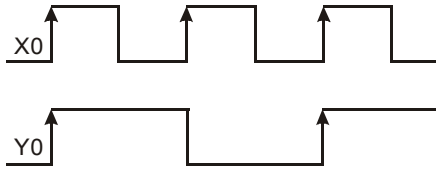
D: Destination device

Explanations:

1. See the specifications of each model for their range of use.
2. When ALT instruction is executed, “On” and “Off” of **D** will switch.
3. This instruction adopts pulse execution instructions (ATLP).

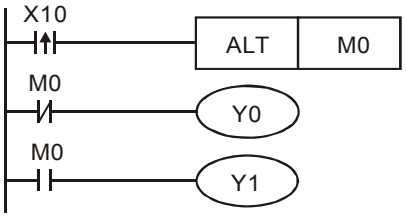
Program Example 1:

When X0 goes from Off to On, Y0 will be On. When X0 goes from Off to On for the second time, Y0 will be Off.



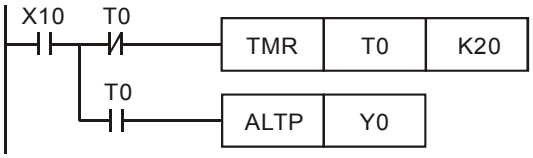
Program Example 2:

Using a single switch to enable and disable control. At the beginning, M0 = Off, so Y0 = On and Y1 = Off. When X10 switches between On/Off for the first time, M0 will be On, so Y1 = On and Y0 = Off. For the second time of On/Off switching, M0 will be Off, so Y0 = On and Y1 = Off.



Program Example 3:

Generating flashing. When X10 = On, T0 will generate a pulse every 2 seconds and Y0 output will switch between On and Off following the T0 pulses.



API	Mnemonic	Operands	Function	Controllers
67	RAMP	S1 S2 D n	Ramp Variable Value	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1													*			RAMP: 9 steps
S_2													*			
D													*			
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

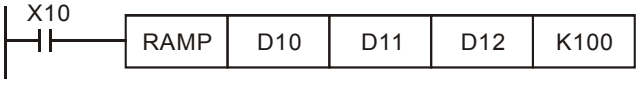
S_1: Start of ramp signal **S_2**: End of ramp signal **D**: Duration of ramp signal **n**: Scan times

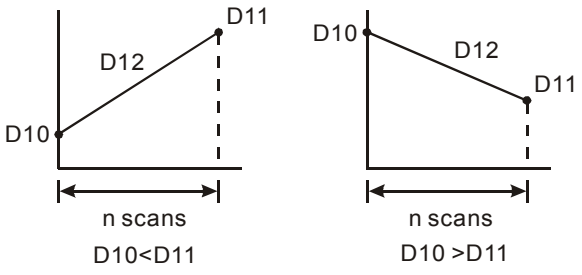
Explanations:

1. Range of **n**: 1 ~ 32,767
2. **D** will occupy 2 consecutive points.
3. See the specifications of each model for their range of use.
4. Flags: M1026 (enabling RAMP; see remarks for more details); M1029 (RAMP execution completed).
5. This instruction is for obtaining slope (the relation between linearity and scan time). Before using this instruction, you have to preset the scan time.
6. The set value of start ramp signal is pre-written in D10 and set value of end ramp signal in D11. When X10 = On, D10 increases towards D11 through n (= 100) scans (the duration is stored in D12). The times of scans are stored in D13.
7. In the program, first drive M1039 = On to fix the scan time. Use MOV instruction to write the fixed scan time to the special data register D1039. Assume the scan time is 30ms and take the above program for example, n = J100, the time for D10 to increase to D11 will be 3 seconds (30ms × 100).
8. When X10 goes Off, the instruction will stop its execution. When X10 goes On again, the content in D12 will be reset to 0 for recalculation.
9. When M1026 = Off, M1029 will be On and the content in D12 will be reset to the set value in D10.

Program Example:

When this instruction is used with analog signal outputs, it will be able to buffer START and STOP.

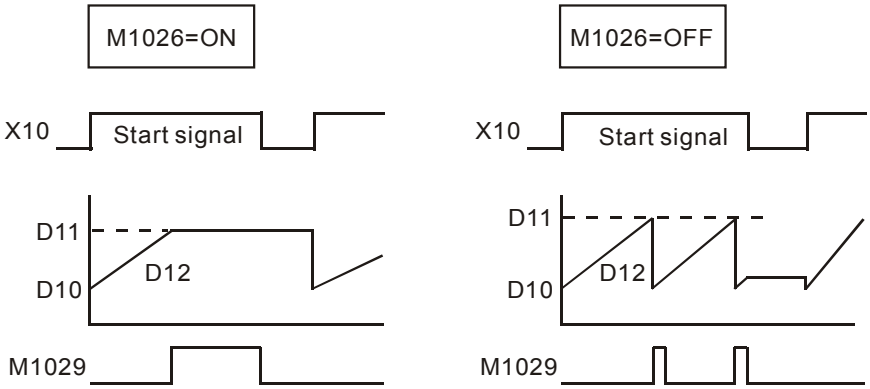




The scan times is stored in D13

Remarks:

D12 for enabling On/Off of M1026:



API	Mnemonic	Operands	Function	Controllers
69	SORT	(S) (m1) (m2) (D) (n)	Sort Tabulated Data	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			SORT: 11 steps
m_1					*	*										
m_2					*	*										
D													*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start device for the original data **m_1**: Groups of data to be sorted **m_2**: Number of columns of data

D: Start device for the sorted data **n**: Reference value for data sorting

Explanations:

1. Range of **m_1**: 1 ~ 32.
2. Range of **m_2**: 1 ~ 6
3. Range of **n**: 1 ~ m_2
4. See the specifications of each model for their range of use.
5. Flag: M1029 (SORT execution completed).
6. The sorted result is stored in m1 × m2 registers starting from the device designated in **D**. Therefore, if **S** and **D** designate the same register, the sorted result will be the same as the data designated in **S**.
7. It is better that the start No. designated in **S** is 0.
8. The sorting will be completed after **m_1** times of scans. After the sorting is completed, M1029 will be On.
9. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example:

1. When X0 = On, the sorting will start. When the sorting is completed, M1029 will be On. DO NOT change the data to be sorted during the execution of the instruction. If you wish to change the data, please make X0 go from Off to On again.

2. Example table of data sorting

Columns of data: m_2

Groups of data: m_1	Data Column 1	2	3	4	5
Row \ Column	Students No.	Physics	English	Math	Chemistry
1	(D0) 1	(D5) 90	(D10) 75	(D15) 66	(D20) 79
2	(D1) 2	(D6) 55	(D11) 65	(D16) 54	(D21) 63
3	(D2) 3	(D7) 80	(D12) 98	(D17) 89	(D22) 90
4	(D3) 4	(D8) 70	(D13) 60	(D18) 99	(D23) 50
5	(D4) 5	(D9) 95	(D14) 79	(D19) 75	(D24) 69

Sorted data when D100 = K3.

Columns of data: m_2

Groups of data: m_1	Data Column 1	2	3	4	5
Row \ Column	Students No.	Physics	English	Math	Chemistry
1	(D50) 4	(D55) 70	(D60) 60	(D65) 99	(D70) 50
2	(D51) 2	(D56) 55	(D61) 65	(D66) 54	(D71) 63
3	(D52) 1	(D57) 90	(D62) 75	(D67) 66	(D72) 79
4	(D53) 5	(D58) 95	(D63) 79	(D68) 75	(D73) 69
5	(D54) 3	(D59) 80	(D64) 98	(D69) 89	(D74) 90

Sorted data when D100 = K5.

Columns of data: m_2

Groups of data: m_1	Data Column 1	2	3	4	5
Row \ Column	Students No.	Physics	English	Math	Chemistry
1	(D50) 4	(D55) 70	(D60) 60	(D65) 99	(D70) 50
2	(D51) 2	(D56) 55	(D61) 65	(D66) 54	(D71) 63
3	(D52) 5	(D57) 95	(D62) 79	(D67) 75	(D72) 69
4	(D53) 1	(D58) 90	(D63) 75	(D68) 66	(D73) 79
5	(D54) 3	(D59) 80	(D64) 98	(D69) 89	(D74) 90

API	Mnemonic	Operands	Function	Controllers
70	D TKY	S D_1 D_2	Ten Key Input	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TKY: 7 steps DTKY: 13 steps
S	*	*	*	*												
D_1								*	*	*	*	*	*	*	*	
D_2		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

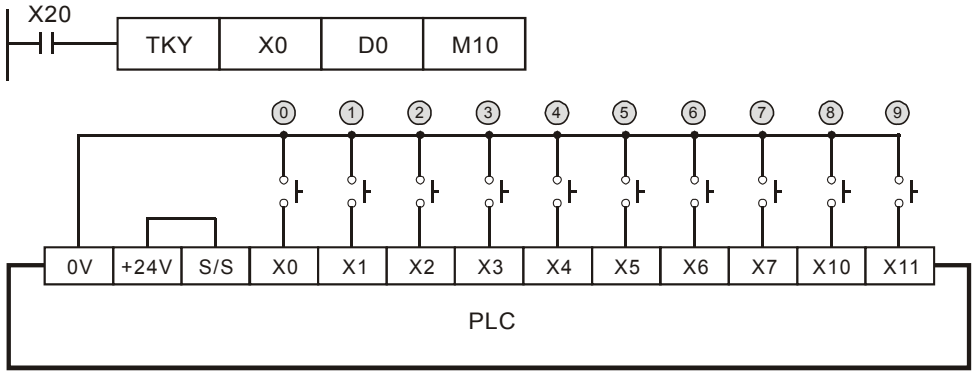
S: Start device for key input **D_1**: Device for storing keyed-in value **D_2**: Key output signal

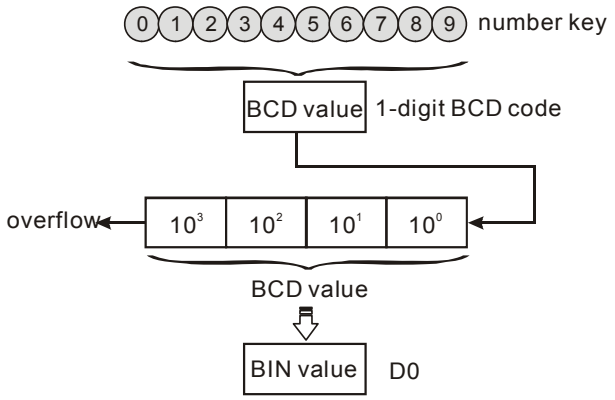
Explanations:

1. **S** will occupy 10 consecutive points; **D_2** will occupy 11 consecutive points.
2. See the specifications of each model for their range of use.
3. For SA series MPU, **S** and **D_2** do not support E, F index register modification.
4. This instruction designates 10 external input points (representing decimal numbers 0 ~ 9) starting from **S**. The 10 points are respectively connected to 10 keys. By pressing the keys, you can enter a 4-digit decimal figure 0 ~ 9,999 (16-bit instruction) or a 8-digit figure 0 ~ 99,999,999 (32-bit instruction) and store the figure in **D_1**. **D_2** is used for storing key status.
5. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

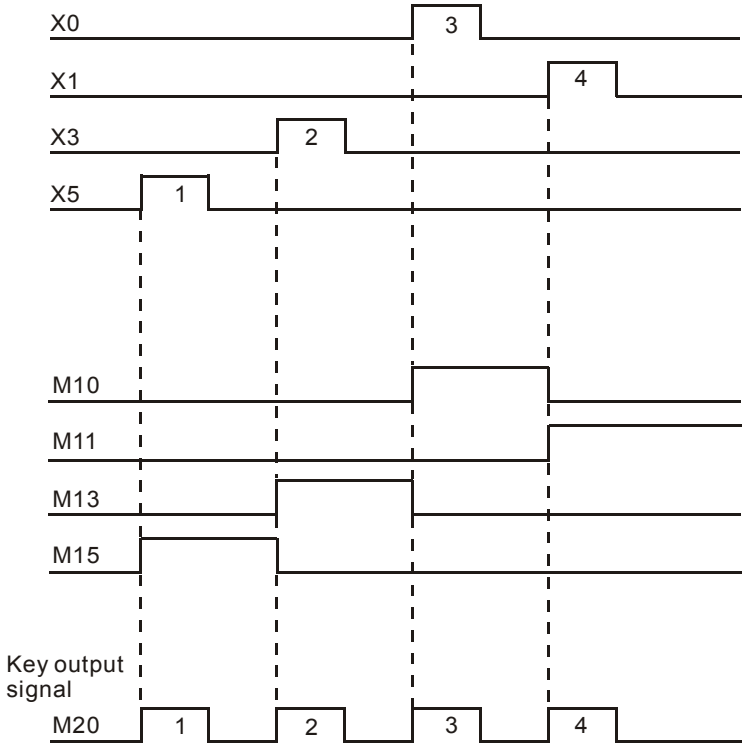
Program Example:

1. Connect the 10 input points starting from X0 to the 10 keys (0 ~ 9). When X20 = On, the instruction will be executed and the keyed-in values will be stored in D0 in bin form. The key status will be stored in M10 ~ M19.





2. As shown in the timing chart below, the 4 points X5, X3, X0, and X1 connected to the keys are entered in order and you can obtain the result 5,301. Store the result in D0. 9,999 is the maximum value allowed to stored in D0. Once the value exceeds 4 digits, the highest digit will overflow.
3. M12 = On when from X2 is pressed to the other key is pressed. Same to other keys.
4. When any of the keys in X0 ~ X11 is pressed, one of M10 ~ 19 will be On correspondingly.
5. M20 = On when any of the keys is pressed.
6. When X20 goes Off, the keyed-in value prior to D0 will remain unchanged, but M10 ~ M20 will all be Off.



API	Mnemonic	Operands	Function	Controllers
71	D HKY	(S) (D1) (D2) (D3)	Hexadecimal Key Input	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*															HKY: 9 steps
D_1		*														DHKY: 17 steps
D_2											*	*	*	*	*	
D_3		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

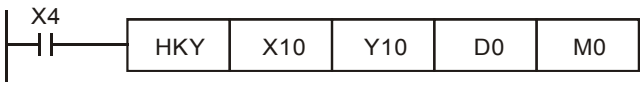
S: Start device for key scan input **D_1**: Start device for key scan output **D_2**: Device for storing keyed-in value **D_3**: Key output signal

Explanations:

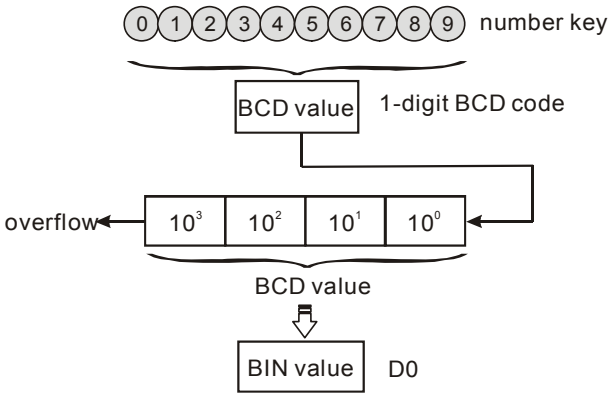
1. **S** will occupy 4 consecutive points.
2. **D_1** will occupy 4 consecutive points.
3. **D_3** will occupy 8 consecutive points.
4. See the specifications of each model for their range of use.
5. For SA series MPU, **S**, **D_1** and **D_3** do not support E, F index register modification.
6. Flags: M1029 (On whenever a matrix scan period is completed); M1167 (HKY input modes switch). See remarks for more details.
7. This instruction designates 4 continuous external input points starting from **S** and 4 continuous external input points starting from **D_1** to construct a 16-key keyboard by a matrix scan. The keyed-in value will be stored in **D_2** and **D_3** is used for storing key status. If several keys are pressed at the same time, the first key pressed has the priority.
8. The keyed-in value is termporarily stored in D0. When the 16-bit instruction HKY is in use, 9,999 is the maximum value D0 is able to store. When the value exceeds 4 digits, the highest digit will overflow. When the 32-bit instruction DHKY is in use, 99,999,999 is the maximum value D0 is able to store. When the value exceeds 8 digits, the highest digit will overflow.
9. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example:

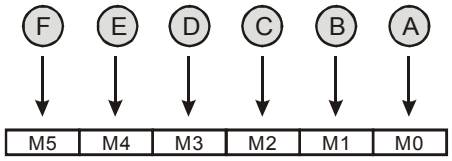
1. Designate 4 input points X10 ~ X13 and the other 4 input points Y10 ~ Y13 to construct a 16-key keyboard. When X4 = On, the instruction will be executed and the keyed-in value will be stored in D0 in bin form. The key status will be stored in M0 ~ M7.



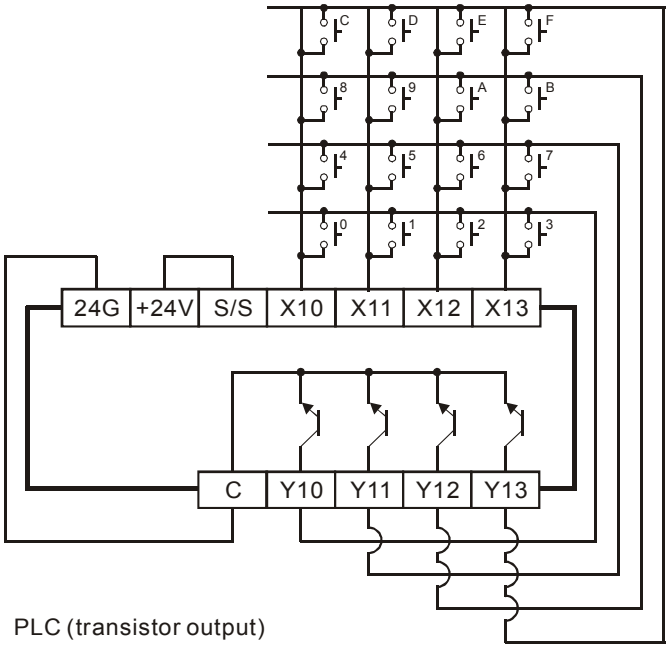
2. Key in numbers:



3. Function keys input:
   a) When A is pressed, M0 will be On and retained. When D is pressed next, M0 will be Off, M3 will be On and retained.
   b) When many keys are pressed at the same time, the first key pressed has the priority.



4. Key output signal:
   a) When any of A ~ F is pressed, M6 will be On for once.
   b) When any of 0 ~ 9 is pressed, M7 will be On for once.
5. When X4 goes Off, the keyed-in value prior to D0 will remain unchanged, but M0 ~ M7 will all be Off.
6. External wiring:



Remarks:

1. When this instruction is being executed, it will require 8 scans to obtain one valid keyed-in value. A scan period that is too long or too short may result in poor keyed-in effect, which can be avoided by the following methods:
   a) If the scan period is too short, I/O may not be able to respond in time, resulting in not being able to read the keyed-in value correctly. In this case, please fix the scan time.
   b) If the scan period is too long, the key may respond slowly. In this case, write this instruction into the time interruption subroutine to fix the time for the execution of this instruction.
2. Functions of M1167:
   a) When M1167 = On, HKY instruction will be able to input the hexadecimal value of 0 ~ F.
   b) When M1167 = Off, HKY instruction will see A ~ F as function keys.
3. Functions of D1037 (only supports EH/EH2/SV series MPU):
   Write D1037 to set the overlapping time for keys (unit: ms). The overlapping time will vary upon different program scan time and the settings in D1037.

API	Mnemonic	Operands	Function	Controllers
72	DSW	S D1 D2 n	Digital Switch	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*															DSW: 9 steps
D_1		*														
D_2											*	*	*			
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

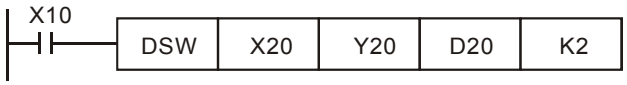
S: Start device for switch scan input **D_1**: Start device for switch scan output **D_2**: Device for storing the set value of switch **n**: Groups of switches

Explanations:

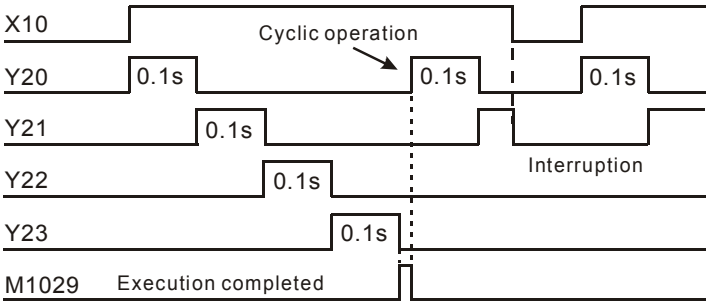
1. Range of **n**: 1 ~ 2
2. See the specifications of each model for their range of use.
3. Flag: M1029 (DSW execution completed)
4. This instruction designates 4 or 8 consecutive external input points starting from **S** and 4 consecutive external input points starting from **D_1** to scan read 1 or 2 4-digit DIP switches.The set values of DIP switches are stored in **D_2**. **n** decides to read 1 or 2 4-digit DIP switches.
5. There is no limitation on the times of using this instruction in the program. However, for SA series MPU, only one instruction can be executed at a time. For EH series MPU, two instructions are allowed to be executed at a time.

Program Example:

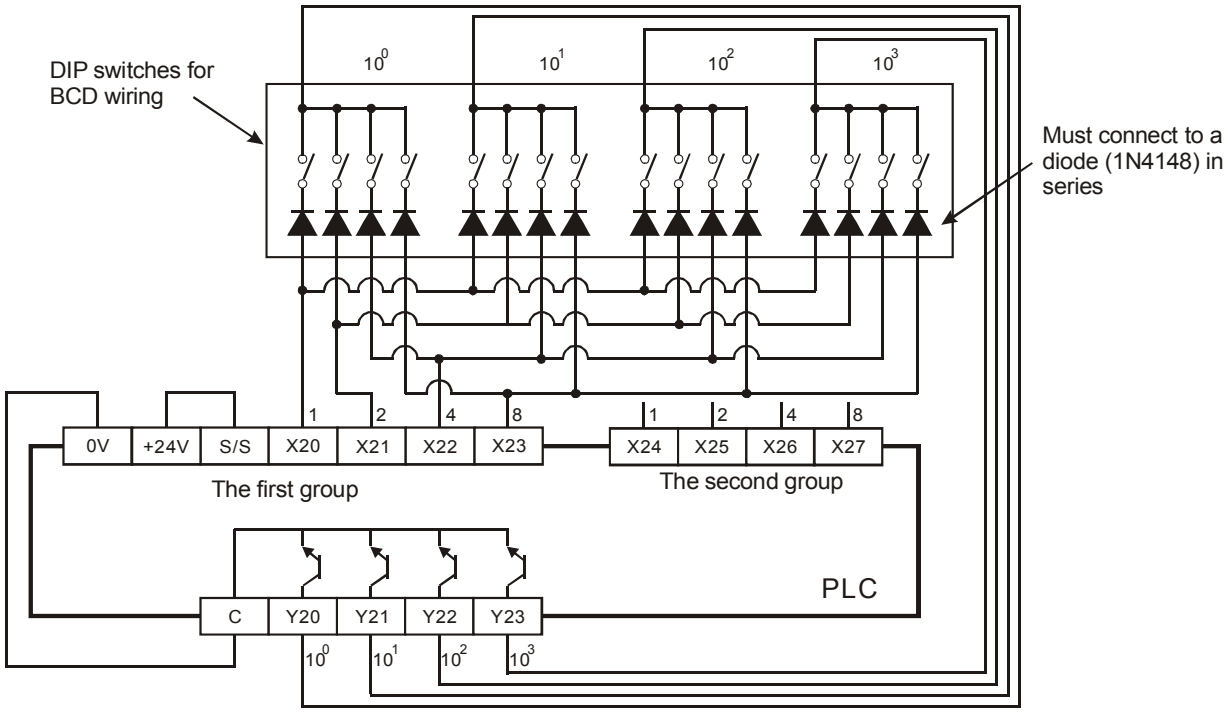
1. The first group of DIP switches consist of X20 ~ X23 and Y20 ~ Y23. The second group of switches consist of X24 ~ X27 and Y20 ~ Y23. When X10 = On, the instruction will be executed and the set values of the first group switches will be read and converted into bin values before being stored in D20. The set values of the second group switches will be read, converted into bin values and stored in D21.



2. When X10 = On, the Y20 ~ Y23 auto scan cycle will be On. Whenever a scan cycle is completed, M1029 will be On for a scan period.
3. Please use transistor output for Y20 ~ Y23. Every pin 1, 2, 4, 8 shall be connected to a diode (0.1A/50V) before connecting to the input terminals on PLC.

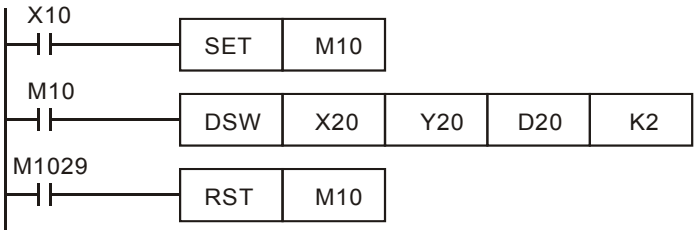


4. Wiring for DIP swich input:



Remarks:

1. When **n** = K1, $\mathbf{D_2}$ will occupy one register. When **n** = K2, $\mathbf{D_2}$ will occupy 2 consecutive registers.
2. Follow the methods below for the transistor scan output:
   a) When X10 = On, DSW instruction will be executed. When X10 goes Off, M10 will keep being On until the scan output completes a scan cycle and go Off.
   b) When X10 is used as a button switch, whenever X10 is pressed once, M10 will be reset to Off when the scan output designated by DSW instruction completes a scan cycle. The DIP switch data will be read completely and the scan output will only operate during the time when the button switch is pressed. Therefore, even the scan output is a transistor type, the life span of the transistor can be extended because it does not operate too frequently.



API	Mnemonic	Operands	Function	Controllers
73	SEGD P	(S) (D)	Seven Segment Decoder	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	SEGD, SEGDP: 5 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

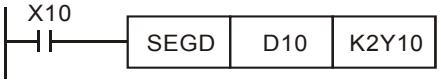
S: Source device to be decoded **D**: Output device after the decoding

Explanations:

See the specifications of each model for their range of use.

Program Example:

When X10 = On, the contents (0 ~ F in hex) of the lower 4 bits (b0 ~ b3) of D10 will be decoded into a 7-segment display for output. The decoded results will be stored in Y10 ~ Y17. If the content exceeds 4 bits, the lower 4 bits are still used for the decoding.



Decoding table of the 7-segment display:

Hex	Bit combination	Composition of the 7-segment display	Status of each segment							Data displayed
			B0(a)	B1(b)	B2(c)	B3(d)	B4(e)	B5(f)	B6(g)	
0	0000		ON	ON	ON	ON	ON	ON	OFF	0
1	0001		OFF	ON	ON	OFF	OFF	OFF	OFF	1
2	0010		ON	ON	OFF	ON	ON	OFF	ON	2
3	0011		ON	ON	ON	ON	OFF	OFF	ON	3
4	0100		OFF	ON	ON	OFF	OFF	ON	ON	4
5	0101		ON	OFF	ON	ON	OFF	ON	ON	5
6	0110		ON	OFF	ON	ON	ON	ON	ON	6
7	0111		ON	ON	ON	OFF	OFF	ON	OFF	7
8	1000		ON	ON	ON	ON	ON	ON	ON	8
9	1001		ON	ON	ON	ON	OFF	ON	ON	9
A	1010		ON	ON	ON	OFF	ON	ON	ON	A
B	1011		OFF	OFF	ON	ON	ON	ON	ON	b
C	1100		ON	OFF	OFF	ON	ON	ON	OFF	C
D	1101		OFF	ON	ON	ON	ON	OFF	ON	d
E	1110		ON	OFF	OFF	ON	ON	ON	ON	E
F	1111		ON	OFF	OFF	OFF	ON	ON	ON	F

API	Mnemonic	Operands	Function	Controllers
74	SEGL	S D n	Seven Segment with Latch	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	SEGL: 7 steps
D		*														
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source device to be displayed in 7-segment display **D**: Start device for 7-segment display scan output

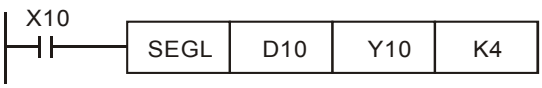
n: Polarity setting of output signal and scan signal

Explanations:

1. Range of **n**: 0 ~ 7. See remarks for more details.
2. For ES series MPU, the instruction can only be used once in the program. For EH series MPU, the instruction can be used twice in the program. For SA series MPU, there is no limitation on the times of using the instruction, but only one instruction can be executed at a time.
3. For ES/EX/SS/SA/SX/SC series MPU, the last digit of **D** should be 0 and it does not support E, F index register modification.
4. Flag: M1029 (SEGL execution completed)
5. This instruction occupies 8 or 12 continuous external input points starting from **D** for displaying 1 or 2 4-digit 7-segment display data and outputs of scanned signals. Every digit carries a 7-segment display drive (to convert the BCD codes into 7-segment display signal). The drive also carries latch control signals to retain the 7-segment display.
6. **n** decides there be 1 group or 2 groups of 4-digit 7-segment display and designates the polarity for the output.
7. When there is 1 group of 4-digit output, 8 output points will be occupied. When there are 2 groups of 4-digit output, 12 output points will be occupied.
8. When this instruction is being executed, the scan output terminals will circulate the scan in sequence. When the drive contact of the instruction goes from Off to On again, the scan output terminal will restart the scan again.

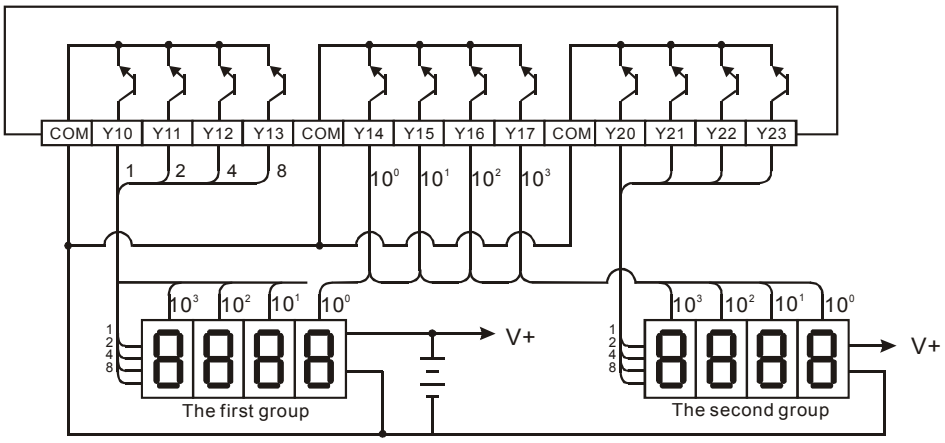
Program Example:

1. When X10 = On, this instruction starts to be executed, Y10 ~ Y17 construct a 7-segment display scan circuit. The value in D10 will be converted into BCD codes and sent to the first group 7-segment display. The value in D11 will be converted into BCD codes as well and sent to the second group 7-segment display. If the values in D10 and D11 exceed 9,999, operational error will occur.



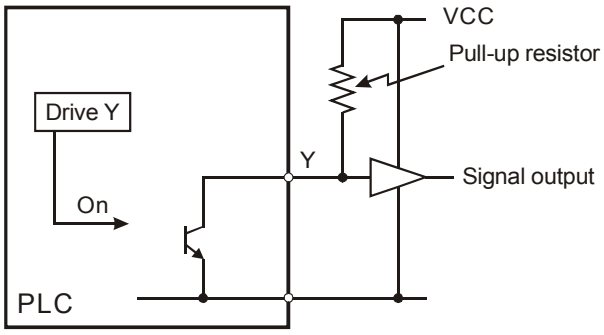
2. When X10 = On, Y14 ~ Y17 will circulate the scan automatically. Every cycle requires 12 scan period. Whenever a cycle is completed, M1029 will be On for a scan period.

3. When there is 1 group of 4-digit 7-segment display, **n** = 0 ~ 3.
   a) Connect the already decoded 7-segment display terminals 1, 2, 4, 8 in parallel an connect them to Y10 ~ Y13 on the PLC. Connect the latch terminals of each digit to Y14 ~ Y17 on the PLC.
   b) When X10 = On, the instruction will be executed and the content in D10 will be sent to the 7-segment displays in sequence by the circulation of Y14 ~ Y17.
4. When there is 2 groups of 4-digit 7-segment display, **n** = 4 ~ 7.
   a) Connect the already decoded 7-segment display terminals 1, 2, 4, 8 in parallel an connect them to Y20 ~ Y23 on the PLC. Connect the latch terminals of each digit to Y14 ~ Y17 on the PLC.
   b) The contents in D10 are sent to the first group 7-segment display. The contents in D11 are sent the the second group 7-segment display. If D10 = K1234 and D11 = K4321, the first group will display 1 2 3 4, and the second group will display 4 3 2 1.
5. Wiring of the 7-segment display scan output:



Remarks:

1. ES/EX/SS series MPU (V4.9 and above) supports this instruction but only supports 1 group of 4-digit 7-segment display and 8 points of output. This instruction can only be used once in the program. Range of **n**: 0 ~ 3.
2. **D** of ES/EX/SS series MPU can only designate Y0.
3. When this instruction is executed, the scan time has to be longer than 10ms. If the scan time is shorter than 10ms, please fix the scan time at 10ms.
4. **n** is for setting up the polarity of the transistor output and the number of groups of the 4-digit 7-segment display.
5. The output point must be a transistor module of NPN output type with open collector outputs. The output has to connect to a pull-up resistor to VCC (less than 30VDC). Therefore, when output point Y is On, the signal output will be in low voltage.



6. Positive logic (negative polarity) output of BCD code

BCD value				Y output (BCDcode)				Signal output			
b_3	b_2	b_1	b_0	8	4	2	1	A	B	C	D
0	0	0	0	0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	1	1	1	1	0
0	0	1	0	0	0	1	0	1	1	0	1
0	0	1	1	0	0	1	1	1	1	0	0
0	1	0	0	0	1	0	0	1	0	1	1
0	1	0	1	0	1	0	1	1	0	1	0
0	1	1	0	0	1	1	0	1	0	0	1
0	1	1	1	0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	1	0	1	1	0

7. Negative logic (positive polarity) output of BCD code

BCD value				Y output (BCDcode)				Signal output			
b_3	b_2	b_1	b_0	8	4	2	1	A	B	C	D
0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	0	0	0	0	1
0	0	1	0	1	1	0	1	0	0	1	0
0	0	1	1	1	1	0	0	0	0	1	1
0	1	0	0	1	0	1	1	0	1	0	0
0	1	0	1	1	0	1	0	0	1	0	1
0	1	1	0	1	0	0	1	0	1	1	0
0	1	1	1	1	0	0	0	0	1	1	1
1	0	0	0	0	1	1	1	1	0	0	0
1	0	0	1	0	1	1	0	1	0	0	1

8. Scan latched signal display

Positive logic (negative polarity)		Negative logic (positive polarity)	
Y output (latch)	Output signal	Y output (latch)	Output signal
1	0	0	1

9. Settings of **n**:

Groups of 7-segment display	1 group				2 groups			
Y output of BCD code	+		−		+		−	
Scan latched signal display	+	−	+	−	+	−	+	−
n	0	1	2	3	4	5	6	7

+: Positive logic (negative polarity) output −: Negative logic (positive polarity) output

10. The polarity of transistor output and the polarity of the 7-segment display input can be the same or different by the setting of **n**.

API	Mnemonic	Operands	Function	Controllers
75	ARWS	S D1 D2 n	Arrow Switch	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*	*												ARWS: 9 steps
D_1											*	*	*	*	*	
D_2		*														
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

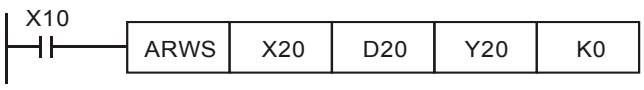
S: Start device for key input **D_1**: Device to be displayed in 7-segment display **D_2**: Start device for 7-segment display scan output **n**: Polarity setting of output signal and scan signal

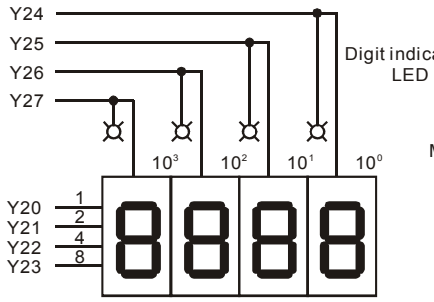
Explanations:

1. **S** will occupy 4 consecutive points.
2. Range of **n**: 0 ~ 3. See remarks of API 74 SEGL for more details.
3. There no limitation on the times of using this instruction in the program. However, only one instruction is allowed to be executed at a time.
4. **S** and **D_2** of SA series MPU do not support E, F index register modification and can only designate the devices whose last digit is 0 (e.g. Y0, Y10….)
5. See the specifications of each model for their range of use.
6. The output points designated by this instruction shall be transistor output.
7. When using this instruction, please fix the scan time, or place this instruction in the time interruption subroutine (I6□□ ~ I8□□).

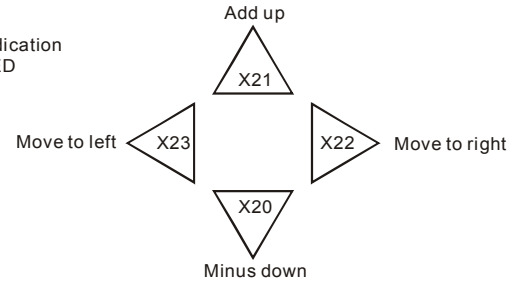
Program Example:

1. When this instruction is executed, X20 is defined as down key, X21 is defined as up key, X22 is defined as right key and X23 is defined as left key. The keys are used for setting up and displaying external set values. The set values (range: 0 ~ 9,999) are stored in D20.
2. When X10 = On, digit 10^3 will be the valid digit for setup. If you press the left key at this time, the valid digit will circulate as $10^3 \rightarrow 10^0 \rightarrow 10^1 \rightarrow 10^2 \rightarrow 10^3 \rightarrow 10^0$.
3. If you press the right key at this time, the valid digit will circulate as $10^3 \rightarrow 10^2 \rightarrow 10^1 \rightarrow 10^0 \rightarrow 10^3 \rightarrow 10^2$. During the circulation, the digit indicators connected Y24 ~ Y27 will also be On interchangeably following the circulation.
4. If you press the up key at this time, the valid digit will change as 0 → 1 → 2 … → 8 → 9 → 0 →1. If you press the down key, the valid digit will change as 0 → 9 → 8 … → 1 → 0 → 9. The changed value will also be displayed in the 7-segment display.





7-segment display for the 4-digit set value



The 4 switches are used for moving the digits and increasing/decreasing set values.

API	Mnemonic	Operands	Function	Controllers
76	ASC	(S) (D)	ASCII Code Conversion	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S																ASC: 11 steps
D											*	*	*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

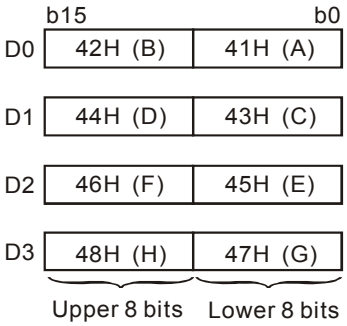
S: English letter to be converted into ASCII code **D**: Device for storing ASCII code

Explanations:

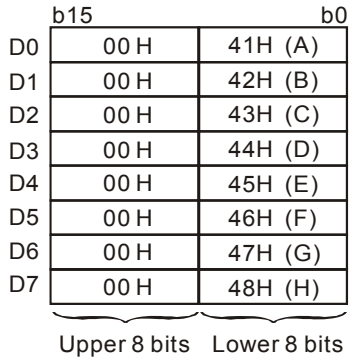
1. **S**: enter 8 Engligh letters by using WPLSoft on computer or enter ASCII code by HPP.
2. See the specifications of each model for their range of use.
3. Flag: M1161 (8/16 bit mode switch)
4. If the execution of this instruction is connected to a 7-segment display, the error message can be displayed by English letters.

Program Example:

1. When X0 = On, convert A ~ H into ASCII code and stored it in D0 ~ D3.



2. When M1161 = On, every ASCII code converted from the letters will occupy the lower 8 bits (b7 ~ b0) of a register. The upper 8 bits are invalid (filled by 0). One register stores a letter.



API	Mnemonic	Operands	Function	Controllers
77	PR	(S) (D)	Print (ASCII Code Output)	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S											*	*	*			PR: 5 steps
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

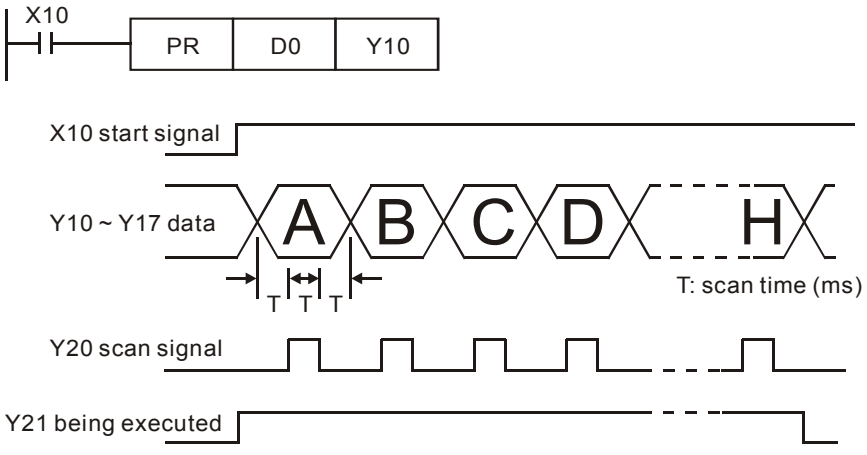
S: Device for storing ASCII code **D**: External ASCII code output points

Explanations:

1. **S** will occupy 4 consecutive points.
2. **D** will occupy 10 consecutive points.
3. This instruction can only be used twice in the program.
4. See the specifications of each model for their range of use.
5. Flags: M1029 (PR execution completed); M1027 (number of PR outputs)
6. This instruction will output the ASCII codes in the 4 registers starting from **S** from the output devices in the order designated in **D**.

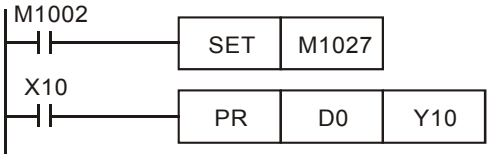
Program Example 1:

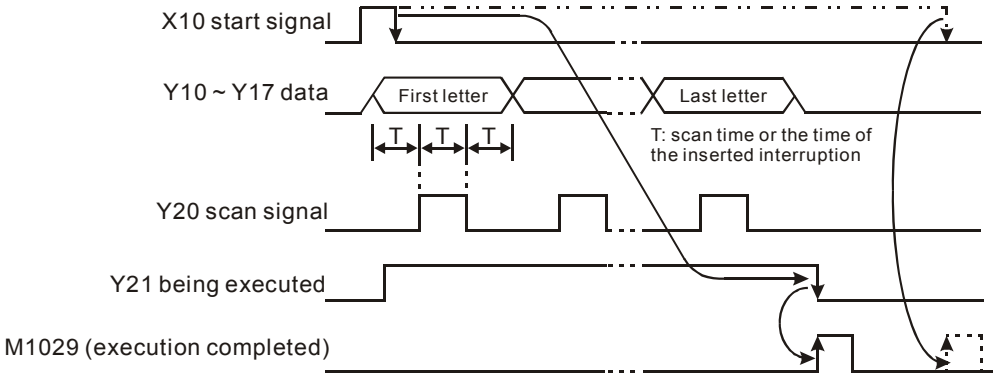
1. Use API 76 ASC to convert A ~ H into ASCII codes and store them in D0 ~ D3 and use this instruction to output the codes in sequence.
2. When M1027 = Off and X10 goes On, the instruction will be executed. Designate Y10 (low bits) ~ Y17 (high bits) as the data output points and Y20 for scan signals. Designate Y21 for the monitor signals during the execution. In this mode, you can execute an output for 8 letters in sequence. During the output, if the drive contact goes Off, the data output will stop immediately and all the outputs will go Off.
3. During the execution of the instruction, when X10 goes Off, all the data output will be interrupted. When X10 is On again, the output will be restarted.



Program Example 2:

1. PR instruction is for outputing a string of 8 bits. When the special auxiliary relay M1027 = Off, PR is able to execute an output of maximum 8 letters in string. When M1027 = On, PR is able to execute an output of 1 ~ 16 letters in string.
2. When M1027 = On and X10 goes from Off to On, the instruction will be executed. Designate Y10 (low bits) ~ Y17 (high bits) as the data output points and Y20 for scan signals. Designate Y21 for the monitor signals during the execution. In this mode, you can execute an output for 16 letters in sequence. During the output, if the drive contact goes Off, the data output will stop after it is completed.
3. When the string encounters 00H (NUL), the string output will finish. The letters coming after it will not be processed.
4. When X10 goes from On to Off, the data output will automatically stop after one cycle. If X10 keeps being On, M1029 will not be enabled.





Remarks:

1. Please use transistor output for the output designated by this instruction.
2. When using this instruction, please fix the scan time or place this instruction in a timed interruption subroutine.

API	Mnemonic			Operands	Function	Controllers		
78	D	FROM	P	(m1) (m2) (D) (n)	Read CR Data in Special Modules	ES/EX/SS	SA/SX/SC	EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
m_1					*	*							*			FROM, FROMP: 9 steps
m_2					*	*							*			DFROM, DFROMP: 17 steps
D								*	*	*	*	*	*	*	*	
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

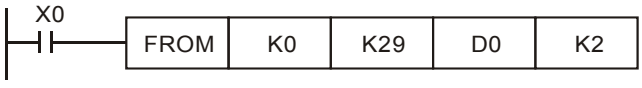
m_1: No. of special module **m_2**: CR# in special module to be read **D**: Device for storing read data **n**: Number of data to be read at a time

Explanations:

1. Range of **m_1** (16-bit and 32-bit): for ES/SA: 0 ~ 7, for EH/EH2 :0 ~ 255, for SV: 0 ~ 107.
2. Range of **m_2** (16-bit and 32-bit): for ES/SA: 0 ~ 48, for EH: 0 ~ 254, for EH2/SV: 0 ~ 499.
3. Range of **n**:
   a) 16-bit: for ES/SA: 1 ~ (49 – **m_2**), for EH: 1 ~ (255 – **m_2**), for EH2/SV: 1 ~ (500 – **m_2**).
   b) 32-bit: for ES/SA: 1 ~ (49 – **m_2**)/2, for EH: 1 ~ (255 – **m_2**)/2, for EH2/SV: 1 ~ (500 – **m_2**)/2.
4. ES series MPU does not support E, F index register modification.
5. **m_1**, **m_2** and **n** of EH series MPU do not support word device D.
6. Flag: M1083 (On when allowing interruptions during FROM/TO instruction).
7. This instruction is for reading the data in the CR in special modules.
8. The 16-bit instruction can designate **D** = K1 ~ K4; the 32-bit instruction can designate **D** = K1 ~ K8.

Program Example:

1. Read CR#29 of special module No.0 into D0 and CR#30 into D1. Only 2 groups of data is read at a time (n = 2).
2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the data read will not be changed.



API	Mnemonic			Operands	Function	Controllers
79	D	TO	P	m1 m2 S n	Write CR Data into Special Modules	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
m_1					*	*							*			TO, TOP: 9 steps
m_2					*	*							*			DTO, DTOP: 17 steps
S					*	*	*	*	*	*	*	*	*	*	*	
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

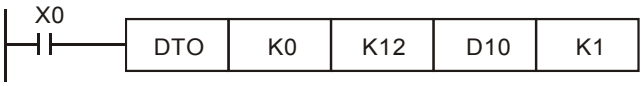
m_1: No. of special module **m_2**: CR# in special module to be written **S**: Data to be written in CR **n**: Number of data to be written at a time

Explanations:

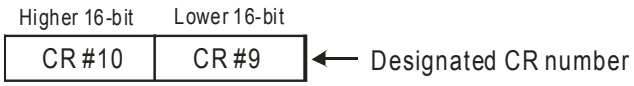
1. Range of **m_1** (16-bit and 32-bit): for ES/SA: 0 ~ 7, for EH/EH2: 0 ~ 255, for SV: 0 ~ 107.
2. Range of **m_2** (16-bit and 32-bit): for ES/SA: 0 ~ 48, for EH: 0 ~ 254, for EH2/SV: 0 ~ 499.
3. Range of **n**:
   a) 16-bit: for ES/SA: 1 ~ (49 – **m_2**), for EH: 1 ~ (255 – **m_2**), for EH2/SV: 1 ~ (500 – **m_2**).
   b) 32-bit: for ES/SA: 1 ~ (49 – **m_2**)/2, for EH: 1 ~ (255 – **m_2**)/2, for EH2/SV: 1 ~ (500 – **m_2**)/2.
4. ES series MPU does not support E, F index register modification.
5. **m_1**, **m_2** and **n** of EH series MPU do not support word device D.
6. Flag: M1083 (On when allowing interruptions during FROM/TO instruction). See remarks for more details.
7. This instruction is for writing the data into the CR in special modules.
8. The 16-bit instruction can designate **S** = K1 ~ K4; the 32-bit instruction can designate **S** = K1 ~ K8.

Program Example:

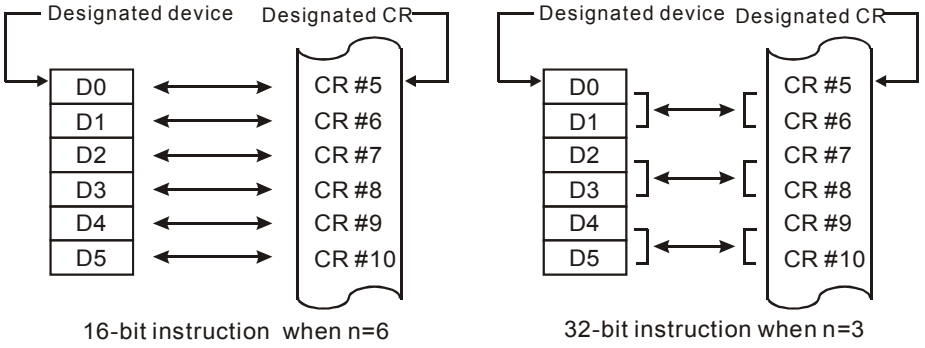
1. Use 32-bit instruction DTO to write the content in D11 and D10 into CR#13 and CR#12 of special module No.0. Only 1 group of data is written in at a time (n = 1).
2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the data written will not be changed.



3. Operand rules
   a) **m_1**: The No. of special modules connected to PLC MPU. No. 0 is the module closest to te MPU. Maximum 8 modules are allowed to connected to a PLC MPU and they will not occupy any I/O points.
   b) **m_2**: CR#. CR (control register) is the n 16-bit memories built in the special module, numbered in decimal as #0 ~ #n. All operation status and settings of the special module are contained in the CR.
   c) FROM/TO instruction is for reading/writing 1 CR at a time. DFROM/DTO instruction is for reading/writing 2 CRs at a time.



d) Number of groups “n” to be transmitted: n = 2 in 16-bit instructions and n = 1 in 32-bit instructions mean the same.



4. ES/EX/SS series MPU does not have M1083. During the execution of FROM/TO instruction, all external or internal interruption subroutines will be forbidden. The interruptions are allowed only after FROM/TO instruction finishes its execution. FROM/TO instruction can also be used in an interruption subroutine.
5. M1083 for switching instruction modes in SA/SX/SC/EH/EH2/SV series MPU:
   a) When M1083 = Off, during the execution of FROM/TO instruction, all external or internal interruption subroutines will be forbidden. The interruptions are allowed only after FROM/TO instruction finishes its execution. FROM/TO instruction can also be used in an interruption subroutine.
   b) When M1083 = On and an interruption signal occurs during the execution of FROM/TO instruction, the interruption will be processed first (with a 100us delay) and the execution of FROM/TO will be stopped. After the interruption subroutine finishes its execution, the program will jump to the next instructio of FROM/TO. FROM/TO cannot be used in an interruption subroutine.

FROM/TO Application Example 1:

Adjust the A/D conversion curve of DVP-04AD. Set the OFFSET value of CH1 as 0V (= $K0_{LSB}$) and GAIN value as 2.5V (= $K2{,}000_{LSB}$).

1. Write H0 to CR#1 of anlog input module No. 0 and set CH1 as mode 0 (voltage input: -10V ~ +10V).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 ~ CH4.
3. When X0 goes from Off to On, write the OFFSET value $K0_{LSB}$ into CR#18 and the GAIN value $K2{,}000_{LSB}$ into

CR#24.

FROM/TO Application Example 2:

Adjust the A/D conversion curve of DVP-04AD. Set the OFFSET value of CH2 as 2mA (= $K400_{LSB}$) and GAIN value as 18mA (= $K3{,}600_{LSB}$).

1. Write H18 to CR#1 of anlog input module No. 0 and set CH2 as mode 3 (current input: -20mA ~ +20mA).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 ~ CH4.
3. When X0 goes from Off to On, write the OFFSET value $K400_{LSB}$ into CR#19 and the GAIN value $K3{,}600_{LSB}$ into CR#25.

FROM/TO Application Example 3:

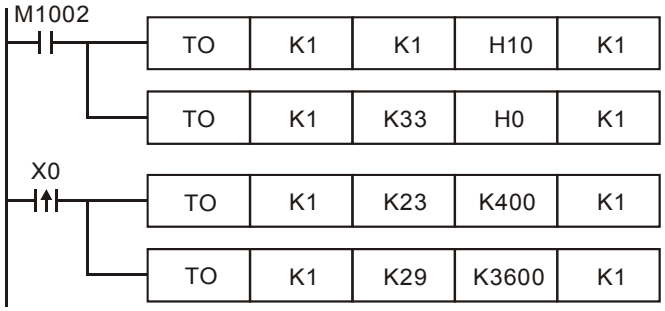
Adjust the D/A conversion curve of DVP-02DA. Set the OFFSET value of CH2 as 0mA (= $K0_{LSB}$) and GAIN value as 10mA (= $K1{,}000_{LSB}$).

1. Write H18 to CR#1 of anlog output module No. 1 and set CH2 as mode 3 (current output: 0mA ~ +20mA).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 and CH2.
3. When X0 goes from Off to On, write the OFFSET value $K0_{LSB}$ into CR#22 and the GAIN value $K1{,}000_{LSB}$ into CR#28.

FROM/TO Application Example 4:

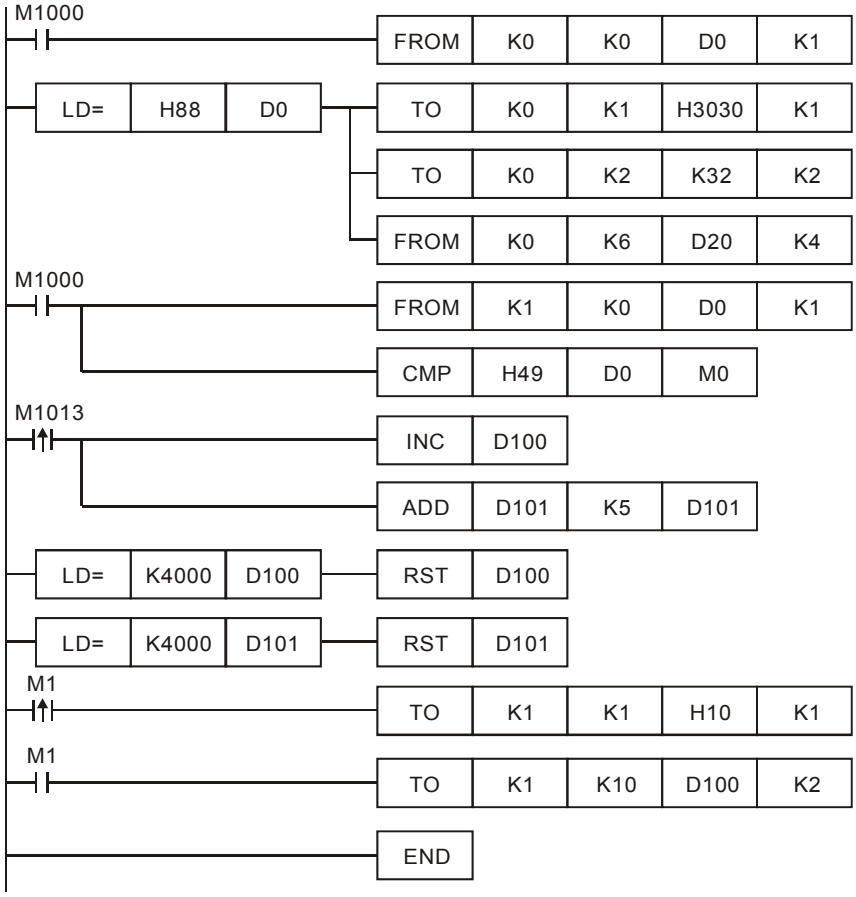
Adjust the D/A conversion curve of DVP-02DA. Set the OFFSET value of CH2 as 2mA (= $K400_{LSB}$) and GAIN value as 18mA (= $K3{,}600_{LSB}$).

1. Write H10 to CR#1 of anlog output module No. 1 and set CH2 as mode 2 (current output: +4mA ~ +20mA).
2. Write H0 to CR#33 and allow OFFSET/GAIN tuning in CH1 and CH2.
3. When X0 goes from Off to On, write the OFFSET value $K400_{LSB}$ into CR#23 and the GAIN value $K3{,}600_{LSB}$ into CR#29.



FROM/TO Application Example 5:

When DVP-04AD-S is used with DVP-02DA-S



1. Read CR#0 of the extension module No. 0 and see if it is DVP-04AD-S: H88.
2. If D0 = H88, set the input modes: (CH1, CH3) mode 0, (CH2, CH4) mode 3.
3. Set the average times in CH1 and CH2 from CR#2 and CR#3 as K32.
4. Read the average of input signals at CH1 ~ CH4 from CR#6 ~ CR#9 and store the 4 data in D20 ~ D23.
5. Read CR#0 of the extension module No. 1 and see if it is DVP-02DA-S: H49.
6. D100 increases K1 and D101 increases K5 every second.
7. When D100 and D101 reach K4,000, they will be cleared as 0.
8. See if the model is DVP-02DA-S when M1 = On. If so, set up output mode: CH1 in mode 0 and CH2 is mode 2.
9. Write the output settings of D100 and D101 into CR#10 and CR#11. The analog output will change by the changes in D100 and D101.

API	Mnemonic	Operands	Function	Controllers
80	RS	(S) (m) (D) (n)	Serial Communication Instruction	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			RS: 9 steps
m					*	*							*			
D													*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start device for the data to be transmitted **m**: Length of data to be transmitted **D**: Start device for receiving data **n**: Length of data to be received

Explanations:

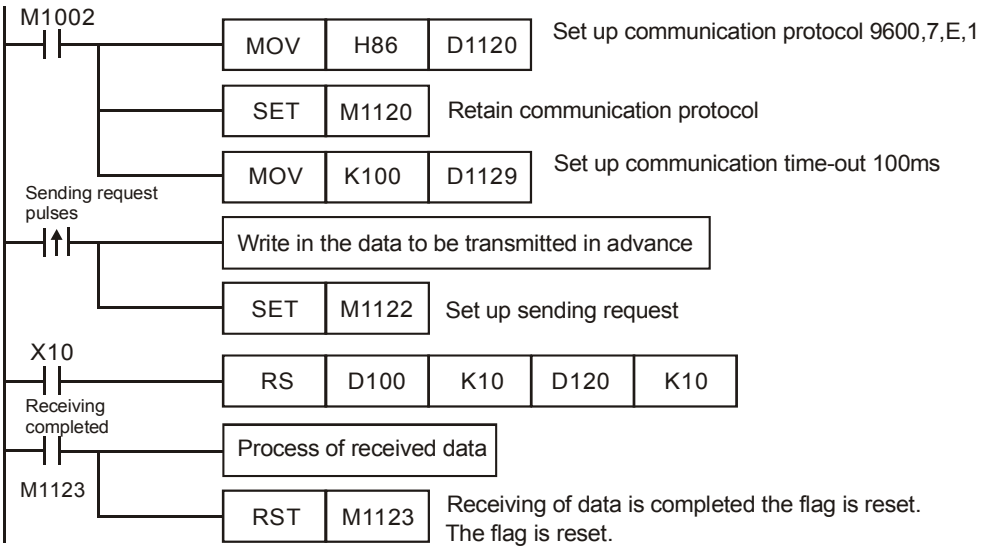
1. Range of **m**: 0 ~ 256
2. Range of **n**: 0 ~ 256
3. See the specifications of each model for their range of use.
4. ES series MPU does not support E, F index register modification.
5. This instruction is a handy instruction exclusively for MPU to use RS-485 serial communication interface. The user has to pre-store word data in **S** data register, set up data length **m** and the data receiving register **D** and received data length **n**. If E, F index registers are used to modify **S** and **D**, the user cannot change the set values of E and F when the instruction is being executed; otherwise errors may cause in data writing or reading.
6. Designate **m** as K0 if you do not need to send data. Designate **n** as K0 if you do not need to receive data.
7. There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed at a time.
8. During the execution of RS instruction, changing the data to be transmitted will be invalid.
9. If the peripheral devices, e.g. AC motor drive, are equipped with RS-485 serial communication and its communication format is open, you can use RS instruction to design the program for the data transmission between PLC and the peripheral device.
10. If the communication format of the peripheral device is Modbus, DVP series PLC offers handy communication instructions API 100 MODRD, API 101 MODWR, and API 150 MODRW, to work with the device. See explanations of the instructions in this application manual.
11. For the special auxiliary relays M1120 ~ M1161 and special data registers D1120 ~ D1131 relevant to RS-485 communication, see remarks for more details.

Program Example 1:

1. Write the data to be transmitted in advance into registers starting from D100 and set M1122 (sending request flag) as On.
2. When X10 = On, RS instruction will be executed and PLC will start to wait for the sending and receiving of data. D100 starts to continuousl send out 10 data and when the sending is over, M1122 will be automatically reset to

Off (DO NOT use the program to execute RST M1122). After 1ms of waiting, PLC will start to receive the 10 data. Store the data in consecutive registers starting from D120.

3. When the receiving of data is completed, M1123 will automatically be On. After the program finishes processing the received data, M1123 has to be reset to Off and the PLC will start to wait for the sending and receiving of data again. DO NOT use the program to continuously execute RST M1123.

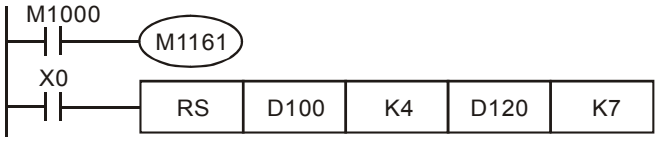


Program Example 2:

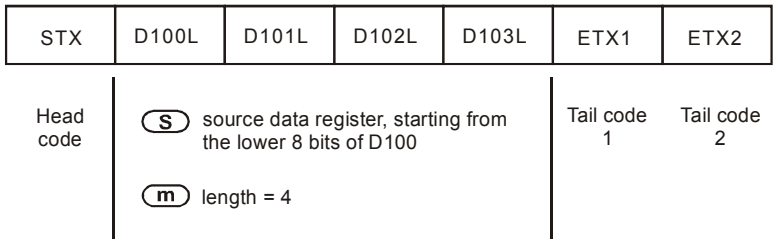
Switching between 8-bit mode (M1161 = On) and 16-bit mode (M1161 = Off)

1. 8-bit mode:

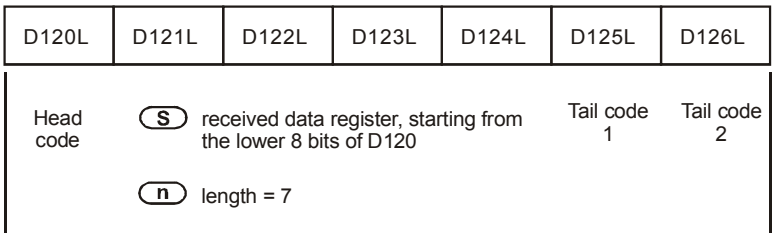
   The head code and tail code of the data are set up by M1126 and M1130 together with D1124 ~ D1126. When PLC is executing RS instruction, the head code and tail code set up by the user will be sent out automatically. M1161 = On indicates PLC in 8-bit conversion mode. The 16-bit data will be divided into the higher 8 bits and lower 8 bits. The higher 8 bits are ignored and only the lower 8 bits are valid for data transmission.



Sending data: (PLC -> external equipment)



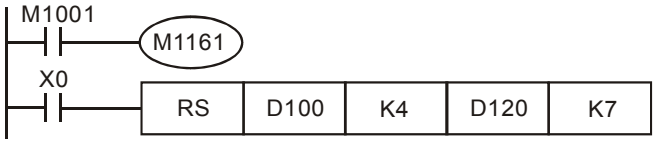
Receiving data: (External equipment -> PLC)



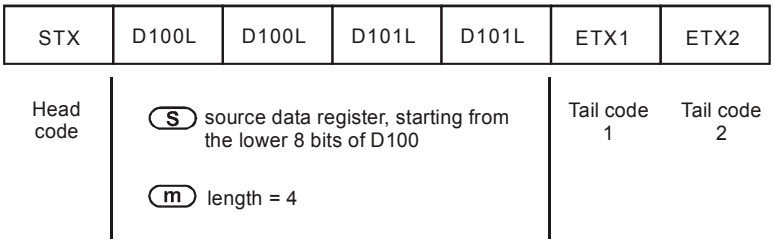
When receiving data, PLC will receive the head code and tail code of the data from the external equipment; therefore, the user has to be aware of the setting of data length **n**.

2. 16-bit mode:

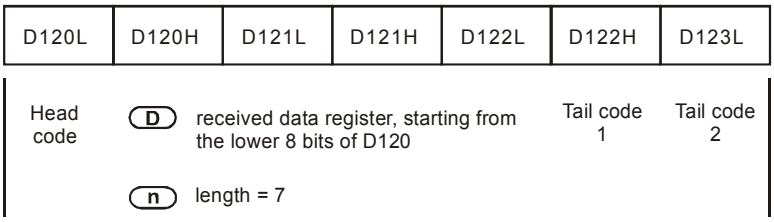
The head code and tail code of the data are set up by M1126 and M1130 together with D1124 ~ D1126. When PLC is executing RS instruction, the head code and tail code set up by the user will be sent out automatically. M1161 = Off indicates PLC in 16-bit conversion mode. The 16-bit data will be divided into the higher 8 bits and lower 8 bits for data transmission.



Sending data: (PLC -> external equipment)



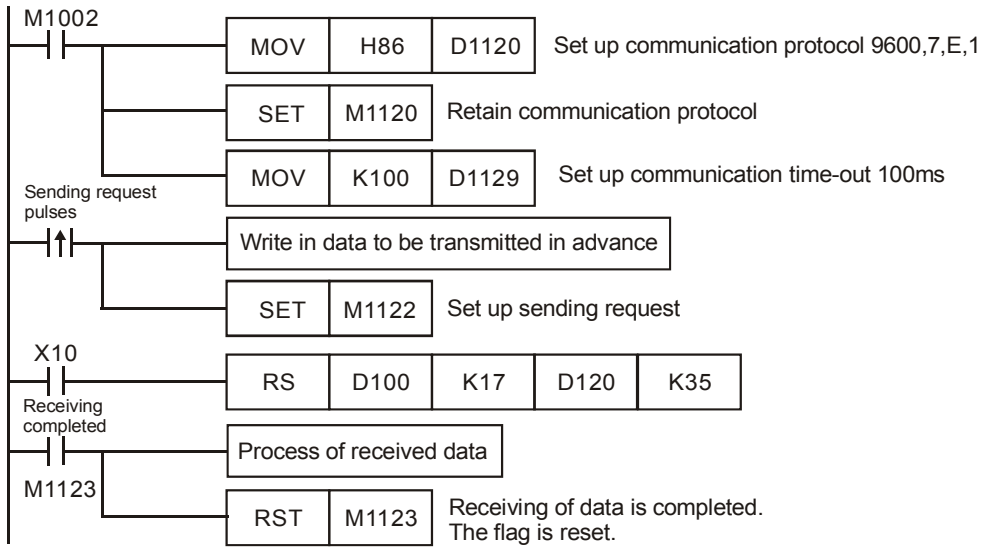
Receiving data: (External equipment -> PLC)



When receiving data, PLC will receive the head code and tail code of the data from the external equipment; therefore, the user has to be aware of the setting of data length **n**.

Program Example 3:

Connect PLC to VFD-B series AC motor drives (AC motor drive in ASCII Mode; PLC in 16-bit mode and M1161 = Off). Write in the 6 data starting from parameter address H2101 in VFD-B in advance as the data to be transmitted.



PLC ⇨ VFD-B, PLC sends **": 01 03 2101 0006 D4 CR LF "**

VFD-B ⇨ PLC, PLC receives **": 01 03 0C 0100 1766 0000 0000 0136 0000 3B CR LF "**

Registers for sent data (PLC sends out message)

Register	Data		Explanation	
D100 low	': '	3A H	STX	
D100 high	'0'	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D101 low	'1'	31 H	ADR 0	
D101 high	'0'	30 H	CMD 1	Instruction code: CMD (1,0)
D102 low	'3'	33 H	CMD 0	
D102 high	'2'	32 H	Start data address	
D103 low	'1'	31 H		
D103 high	'0'	30 H		
D104 low	'1'	31 H		
D104 high	'0'	30 H	Number of data (counted by words)	
D105 low	'0'	30 H		
D105 high	'0'	30 H		
D106 low	'6'	36 H		
D106 high	'D'	44 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
D107 low	'4'	34 H	LRC CHK 0	
D107 high	CR	D H	END	
D108 low	LF	A H		

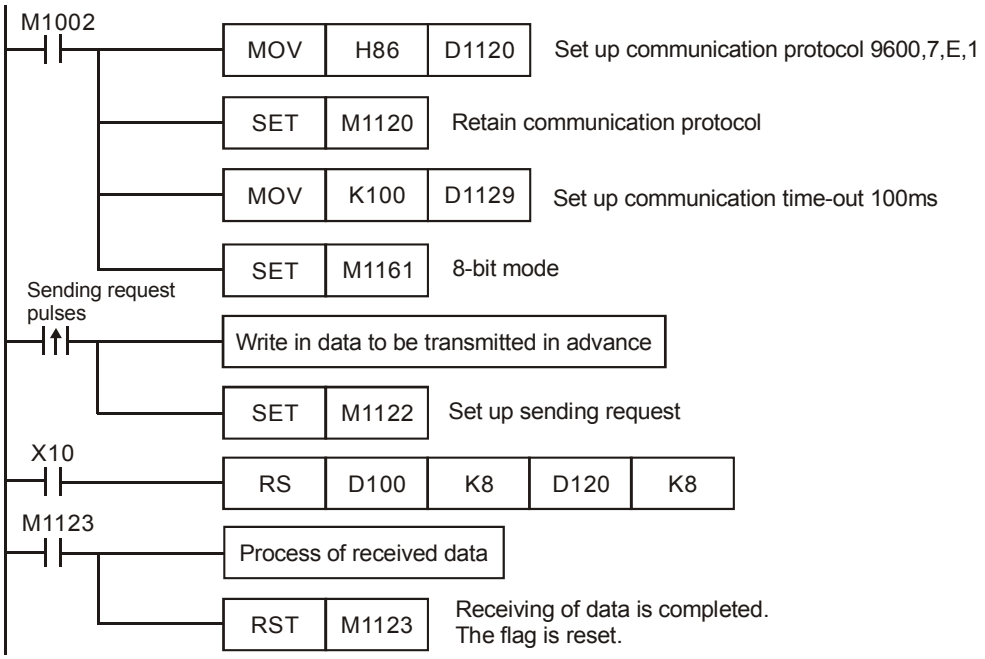
Registers for received data (VFD-B responds with messages)

Register	Data		Explanation
D120 low	': '	3A H	STX
D120 high	'0'	30 H	ADR 1
D121 low	'1'	31 H	ADR 0
D121 high	'0'	30 H	CMD 1
D122 low	'3'	33 H	CMD 0
D122 high	'0'	30 H	Number of data (counted by byte)
D123 low	'C'	43 H	

Register	Data		Explanation
D123 high	'0'	30 H	Content of address 2101 H
D124 low	'1'	31 H	
D124 high	'0'	30 H	
D125 low	'0'	30 H	
D125 high	'1'	31 H	Content of address 2102 H
D126 low	'7'	37 H	
D126 high	'6'	36 H	
D127 low	'6'	36 H	
D127 high	'0'	30 H	Content of address 2103 H
D128 low	'0'	30 H	
D128 high	'0'	30 H	
D129 low	'0'	30 H	
D129 high	'0'	30 H	Content of address 2104 H
D130 low	'0'	30 H	
D130 high	'0'	30 H	
D131 low	'0'	30 H	
D131 high	'0'	30 H	Content of address 2105 H
D132 low	'1'	31 H	
D132 high	'3'	33 H	
D133 low	'6'	36 H	
D133 high	'0'	30 H	Content of address 2106 H
D134 low	'0'	30 H	
D134 high	'0'	30 H	
D135 low	'0'	30 H	
D135 high	'3'	33 H	LRC CHK 1
D136 low	'B'	42 H	LRC CHK 0
D136 high	CR	D H	END
D137 low	LF	A H	

Program Example 4:

Connect PLC to VFD-B series AC motor drives (AC motor drive in RTU Mode; PLC in 16-bit mode and M1161 = On). Write in H12 to parameter address H2000 in VFD-B in advance as the data to be transmitted.



PLC ⇨ VFD-B, PLC sends: **01 06 2000 0012 02 07**

VFD-B ⇨ PLC, PLC receives: **01 06 2000 0012 02 07**

Registers for sent data (PLC sends out messages)

Register	Data	Explanation
D100 low	01 H	Address
D101 low	06 H	Function
D102 low	20 H	Data address
D103 low	00 H	
D104 low	00 H	Data content
D105 low	12 H	
D106 low	02 H	CRC CHK Low
D107 low	07 H	CRC CHK High

Registers for received data (VFD-B responds with messages)

Register	Data	Explanation
D120 low	01 H	Address
D121 low	06 H	Function
D122 low	20 H	Data address
D123 low	00 H	
D124 low	00 H	Data content
D125 low	12 H	
D126 low	02 H	CRC CHK Low
D127 low	07 H	CRC CHK High

Remarks:

1. Flags for the RS-485 communication of RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions

Flag	Function	Action
M1120	For retain the communication setting. After the first program scan is completed, the communication setting will be reset according to the setting in the special data register D1120. When the second program scan starts and RS instruction is being executed, the communication settings will all be reset according to the settings in D1120. If your communication protocol is fixed, you can set M1120 to On and the communication protocol will not be reset whenever RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instruction is executed. In this case, even the settings in D1120 are modified, the communication protocol will not be changed.	Set up and reset by the user.
M1121	Off when the RS-485 communication data is being transmitted.	By the system.
M1122	Sending request. When you need to send out or receive data by RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions, you have to set M1122 to On by a pulse instruction. When these instructions start to execute, PLC will start to send out or receive data. When the data transmission is completed, M1122 will be reset automatically.	Set up by the user; reset automatically by the system.
M1123	Receiving is completed. When the execution of RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions is completed, M1123 will be set to On. You can process the data received when M1123 is On in the program. You have to reset M1123 to Off when the process of received data is completed.	Set up automatically by the system; reset by the user.
M1124	Waiting for receiving. On when PLC is waiting for receiving data.	By the system.
M1125	Receiving status cleared. When M1125 = On, the waiting for receiving status of PLC will be cleared. You have to reset M1125 to Off after the staus is cleared.	Set up and reset by the user.
M1126	User/system defined STX/ETX selection of RS instruction (see the next table for details.)	
M1130	User/system defined STX/ETX selection of RS instruction (see the next table for details.)	
M1127	Data transmission is completed for communication instructions (RS instruction not included)	Set up automatically by the system; reset by the user.
M1128	Data being sent/received indication	By the systme.
M1129	Receiving time-out. If you already set up a communication time-out in D1129 and the data have not been received completey when the time-out set is reached, M1129 will be On. You have to reset M1129 to Off after the problem is solved.	Set up automatically by the system; reset by the user.
M1131	On when the data are converted into hex of MODRD/RDST/MODRW instructions when in ASCII mode; otherwise, M1131 is Off.	By the system
M1140	Data receiving error of MODRD/MODWR/MODRW instructions	
M1141	Parameter error of MODRD/MODWR/MODRW instructions	
M1142	Data receiving error of VFD-A handy commands	
M1143	ASCII/RTU mode selection (used with MODRD/MODWR/MODRW instructions). On = RTU; Off = ASCII	Set up and reset by the user.
M1161	8/16-bit mode selection. On = 8-bit; Off = 16-bit	

2. Special data register for the RS-485 communication of RS/MODRD/MODWR/FWD/REV/STOP/RDST/RSTEF /MODRW instructions

Special D	Function
D1038	For setting up the data responding delay time when a PLC MPU using RS-485 communication is used as a slave. Range: 0 ~ 10,000 (unit: 0.1ms)
D1050 ~ D1055	When MODRD/RDST instruction is executed, PLC will automatically convert the ASCII characters in D1070 ~ D1085 into hex and store the hex value in D1050 ~ D1055.
D1070 ~ D1085	When the RS-485 communication instructions built in PLC are executed, the receiving end will respond with a message and the messages will be stored in D1070 ~ D1085. You can check on the responded data stored in these registers (not applicable for RS instruction).
D1089 ~ D1099	When the RS-485 communication instructions built in PLC are executed, the data sent will be stored in D1089 ~ D1099. You can check on whether the data sent are correct by checking these registers (not applicable for RS instruction).
D1120	RS-485 communication protocol. See the next table for more details.
D1121	The communication address of PLC when it operates as a slave.
D1122	Remaining number of words of the data being sent
D1123	Remaining number of words of the data being received
D1124	Definition of the start word (STX). See the table above for more details.
D1125	Definition of the first end word (ETX1) of RS instruction. See the table above for more details.
D1126	Definition of the second end word (ETX2) of RS instruction. See the table above for more details.
D1129	Abnormal communication time-out (in ms). When D1129 = 0, there will be no time-out occurring. When D1129 > 0 and RS/MODRD/ MODWR/FWD/REV/STOP/RDST/RSTEF/MODRW instructions are being executed, if the first word has not been received within designated time or the time interval between any two words exceeds the value (>0) after PLC enters the receiving mode, PLC will automatically set M1129 to On. You can also use M1129 for handling the communication time-out. Please be noted that you have to reset M1129 after the time-out.
D1130	Error code sent back by Modbus
D1168	For RS instruction, when the received number of words = the low byte of D1168, the interruption I150 will be triggered.
D1169	For RS instruction, when the received data length = the low byte of D1169, the interruption I160 will be triggered. When D1169 = 0, I160 will not be triggered.
D1256 ~ D1295	When the RS-485 communication instruction MODRW built in PLC is executed, the data sent will be stored in D1256 ~ D1295. You can check on whether the data sent are correct by checking these registers.
D1296 ~ D1311	For MODRW instruction, PLC will automatically convert the ASCII characters into hex.

3. How to set up RS-485 communication protocol in D1120

	Content	0	1
b0	Data length	7	8
b1 **b2**	Parity bits	00: None 01: Odd 11: Even	
b3	Stop bits	1 bit	2 bits
b4 **b5** **b6** **b7**	0001 (H1) : 110 0010 (H2) : 150 0011 (H3) : 300 0100 (H4) : 600 0101 (H5) : 1200 0110 (H6) : 2400 0111 (H7) : 4800 1000 (H8) : 9600 1001 (H9) : 19200 1010 (HA) : 38400 1011 (HB) : 57600 (does not support ES/SS V5.8 and below) 1100 (HC) : 115200 (does not support ES/SS V5.8 and below)		
b8	Start word	None	D1124
b9	First end word	None	D1125
b10	Second end word	None	D1126
b15 ~ b11	Not defined		

4. When RS instruction is in use, the frequently used communication format in the peripheral device will define the start word and end word of the control string. Therefore, you can set up the start word and end word in D1124 ~ D1126 or use the start word and end word defined by the PLC. When you use M1126, M1130 and D1124 ~ D1126 to set up the start word and end word, b8 ~ b10 of D1120 have to be set as 1 to make valid the RS-485 communication protocol. See the table below for how to set up.

		M1130	
		0	1
M1126	0	D1124: user defined D1125: user defined D1126: user defined	D1124: H 0002 D1125: H 0003 D1126: H 0000 (no setting)
	1	D1124: user defined D1125: user defined D1126: user defined	D1124: H 003A (':') D1125: H 000D (CR) D1126: H 000A (LF)

5. Example of how to set up the communication format:

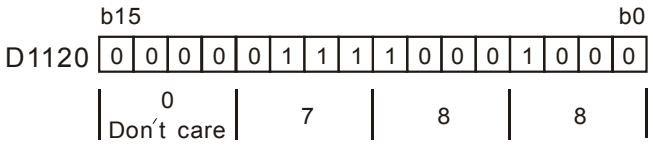
Assume there is a communication format: Baud rate 9600 7, N, 2

STX : ":"

ETX1 : "CR"

ETX2 : "LF"

Check the table and obtain the communication format H788 and write it into D1120.



When STX, ETX1 and EXT2 are in use, please be aware of the On and Off of the special auxiliary relays M1126 and M1130.

6. M1143 is for the selection of ASCII mode or RTU mode. On = RTU mode; Off = ASCII mode.

Take the standard Modbus format for example:

In ASCII mode (M1143 = Off)

STX	Start word = ':' (3AH)
Address Hi	Communication address: The 8-bit address consists of 2 ASCII codes
Address Lo	
Function Hi	Function code: The 8-bit function code consists of 2 ASCII codes
Function Lo	
DATA (n-1)	Data: The n × 8-bit data consists of 2n ASCll codes
…….	
DATA 0	
LRC CHK Hi	LRC checksum: The 8-bit checksum consists of 2 ASCll code
LRC CHK Lo	
END Hi	End word: END Hi = CR (0DH), END Lo = LF(0AH)
END Lo	

The communication protocol is in Modbus ASCII mode, i.e. every byte is composed of 2 ASCII characters. For example, 64Hex is '64' in ASCII, composed by '6' (36Hex) and '4' (34Hex). Every hex '0'…'9', 'A'…'F' corresponds to an ASCII code.

Character	'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'
ASCII code	30H	31H	32H	33H	34H	35H	36H	37H

Character	'8'	'9'	'A'	'B'	'C'	'D'	'E'	'F'
ASCII code	38H	39H	41H	42H	43H	44H	45H	46H

Start word (STX):

Fixed as ':' (3AH)

Address:

'0' '0': Broadcasting to all drivers

'0' '1': To the driver at address 01

'0' 'F': To the driver at address 15

'1' '0': To the driver at address 16

….and so on, maximum to the driver at address 255 ('F' 'F')

Function code:

'0' '3': Read contents of many registers

'0' '6': Write 1 word to register

'1' '0': Write in contents of many registers

Data characters: The data sent by the user.

LRC checksum:

LCR checksum is 2's complement of the value added from Address to Data Content.

For example: 01H + 03H + 21H + 02H + 00H + 02H = 29H. 2's complement of 29H = D7H

End word (END):

Fixed as END Hi = CR (0DH), END Lo = LF (0AH)

For example: Read 2 continuous data stored in the registers of the driver at address 01H (see the table below).

The start register is at address 2102H.

Inquiry message:

STX	': '
Address	'0'
	'1'
Function code	'0'
	'3'
Start address	'2'
	'1'
	'0'
	'2'
Number of data (counted by words)	'0'
	'0'
	'0'
	'2'
LRC checksum	'D'
	'7'
END	CR
	LF

Responding message:

STX	': '
Address	'0'
	'1'
Function code	'0'
	'3'
Number of data (counted by byte)	'0'
	'4'
Content in start address 2102H	'1'
	'7'
	'7'
	'0'
Content of address 2103H	'0'
	'0'
	'0'
	'0'
LRC check	'7'
	'1'
END	CR
	LF

In RTU mode (M1143 = On)

START	See the following explanation
Address	Communication address: In 8-bit binary
Function	Function code: In 8-bit binary
DATA (n-1)	Data: n × 8-bit data
…….	
DATA 0	
CRC CHK Low	CRC checksum: 16-bit CRC consists of 2 8-bit binary
CRC CHK High	
END	See the following explanation

START:

For ES/EX/SS/SA/SX series MPU, no input signal can be ≥ 10ms.

See the table below for EH/EH2/SV series MPU:

Baud rate(bps)	RTU timeout timer (ms)	Baud rate (bps)	RTU timeout timer (ms)
300	40	9,600	2
600	21	19,200	1
1,200	10	38,400	1
2,400	5	57,600	1
4,800	3	115,200	1

Address:

00H: Broadcasting to all drivers

01H: To the driver at address 01

0FH: To the driver at address 15

10H: To the driver at address 16…. And so on, maximum to the driver at address 254 (FE H)

Function code:

03H: Read contents of many registers

06H: Write 1 word to register

10H: Write in contents of many registers

Data characters: The data sent by the user.

CRC checksum: Starting from Address and ending at Data Content.

Step 1: Make the 16-bit register (CRC register) = FFFFH

Step 2: Exclusive OR the first 8-bit message and the low 16-bit CRC register. Store the result in the CRC register.

Step 3: Right shift CRC register for a bit and fill “0” into the high bit.

Step 4: Check the value shifted to the right. If it is 0, fill in the new value obtained in step 3 and store the value in CRC register; otherwise, Exclusive OR A001H and CRC register and store the result in the CRC register.

Step 5: Repeat step 3 – 4 and finish operations of all the 8 bits.

Step 6: Repeat step 2 – 5 for obtaining the next 8-bit message until the operation of all the messages are completed. The final value obtained in the CRC register is the CRC checksum. The CRC checksum has to be placed interchangeably in the checksum of the message.

END:

For ES/EX/SS V5.8 (and below) and SA/SX V1.1 (and below) series MPU, keep no input signal be ≥ 10ms.

See the table below for EH/EH2/SV series MPU:

Baud rate(bps)	RTU timeout timer (ms)	Baud rate (bps)	RTU timeout timer (ms)
300	40	9,600	2
600	21	19,200	1
1,200	10	38,400	1
2,400	5	57,600	1
4,800	3	115,200	1

For example: Read 2 continuous data stored in the registers of the driver at address 01H (see the table below). The start register is at address 2102H.

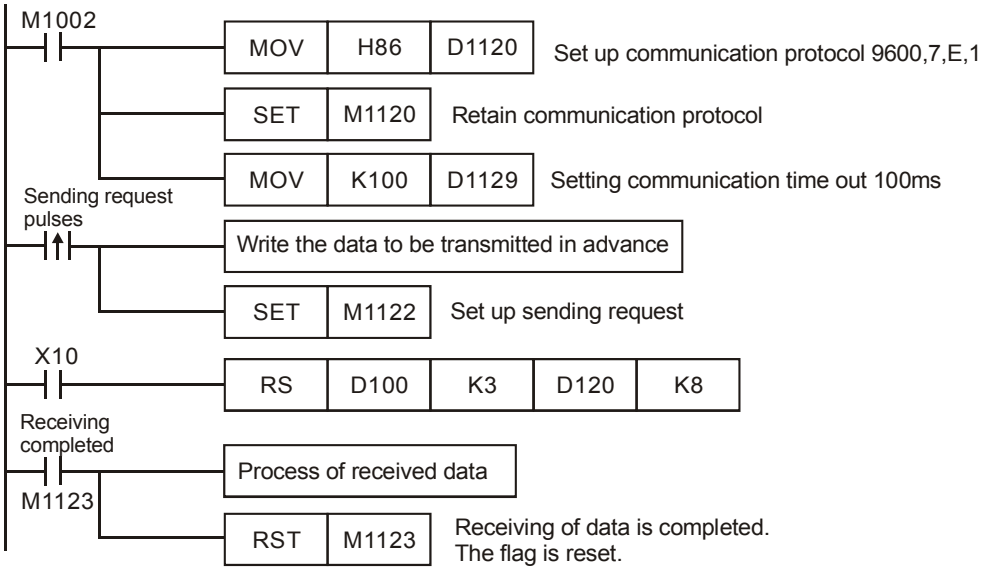
Inquiry message:

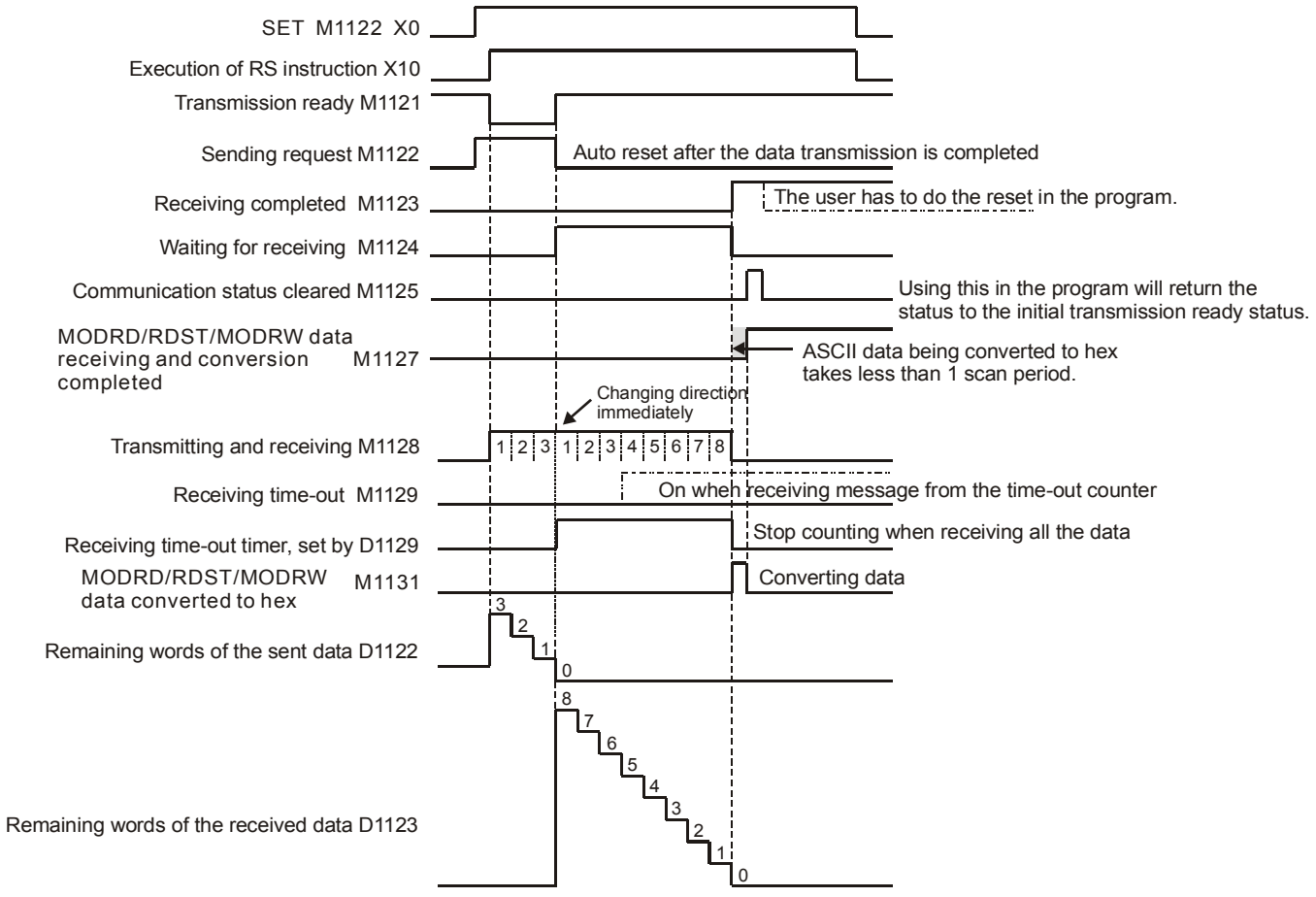
Address	01 H
Function	03 H
Start data address	21 H
	02 H
Number of data (counted by words)	00 H
	02 H
CRC CHK Low	6F H
CRC CHK High	F7 H

Responding message:

Address	01 H
Function	03 H
Number of data (counted by byte)	04 H
Content in data address 8102H	17 H
	70 H
Content in data address 8103H	00 H
	00 H
CRC CHK Low	FE H
CRC CHK High	5C H

7. Timing diagram of RS-485 communication flag:





API	Mnemonic	Operands	Function	Controllers
81	D PRUN P	(S) (D)	Parallel Run	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*		*							PRUN, PRUNP: 5 steps DPRUN, DPRUNP: 9 steps
D								*	*							

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

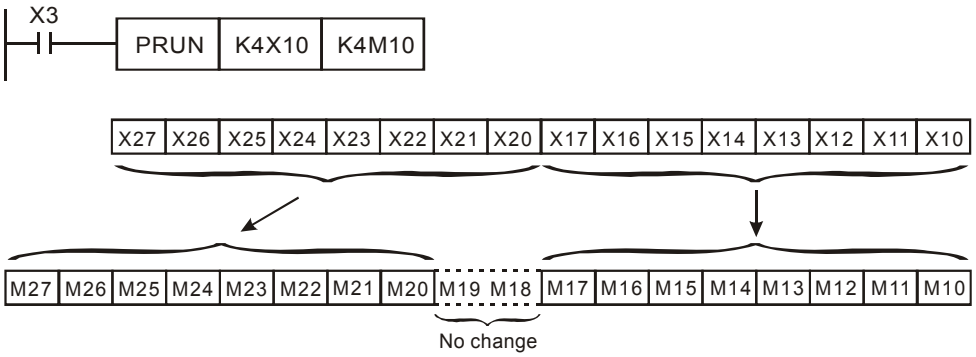
S: Source device **D**: Destination device

Explanations:

1. The most right digit of X, Y and M of KnX, KnY and KnM has to be 0.
2. When **S** designates KnX, **D** has to designate KnM; when **S** designates KnM, **D** has to designate KnY.
3. See the specifications of each model for their range of use.
4. This instruction sends the content in **S** to **D** in the form of octal system.

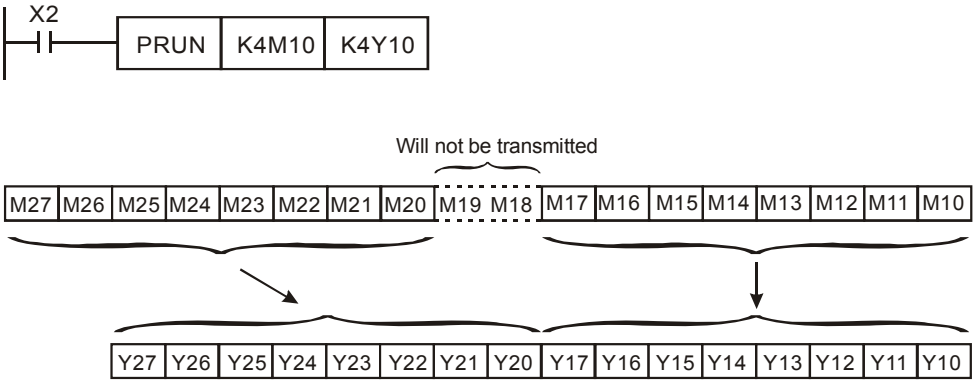
Program Example 1:

When X3 = On, the content in K4X10 will be sent to K4M10 in octal form.



Program Example 2:

When X2 = On, the content in K4M10 will be sent to K4Y10 in octal form.



API	Mnemonic	Operands	Function	Controllers
82	ASCI P	S D n	Converts Hex to ASCII	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*			ASCI, ASCIP: 7 steps
D								*	*	*	*	*	*			
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start device for source data **D**: Start device for storing the converted result **n**: Number of bits to be converted

Explanations:

1. Range of **n**: 1 ~ 256
2. See the specifications of each model for their range of use.
3. Flag: M1161 (8/16 bit mode switch)
4. 16-bit conversion mode: When M1161 = Off, the instruction converts every bit of the hex data in **S** into ASCII codes and send them to the 8 high bits and 8 low bits of **D**. **n** = the converted number of bits.
5. 8-bit conversion mode: When M1161 = On, the instruction converts every bit of the hex data in **S** into ASCII codes and send them to the 8 low bits of **D**. **n** = the number of converted bits. (All 8 high bits of **D** = 0)

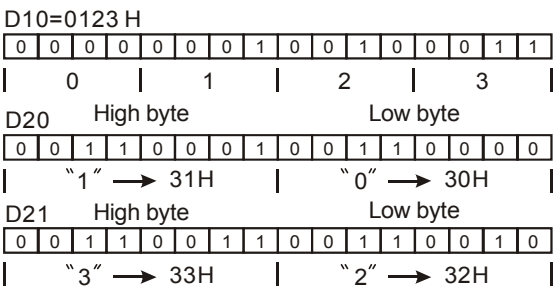
Program Example 1:

1. M1161 = Off: The 16-bit conversion mode
2. When X0 = On, convert the 4 hex values in D10 into ASCII codes and send the result to registers starting from D20.

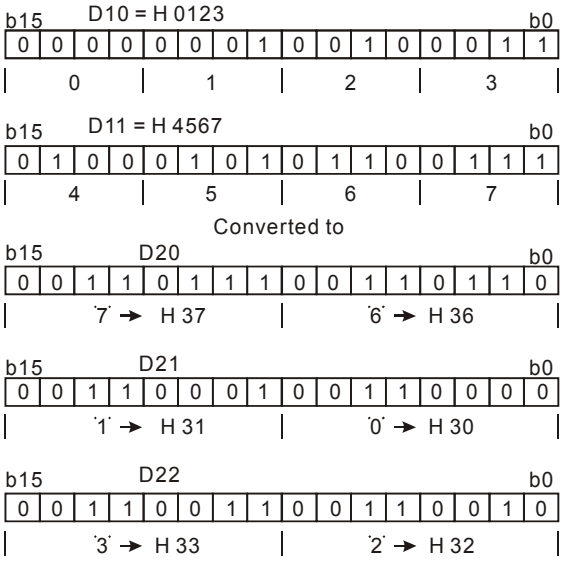
3. Assume

(D10) = 0123 H	'0' = 30H	'4' = 34H	'8' = 38H
(D11) = 4567 H	'1' = 31H	'5' = 35H	'9' = 39H
(D12) = 89AB H	'2' = 32H	'6' = 36H	'A' = 41H
(D13) = CDEF H	'3' = 33H	'7' = 37H	'B' = 42H

4. When **n** = 4, the bit structure will be as:



5. When **n** = 6, the bit structure will be as:



6. When **n** = 1 ~ 16:

D \ n	K1	K2	K3	K4	K5	K6	K7	K8
D20 Low byte	"3"	"2"	"1"	"0"	"7"	"6"	"5"	"4"
D20 High byte		"3"	"2"	"1"	"0"	"7"	"6"	"5"
D21 Low byte			"3"	"2"	"1"	"0"	"7"	"6"
D21 High byte				"3"	"2"	"1"	"0"	"7"
D22 Low byte					"3"	"2"	"1"	"0"
D22 High byte						"3"	"2"	"1"
D23 Llow byte							"3"	"2"
D23 High byte								"3"
D24 Low byte								
D24 High byte				no change				
D25 Low byte								
D25 High byte								
D26 Low byte								
D26 High byte								
D27 Low byte								
D27 High byte								

D \ n	K9	K10	K11	K12	K13	K14	K15	K16
D20 Low byte	"B"	"A"	"9"	"8"	"F"	"E"	"D"	"C"
D20 High byte	"4"	"B"	"A"	"9"	"8"	"F"	"E"	"D"
D21 Low byte	"5"	"4"	"B"	"A"	"9"	"8"	"F"	"E"
D21 High byte	"6"	"5"	"4"	"B"	"A"	"9"	"8"	"F"
D22 Low byte	"7"	"6"	"5"	"4"	"B"	"A"	"9"	"8"
D22 High byte	"0"	"7"	"6"	"5"	"4"	"B"	"A"	"9"
D23 Llow byte	"1"	"0"	"7"	"6"	"5"	"4"	"B"	"A"
D23 High byte	"2"	"1"	"0"	"7"	"6"	"5"	"4"	"B"
D24 Low byte	"3"	"2"	"1"	"0"	"7"	"6"	"5"	"4"
D24 High byte		"3"	"2"	"1"	"0"	"7"	"6"	"5"
D25 Low byte			"3"	"2"	"1"	"0"	"7"	"6"
D25 High byte				"3"	"2"	"1"	"0"	"7"
D26 Low byte					"3"	"2"	"1"	"0"
D26 High byte			no change			"3"	"2"	"1"
D27 Low byte							"3"	"2"
D27 High byte								"3"

Program Example 2:

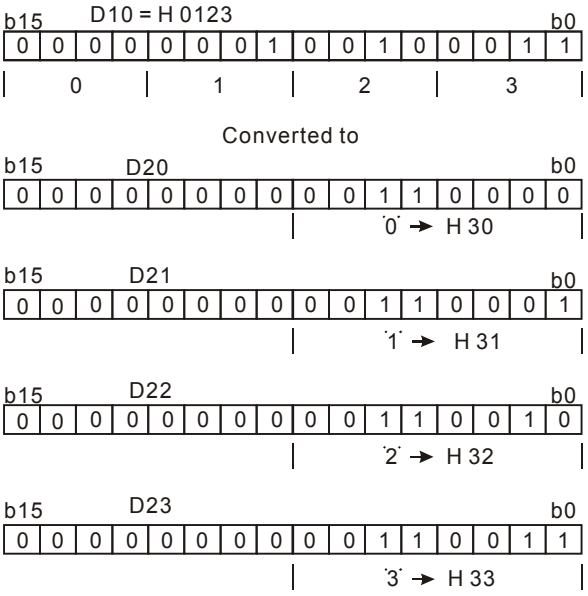
1. M1161 = On: The 8-bit conversion mode
2. When X0 = On, convert the 4 hex values in D10 into ASCII codes and send the result to registers starting from D20.

3. Assume

(D10) = 0123 H	'0' = 30H	'4' = 34H	'8' = 38H
(D11) = 4567 H	'1' = 31H	'5' = 35H	'9' = 39H
(D12) = 89AB H	'2' = 32H	'6' = 36H	'A' = 41H
(D13) = CDEFH	'3' = 33H	'7' = 37H	'B' = 42H

4. When **n** = 2, the bit structure will be as:

5. When **n** = 4, the bit structure will be as:



6. When **n** = 1 ~ 16:

D \ n	K1	K2	K3	K4	K5	K6	K7	K8
D20	“3”	“2”	“1”	“0”	“7”	“6”	“5”	“4”
D21		“3”	“2”	“1”	“0”	“7”	“6”	“5”
D22			“3”	“2”	“1”	“0”	“7”	“6”
D23				“3”	“2”	“1”	“0”	“7”
D24					“3”	“2”	“1”	“0”
D25						“3”	“2”	“1”
D26							“3”	“2”
D27								“3”
D28								
D29				no change				
D30								
D31								
D32								
D33								
D34								
D35								

D \ n	K9	K10	K11	K12	K13	K14	K15	K16
D20	“B”	“A”	“9”	“8”	“F”	“E”	“D”	“C”
D21	“4”	“B”	“A”	“9”	“8”	“F”	“E”	“D”
D22	“5”	“4”	“B”	“A”	“9”	“8”	“F”	“E”
D23	“6”	“5”	“4”	“B”	“A”	“9”	“8”	“F”
D24	“7”	“6”	“5”	“4”	“B”	“A”	“9”	“8”
D25	“0”	“7”	“6”	“5”	“4”	“B”	“A”	“9”
D26	“1”	“0”	“7”	“6”	“5”	“4”	“B”	“A”
D27	“2”	“1”	“0”	“7”	“6”	“5”	“4”	“B”
D28	“3”	“2”	“1”	“0”	“7”	“6”	“5”	“4”
D29		“3”	“2”	“1”	“0”	“7”	“6”	“5”
D30			“3”	“2”	“1”	“0”	“7”	“6”
D31				“3”	“2”	“1”	“0”	“7”
D32					“3”	“2”	“1”	“0”
D33			no change			“3”	“2”	“1”
D34							“3”	“2”
D35								“3”

API	Mnemonic		Operands	Function	Controllers
83	HEX	P	S D n	Converts ASCII to Hex	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*			HEX, HEXP: 7 steps
D								*	*	*	*	*	*			
n					*	*										

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Start device for source data **D**: Start device for storing the converted result **n**: Number of bits to be converted

Explanations:

1. Range of **n**: 1 ~ 256
2. See the specifications of each model for their range of use.
3. Flag: M1161 (8/16 bit mode switch)
4. 16-bit conversion mode: When M1161 = Off, the instruction is in 16-bit conversion mode. ASCII codes of the 8 high bits and 8 low bits of the hex data in **S** are converted into hex value and sent to **D** (every 4 bits as a group). **n** = the number of bits converted into ASCII codes.
5. 8-bit conversion mode: When M1161 = On, the instruction is in 8-bit conversion mode. Every bit of the hex data in **S** are converted into ASCII codes and sent to the 8 low bits of **D**. **n** = the number of converted bits. (All 8 high bits of **D** = 0)

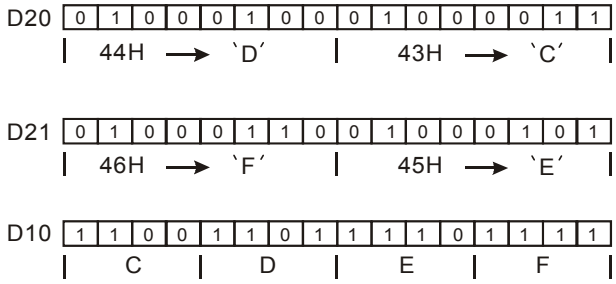
Program Example 1:

1. M1161 = Off: The 16-bit conversion mode
2. When X0 = On, convert the ASCII codes stored in the registers starting from D20 into hex value and send the result (every 4 bits as a group) to registers starting from D10. **n** = 4.

3. Assume

S	ASCII code	Converted to hex	S	ASCII code	Converted to hex
D20 low byte	H 43	“C”	D24 low byte	H 34	“4”
D20 high byte	H 44	“D”	D24 high byte	H 35	“5”
D21 low byte	H 45	“E”	D25 low byte	H 36	“6”
D21 high byte	H 46	“F”	D25 high byte	H 37	“7”
D22 low byte	H 38	“8”	D26 low byte	H 30	“0”
D22 high byte	H 39	“9”	D26 high byte	H 31	“1”
D23 low byte	H 41	“A”	D27 low byte	H 32	“2”
D23 high byte	H 42	“B”	D27 high byte	H 33	“3”

4. When **n** = 4, the bit structure will be as:



5. When **n** = 1 ~ 16:

n \ D	D13	D12	D11	D10
1	The undesignated parts in the registers in use are all 0.			***C H
2				**CD H
3				*CDE H
4				CDEF H
5			***C H	DEF8 H
6			**CD H	EF89 H
7			*CDE H	F89A H
8			CDEF H	89AB H
9		***C H	DEF8 H	9AB4 H
10		**CD H	EF89 H	AB45 H
11		*CDE H	F89A H	B456 H
12		CDEF H	89AB H	4567 H
13	***C H	DEF8 H	9AB4 H	5670 H
14	**CD H	EF89 H	AB45 H	6701 H
15	*CDE H	F89A H	B456 H	7012 H
16	CDEF H	89AB H	4567 H	0123 H

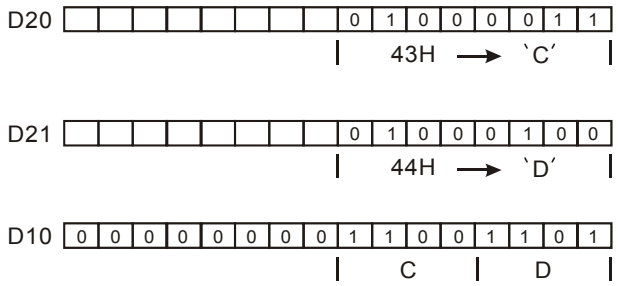
Program Example 2:

1. M1161 = On: The 8-bit converstion mode

2. Assume

S	ASCII code	Converted to hex	S	ASCII code	Converted to hex
D20	H 43	“C”	D28	H 34	“4”
D21	H 44	“D”	D29	H 35	“5”
D22	H 45	“E”	D30	H 36	“6”
D23	H 46	“F”	D31	H 37	“7”
D24	H 38	“8”	D32	H 30	“0”
D25	H 39	“9”	D33	H 31	“1”
D26	H 41	“A”	D34	H 32	“2”
D27	H 42	“B”	D35	H 33	“3”

3. When **n** = 2, the bit structure will be as:



4. When **n** = 1 ~ 16:

n \ D	D13	D12	D11	D10
1	The used registers which are not specified are all 0			***C H
2				**CD H
3				*CDE H
4				CDEF H
5			***C H	DEF8 H
6			**CD H	EF89 H
7			*CDE H	F89A H
8			CDEF H	89AB H
9		***C H	DEF8 H	9AB4 H
10		**CD H	EF89 H	AB45 H
11		*CDE H	F89A H	B456 H
12		CDEF H	89AB H	4567 H
13	***C H	DEF8 H	9AB4 H	5670 H
14	**CD H	EF89 H	AB45 H	6701 H
15	*CDE H	F89A H	B456 H	7012 H
16	CDEF H	89AB H	4567 H	0123 H

API	Mnemonic		Operands	Function	Controllers		
84	CCD	P	S D n	Check Code	ES/EX/SS	SA/SX/SC	EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			CCD, CCDP: 7 steps
D									*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

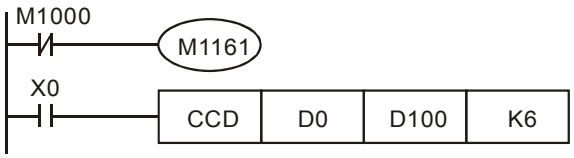
S: Start device for source data **D**: Device for storing the sum check result **n**: Number of data

Explanations:

1. Range of **n**: 1 ~ 256
2. See the specifications of each model for their range of use.
3. Flag: M1161 (8/16 bit mode switch)
4. The sum check is used for ensuring the correctness of the data transmission.
5. 16-bit conversion mode: When M1161 = Off, the instruction is in 16-bit conversion mode. The instruction sums up **n** data (8 bits as a unit) from the start register designated in **S** and stores the results in the registers designated in **D**. The parity bits are stored in **D** + 1.
6. 8-bit conversion mode: When M1161 = On, the instruction is in 8-bit conversion mode. The instruction sums up **n** data (8 bits as a unit; only 8 low bits are valid) from the start register designated in **S** and stores the results in the registers designated in **D**. The parity bits are stored in **D** + 1.

Program Example 1:

1. M1161 = Off: The 16-bit conversion mode
2. When X0 = On, the instruction will sum up 6 data stored in the register designated in D0 (8 bits as a unit; **n** = 6 indicates D0 ~ D2 are designated) and store the result in the register designated in D100. The parity bits are stored in D101.



(S)	Content of data
D0 low byte	K100 = 0 1 1 0 0 1 0 0
D0 high byte	K111 = 0 1 1 0 1 1 1①
D1 low byte	K120 = 0 1 1 1 1 0 0 0
D1 high byte	K202 = 1 1 0 0 1 0 1 0
D2 low byte	K123 = 0 1 1 1 1 0 1①
D2 high byte	K211 = 1 1 0 1 0 0 1①
D100	K867
D101	0 0 0 1 0 0 0①

Total

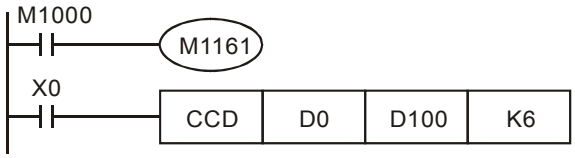
The parity is 1 when there is a odd number of 1.
The parity is 0 when there is a even number of 1.

D100	0	0	0	0	0	0	1	1	0	1	1	0	0	0	1	1
D101	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

⇐ Parity

Program Example 2:

1. M1161 = On: The 8-bit conversion mode
2. When X0 = On, the instruction will sum up 6 data stored in the register designated in D0 (8 bits as a unit; **n** = 6 indicates D0 ~ D5 are designated) and store the result in the register designated in D100. The parity bits are stored in D101.



(S)	Content of data
D0 low byte	K100 = 0 1 1 0 0 1 0 0
D1 low byte	K111 = 0 1 1 0 1 1 1①
D2 low byte	K120 = 0 1 1 1 1 0 0 0
D3 low byte	K202 = 1 1 0 0 1 0 1 0
D4 low byte	K123 = 0 1 1 1 1 0 1①
D5 low byte	K211 = 1 1 0 1 0 0 1①
D100	K867
D101	0 0 0 1 0 0 0①

Total

The parity is 1 when there is a odd number of 1.
The parity is 0 when there is a even number of 1.

D100	0	0	0	0	0	0	1	1	0	1	1	0	0	0	1	1
D101	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

⇐ Parity

API	Mnemonic		Operands	Function	Controllers
85	VRRD	P	(S) (D)	Volume Read	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	VRRD, VRRDP: 5 steps
S					*	*										
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

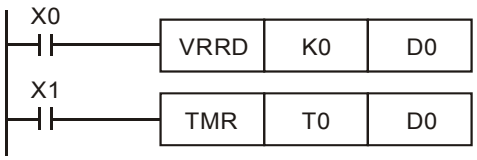
S: No. of VR **D**: Device for storing the volume of VR

Explanations:

1. Range of **S**: 0 ~ 7; without function card: 0 ~ 1.
2. See the specifications of each model for their range of use.
3. Flags: M1178, M1179. See remarks for more details.
4. VRRD instruction is used for reading 2 points (No.0, No.1) of PLC or the VR rotary switch volume change in the 6 points of the function cards (No.2 ~ No.7) and converting the data into values 0 ~ 255 (stored in **D**).
5. If you are to set up the timer by the VR volume, simply rotate the VR to modify the set time in the timer. If you are to acquire a value larger than 255, multiply **D** by a constant.

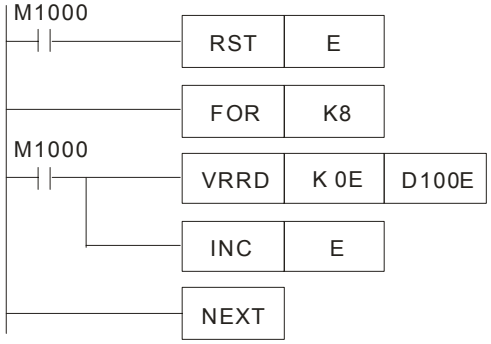
Program Example 1:

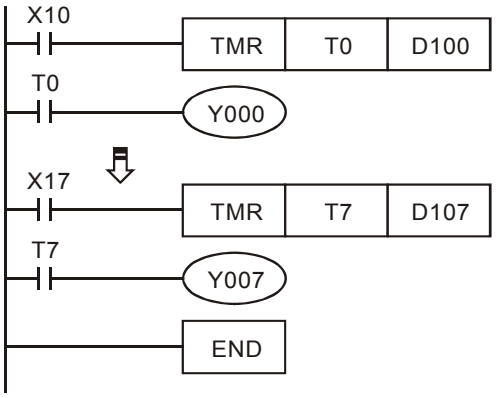
1. When X0 = On, VR0 volume changed will be converted into an 8-bit BIN value (0 ~ 255) and stored in D0.
2. When X1 = On, the imer T0 will start to time with the content in D0 as the set value in the timer.



Program Example 2:

1. Read the VR volume in order: The VR0 ~ VR7 rotary switches on the PLC correspond to **S** = K0 ~ K7 of VRRD instruction. E index register modification is used in the example below, K0E = K0 ~ K7.
2. The timer converts the scale 0 ~ 10 on the rotary switch into 0 ~ 255. The timing unit of T0 ~ T7 is 0.1 second; therefore, the set time in the timer will be 0 ~ 25.5 seconds.





3. Operation of FOR ~ NEXT instruction:
   a) In the area between FOR ~ NEXT instruction, FOR designating K8 indicates the loop between FOR ~ NEXT will be executed repeatedly for 8 times before the next instruction is executed.
   b) Between FOR ~ NEXT (INC E), E will be 0, 1, 2, …7 plusing 1. Therefore, the 8 VR rotary switch volumes will be VR0→D100, VR1→D101, VR2→D102…VR7→D107 and be read to designated registers in order.

Remarks:

1. VR refers to **V**ariable **R**esister.
2. The 2 points of VR rotary switch built in SA/SC/EH/EH2/SV series MPU can be used together with special D and special M.

Device	Function
M1178	Enabling VR0
M1179	Enabling VR1
D1178	VR0 value
D1179	VR1 value

3. If there is no VR extension card inserted in the PLC, setting up the No. of rotary switches as K2 ~ K7 in VRRD and VRSC instruction in the program will result in errors in grammar check.

API	Mnemonic		Operands	Function	Controllers
86	VRSC	P	(S) (D)	Volume Scale	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*										VRSC, VRSCP: 5 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: No. of VR **D**: Device for storing the scale of VR

Explanations:

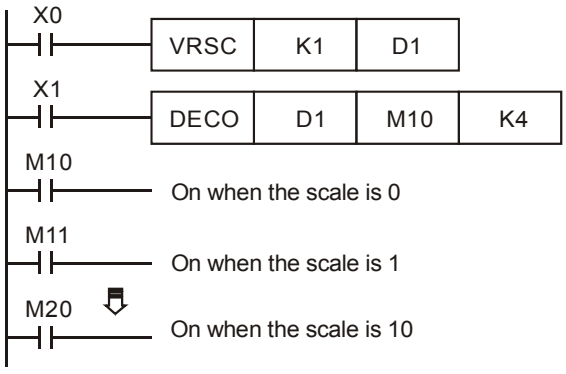
1. Range of **S**: 0 ~ 7; without function card: 0 ~ 1
2. See the specifications of each model for their range of use.
3. VRSC instruction is used for reading 2 points (No.0, No.1) of PLC or the VR rotary switch scale (0 ~ 10) in the 6 points of the function cards (No.2 ~ No.7) and storing the data in **D**. If the position of the VR falls in the middle of two scales, VRSC will round up the value into an integer of 0 ~ 10.

Program Example 1:

When X0 = On, the scale of VR0 (0 ~ 10) will be stored in D10.

Program Example 2:

1. When the VR is used as DIP switch, they will correspond to scale 0 ~ 10 and only one of M10 ~ M20 will be On. Use API 41 DECO instruction to decode the scales into M10 ~ M25.
2. When X0 = On, store the scale (0 ~ 10) of VR1 into D1.
3. When X1 = On, use API 41 DECO to decode the scales into M10 ~ M25.



Remarks:

If the MPU is not inserted with a VR extension card, and the No. of the rotary switches inVRRD or VRSC instruction in the program are set as K2 ~ K7, errors will occur in the execution of grammar check.

API	Mnemonic			Operands	Function	Controllers
87	D	ABS	P	(D)	Absolute Value	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	ABS, ABSP: 3 steps DABS, DABSP: 5 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

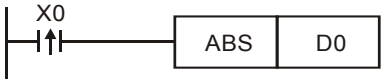
D: Device of the absolute value

Explanations:

1. See the specifications of each model for their range of use.
2. This instruction obtains the absolute value of the content in the designated in **D**.
3. This instruction adopts pulse execution instructions (ABSP, DABSP).

Program Example:

When X0 = Off→On, obtain the absolute value of the content in D0.



API	Mnemonic	Operands	Function	Controllers
88	D PID	S_1 S_2 S_3 D	PID Control Loop	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1													*			PID : 9 steps
S_2													*			DPID: 17 steps
S_3													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

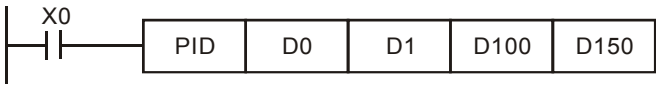
S_1: Set value (SV) **S_2**: Present value (PV) **S_3**: Parameter **D**: Output value (MV)

Explanations:

1. In 16-bit instruction, **S_3** will occupy 20 consecutive devices; in 32-bit instruction, **S_3** will occupy 21 consecutive devices.
2. See the specifications of each model for their range of use.
3. See the Remarks below for the times of using PID instruction allowed in the program.
4. This instruction is specifically for PID control. PID operation will be executed by the scan only when the sampling time is reached. PID refers to “proportion, integration and differential”. PID control is widely applied to many machines, pneumatic and electronic equipments.
5. For the 16-bit instruction, the parameters are **S_3** ~ **S_3**+19; for the 32-bit instruction, the parameters are **S_3** ~ **S_3**+20. After all the parameters are set up, PID instruction will start to be executed and the results will be stored in **D**. **D** has to be the data register area without latched function. (If you wish to designate a latched data register area, place the data register in the latched area at the beginning of the program and clear it as 0.)

Program Example:

1. Complete the parameter setting before executing PID instruction.
2. When X0 = On, the instruction will be executed and the result will be stored in D150. When X0 goes Off, the instruction will not be executed and the data prior to the instruction will stay intact.



Remarks:

1. ES/EX/SS series MPU V5.7 (and above) supports PID instruction. Other versions do not support the instruction.
2. There is no limitation on the times of using this instruction. However, the register No. designated in **S_3** cannot be repeated.
3. For the 16-bit instruction, **S_3** will occupy 20 registers. In the program example above, the area designated in **S_3** is D100 ~ D119. Before the execution of PID instruction, you have to transmit the setting value to the designated register area by MOV instruction, If the designated registers are latched, use MOVP instruction to transmit all setting value at a time.
4. Settings of **S_3** in the 16-bit instruction

Device No.	Function	Setup Range	Explanation
S_3:	Sampling time (T_S) (unit: 10ms)	1 ~ 2,000 (unit: 10ms)	If T_S is less than 1 program scan time, PID instruction will be executed for 1 program scan time. If T_S= 0, PID instruction will not be enabled. The minimum T_S has to be longer than the program scan time.
S_3 +1:	Proportional gain (K_P)	0 ~ 30,000 (%)	The magnified error proportional value between SV – PV.
S_3 +2:	Integral gain (K_I)	0 ~ 30,000 (%)	The magnified proportional value of every sampling time unit × the accumulated value of the error.
S_3 +3:	Differential gain (K_D)	-30,000 ~ 30,000 (%)	The magnified proportional value of the varied error in every samping timme unit.
S_3 +4:	Control direction (DIR)	0: automatic control 1: forward control (E = SV - PV) 2: inverse control (E = PV - SV) 3: Auto-tuning of parameter exclusively for the temperature control. The device will automatically become K4 when the auto-tuning is completed and be filled in with the appropriate parameter K_P, K_I and K_D (not avaliable in the 32-bit instruction). 4: Exclusively for the adjusted temperature control (not avaliable in the 32-bit instruction).	
S_3 +5:	The range that error value (E) doesn't work	0 ~ 32,767	E = the error of SV – PV. When S_3 +5 = K0, the function will not be enabled, e.g. when S_3 +5 is set as 5, MV of E between -5 and 5 will be 0.
S_3 +6:	Upper bound of output value (MV)	-32,768 ~ 32,767	Ex: if S_3 +6 is set as 1,000, the output will be 1,000 when MV is bigger than 1,000. S_3 +6 has to be bigger or equal S_3 +7; otherwise the upper bound and lower bound will switch.
S_3 +7:	Lower bound of output value (MV)	-32,768 ~ 32,767	Ex: if S_3 +7 is set as -1,000, the output will be -1,000 when MV is smaller than -1,000.
S_3 +8:	Upper bound of integral value	-32,768~32,767	Ex: if S_3 +8 is set as 1,000, the output will be 1,000 when the integral value is bigger than 1,000 and the integration will stop. S_3 +8 has to be bigger or equal S_3 +9; otherwier the upper bound and lower bound will switch.
S_3 +9:	Lower bound of integral value	-32,768 ~ 32,767	Ex: if S_3 +9 is set as -1,000, the output will be -1,000 when the integral value is smaller than -1,000 and the integration will stop.
S_3 +10,11:	Accumulated integral value	32-bit floating point	The accumulated integral value is only for reference. You can still clear or modify it (in 32-bit floating point) according to your need.
S_3 +12:	The previous PV	-	The previous PV is only for reference. You can still modify it according to your need.
S_3 +13: ≀ S_3 +19:	For system use only.		

5. When parameter setting exceeds its range, the upper bound and lower bound will become the setting value. However, if the motion direction (DIR) exceeds the range, it will be set to 0.
6. PID instruction can be used in interruption subroutines, step points and CJ instruction.
7. The maximum error of sampling time T_S = - (1 scan time + 1ms) ~ + (1 scan time). When the error affects the output, please fix the scan time or execute PID instruction in the interruption subroutine of the timer.
8. PV of PID instruction has to be stable before the execution of PID instruction. If you are to extract the input value

of DVP04AD/04DA/06XA/04PT/04TC for PID operation, please be aware of the A/D conversion time of these modules.

9. For the 32-bit instruction, If **S_3** designates the parameter setting area of PID instruction as D100 ~ D120, **S_3** occupies 21 registers. Before the execution of PID instruction, you have to use MOV instrction first to send the setting value to the register area for setup. If the designated registers are latched one, use MOVP instruction to send all the setting value at a time.
10. Settings of **S_3** in the 32-bit instruction

Device No.	Function	Setup range	Explanation
S_3 :	Sampling time (T_S) (unit: 10ms)	1 ~ 2,000 (unit: 10ms)	If T_S is less than 1 program scan time, PID instruction will be executed for 1 program scan time. If T_S= 0, PID instruction will not be enabled. The minimum T_S has to be longer than the program scan time.
S_3 +1:	Proportional gain (K_P)	0 ~ 30,000 (%)	The magnified error proportional value between SV – PV.
S_3 +2:	Integral gain (K_I)	0 ~ 30,000 (%)	The magnified proportional value of every sampling time unit × the accumulated value of the error.
S_3 +3:	Differential gain (K_D)	-30,000 ~ 30,000 (%)	The magnified proportional value of the varied error in every samping timme unit.
S_3 +4:	Control direction (DIR)	0: automatic control 1: forward control (E = SV – PV) 2: inverse control (E = PV – SV)	
S_3 +5, 6:	The range that 32-bit error value (E) doesn't work	0 ~ 2,147,483,647	E = the error of SV – PV. When **S_3** +5,6 = K0, the function will not be enabled, e.g. when **S_3** +5,6 is set as 5, MV of E between -5 and 5 will be 0.
S_3 +7, 8:	Upper bound of 32-bit output value (MV)	-2,147,483,648 ~ 2,147,483,647	Ex: if **S_3** +7,8 is set as 1,000, the output will be 1,000 when MV is bigger than 1,000. **S_3** +7,8 has to be bigger or equal **S_3** +9,10; otherwise the upper bound and lower bound will switch.
S_3 +9, 10:	Lower bound of 32-bit output value (MV)	-2,147,483,648 ~ 2,147,483,647	Ex: if **S_3** +9,10 is set as -1,000, the output will be -1,000 when MV is smaller than -1,000.
S_3 +11, 12:	Upper bound of 32-bit integral value	-2,147,483,648 ~ 2,147,483,647	Ex: if **S_3** +11,12 is set as 1,000, the output will be 1,000 when the integral value is bigger than 1,000 and the integration will stop. **S_3** +11,12 has to be bigger or equal **S_3** +13,14; otherwier the upper bound and lower bound will switch.
S_3 +13, 14:	Lower bound of 32-bit integral value	-2,147,483,648 ~ 2,147,483,647	Ex: if **S_3** +13,14 is set as -1,000, the output will be -1,000 when the integral value is smaller than -1,000 and the integration will stop.
S_3 +15, 16:	32-bit accumulated integral value	32-bit floating point	The accumulated integral value is only for reference. You can still clear or modify it (in 32-bit floating point) according to your need.
S_3 +17, 18:	32-bit previous PV	-	The previous PV is only for reference. You can still modify it according to your need.
S_3 +19: ≀ **S_3** +20:	For system use only.		

11. The explanation of 32-bit **S₃** and 16-bit **S₃** are almost the same. The difference is the capacity of **S₃**+5 ~ **S₃**+20.

PID Equations:

1. The PID operation is conducted according to the speed and the differential PV.
2. The PID operation has three control directions: automatic, foreward and inverse. Forward or inverse are designated in **S₃** +4. Other relevant settings of PID operation are set by the registers designated in **S₃** ~ **S₃** + 5.
3. Basic PID equation:

$$MV = K_P * E(t) + K_I * E(t)\frac{1}{S} + K_D * PV(t)S$$

Control direction	PID equation
Forward, automatic	E(t) = SV – PV
Inverse	E(t) = PV – SV

$PV(t)S$ is the differential value of $PV(t)$; $E(t)\frac{1}{S}$ is the integral value of $E(t)$. When E(t) is less than 0 as the control direction is selected as forward or inverse, E(t) will be regarded as “0".

The equation above illustrates that this instruction is different from a general PID instruction by the variable use of the differential value. To avoid the flaw that the transient differential value is too big when a general PID instruction is executed for the first time, our PID instruction monitors the differentiation status of the PV. When the variation of PV is too big, this instruction will reduce the output of MV.

4. Symbol explanation:

MV : Output value

K_P : Proprotional gain

$E(t)$: Error value

PV : Present measured value

SV : Target value

K_D : Differential gain

$PV(t)S$: Differential value of PV(t)

K_I : Integral gain

$E(t)\frac{1}{S}$: Integral value of E(t)

5. Temperature Control Equation:

When **S₃** +4 is K3 and K4, the equation used in diagram 2 (see below) will be changed as:

$$MV = \frac{1}{K_P}\left[E(t) + \frac{1}{K_I}\left(E(t)\frac{1}{S}\right) + K_D * PV(t)S\right]$$

In which the error value is fixed as E(t) = SV – PV

This equation is exclusively designed for temperature control. Therefore, when the sampling time (T_S) is set as 4 seconds (K400), the range of output value (MV) will be K0 ~ K4,000 and the cycle time of GPWM instruction used together has to be set as 4 seconds (K4000) as well.

If you have no idea how to adjust the parameters, you can select K3 (auto-tuning) and after all the parameters are adjusted (the control direction will be automatically set as K4), you can modify your parameters to better ones according to the result of the control.

6. Control diagrams:

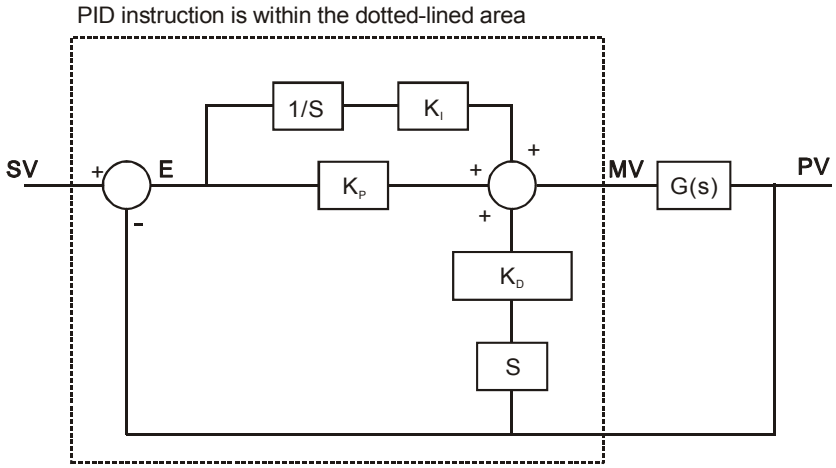


Diagram 1:$\mathbf{S}_3$ + 4 = K0 ~ K2

In Diagram 1, S is differentiation, referring to “PV – previous PV / sampling time”. 1 / S is integration, referring to “(previous integral value + error value) × sampling time”. G(S) refers to the device being controlled.

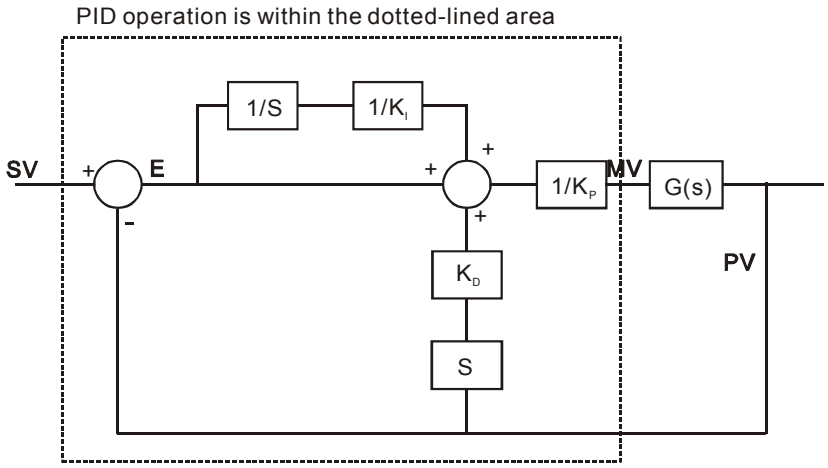


Diagram 2: $\mathbf{S}_3$ + 4 = K3 ~ K4

In Diagram 2, $1/K_I$ and $1/K_P$ refer to “divided by K_I” and “divided by K_P”. Due to that this is exclusively for temperature control, you have to use PID instruction together with GPWM instruction. See Application 3 for more details.

7. Notes:

a) $\mathbf{S}_3$ + 6 ~ $\mathbf{S}_3$ + 13 are only available in SA/SX/SC/EH/EH2/SV series, and ES/EX/SS (v5.7 and above) series MPU.

b) PID instruction can only be used once in ES/EX/SS (v5.6 and below) series MPU. There is no limitation on the times of using PID instruction in ES/EX/SS (v5.7 and above) series and SA/SX/SC/EH/EH2/SV series MPU.

c) $\mathbf{S}_3$ + 3 of ES/EX/SS (v5.7 and below), SA/SX/SC (v1.1 and below) and EH (v1.0 and below) series MPU can only be the value within 0 ~ 30,000.

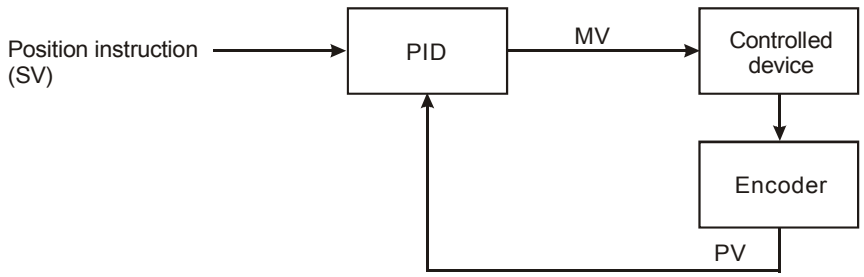
d) There are a lot of circumstances where PID instruction can be applied; therefore, please choose the control functions appropriately. For example, when you select parameter auto-tuning for the temperature ($\mathbf{S}_3$ + 4 = K3), you cannot use it in a motor control environment in case improper control may occur.

e) When you adjust the three main parameters, K_P, K_I and K_D ($\mathbf{S}_3$ + 4 = K0 ~ K2), you have to adjust K_P first (according to your experiences) and set K_I and K_D as 0. When you can roughly handle the control, you then adjust K_I (increasingly) and K_D (increasingly) (see example 4 below for how to adjust). K_P = 100 refers to 100%, i.e. the gain of the error is 1. K_P < 100% will decrease the error and K_P > 100% will increase the error.

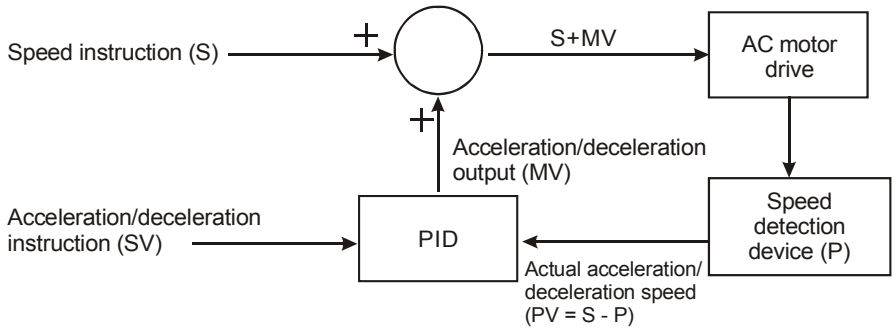
f) When you select the parameter exclusively for temperature control ($\mathbf{S}_3$ + 4 = K3, K4), it is suggested that you store the parameter in D register in the latched area in case the automatically adjusted parameter will disappear after the power is cut off. There is no guarantee that the adjusted parameter is suitable for every control. Therefore, you can modify the adjusted parameter according to your actual need, but it is suggested that you modify only K_I or K_D.

g) PID instruction can to work with many parameters; therefore please do not randomly modify the parameters in case the control cannot be executed normally.

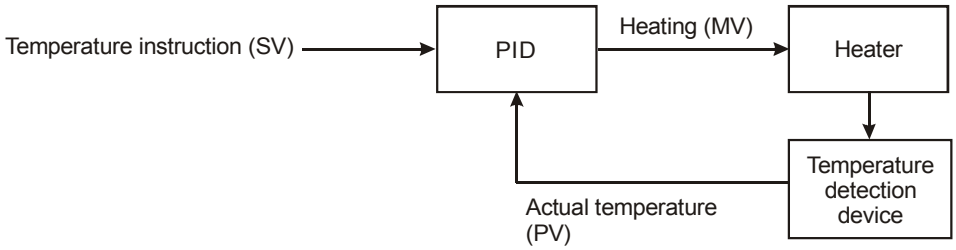
Example 1: Diagram of using PID instruction in position control ($\mathbf{S_3}$ + 4 = 0)



Example 2: Diagram of using PID instruction with AC motor drive on the control ($\mathbf{S_3}$ + 4 = 0)



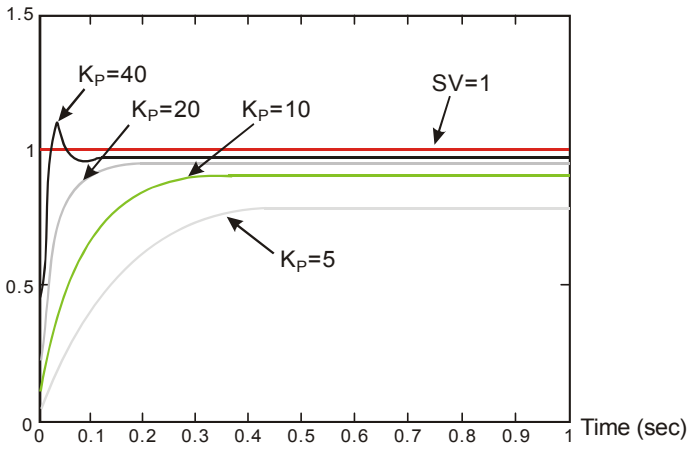
Example 3: Diagram of using PID instruction in temperature control (S_3 + 4 = 1)



Example 4: How to adjust PID parameters

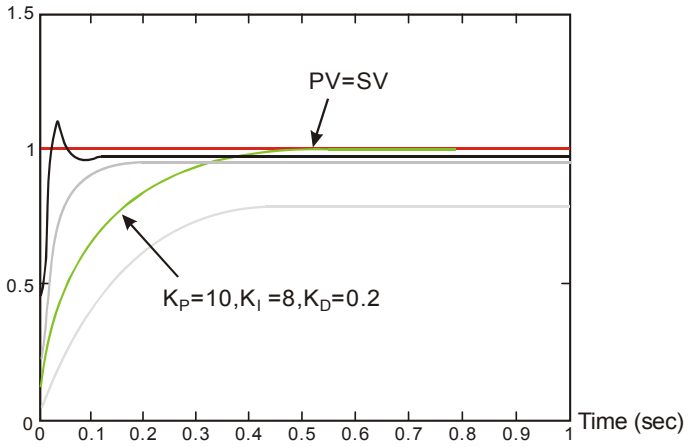
Assume that the transfer function of the controlled device G(S) in a control system is a first-order function $G(s)=\frac{b}{s+a}$ (most models of motors are first-order function), SV = 1, and sampling time (T_S) = 10ms, we suggest you to follow the steps below for adjusting the parameters.

Step 1: Set K_I and K_D as 0 and K_P as 5, 10, 20 and 40. Record the SV and PV respectively and the results are as the figure below.



Step 2: From the figure, we can see that when K_P = 40, there will be over-reaction, so we will not select it. When K_P = 20, the PV reaction curve will be close to SV and there will not be over-reaction, but due to its fast start-up with big transient MV, we will consider to put it aside. When K_P = 10, the PV reaction curve will get close to SV value more smoothly, so we will use it. Finally when K_P = 5, we will not consider it due to the slow reaction.

Step 3: Select K_P = 10 and adjust K_I from small to big (e.g. 1, 2, 4 to 8). K_I should not be bigger than K_P. Adjust K_D from small to big (e.g. 0.01, 0.05, 0.1 and 0.2). K_D should not exceed 10% of K_P. Finally we obtain the figure of PV and SV below.



Note: This example is only for your reference. Please adjust your parameters to proper ones according to your actual condition of the control system.

Application Examples:

Application 1 Using PID instruction in the pressure control system (use the diagram of Example 1).

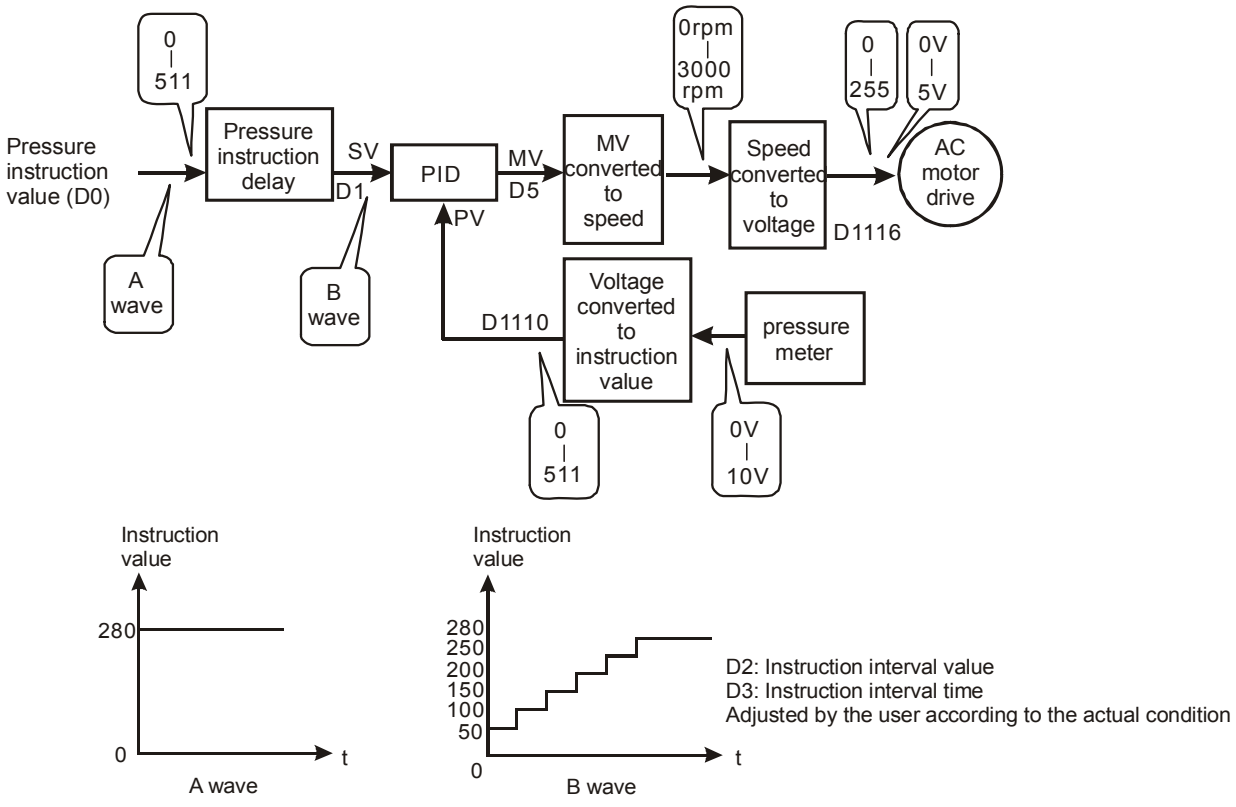
Purpose: Enabling the control system to reach the target pressure.

Explanation: The system requires a gradual control. Therefore, the system will be overloaded or out of control if the process progresses too fast.

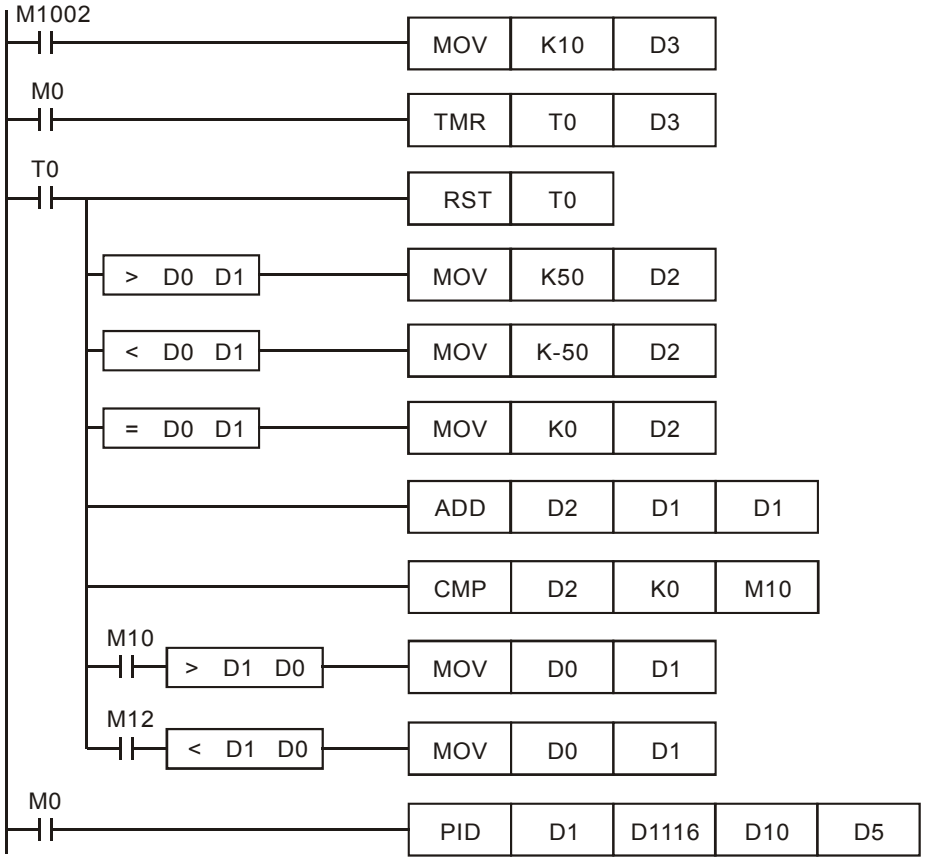
Suggested solution:

Solution 1: Longer sampling time

Solution 2: Using delay instruction. See the figure below.



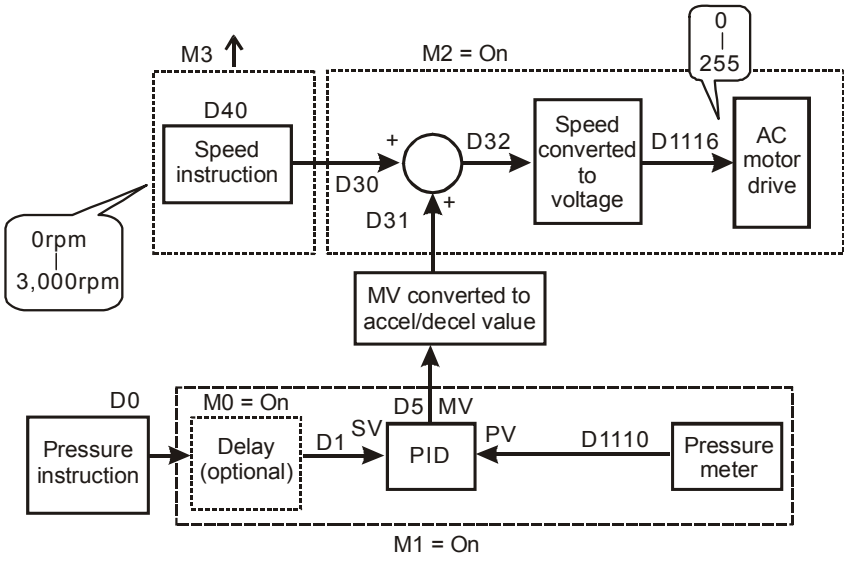
The example program of the instruction delay:



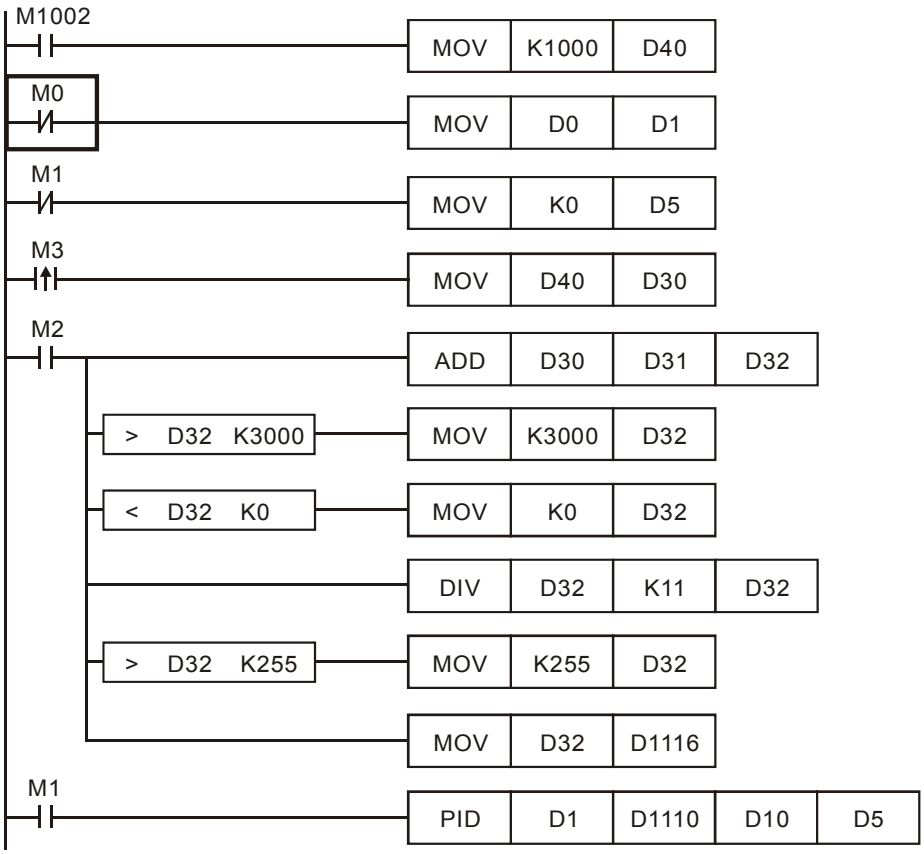
Application 2 Speed control system and pressure control system work individually (use diagram of Example 2).

Purpose: After the speed control operates in open loop for a period of time, adding into it the pressure control system (PID instruction) for close loop control.

Explanation: Since the speed and pressure control systems are not interrelated, we have to structure a open loop for speed control first following by a close loop pressure control. If you fear that the control instruction of the pressure control system changes too fast, you can consider to add the instruction delay illustrated in Application 1 into the control. See the control diagram below.



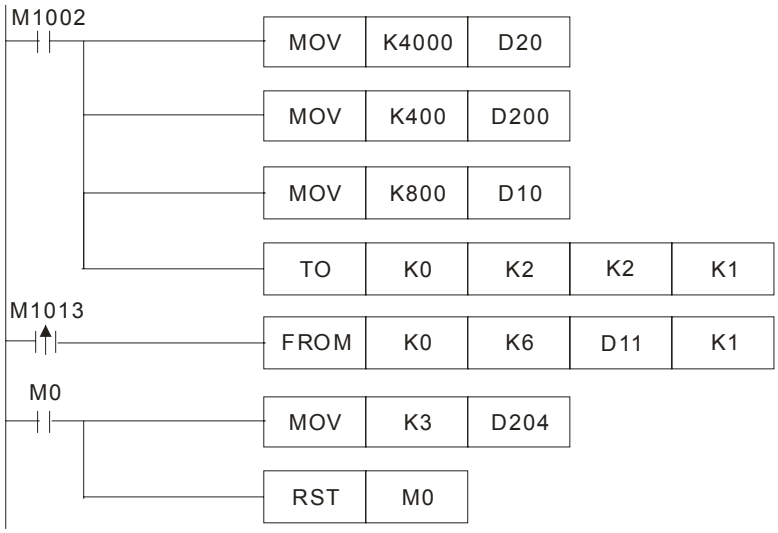
Part of the example program:

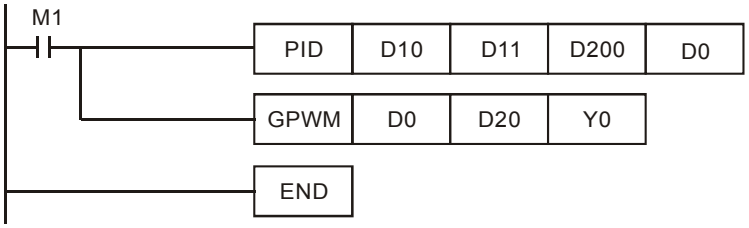


Application 3 Using auto-tuning on the parameter for the temperature control.

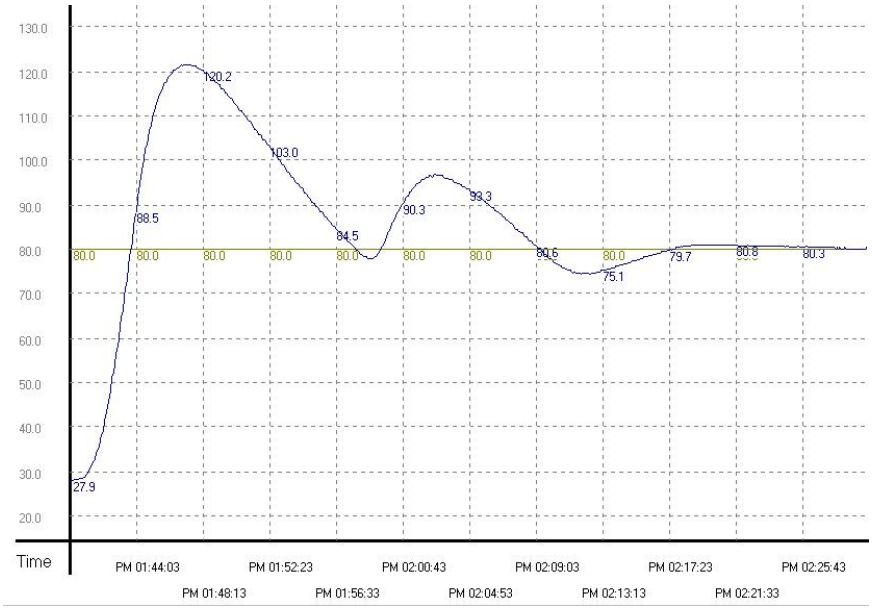
Purpose: Using auto-tuning to calculate the most suitable parameters for PID temperature control.

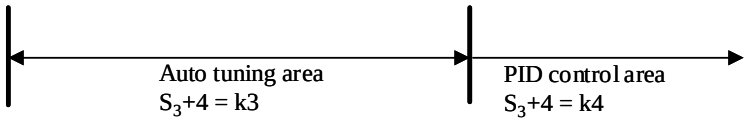
Explanation: You may not be familiar with the temperature environment for the first time, so you can use auto-tuning (S_3 + 4 = K3) for an initial adjustment. After this, PID instruction will become exclusively for temperature control (S_3 + 4 = K4). In this example, the control environment is an oven. See the example program below.



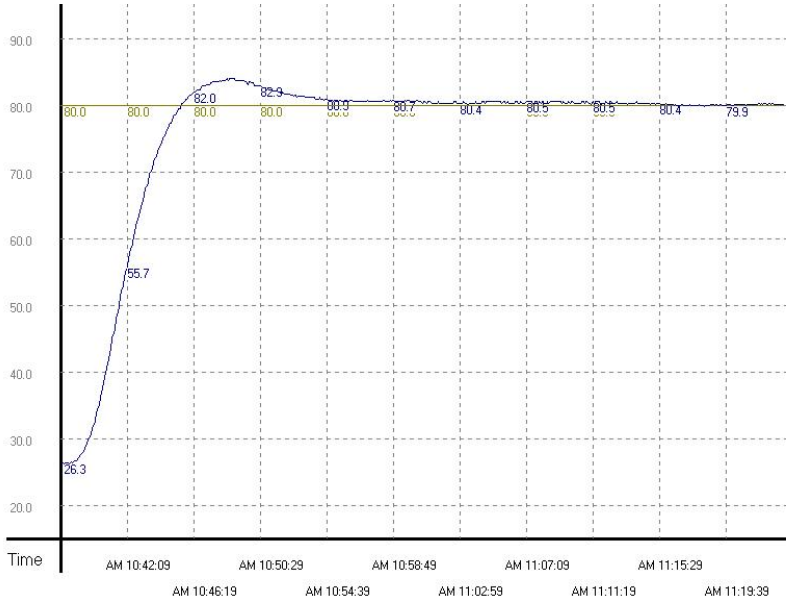


The experiment result of auto-tuning:



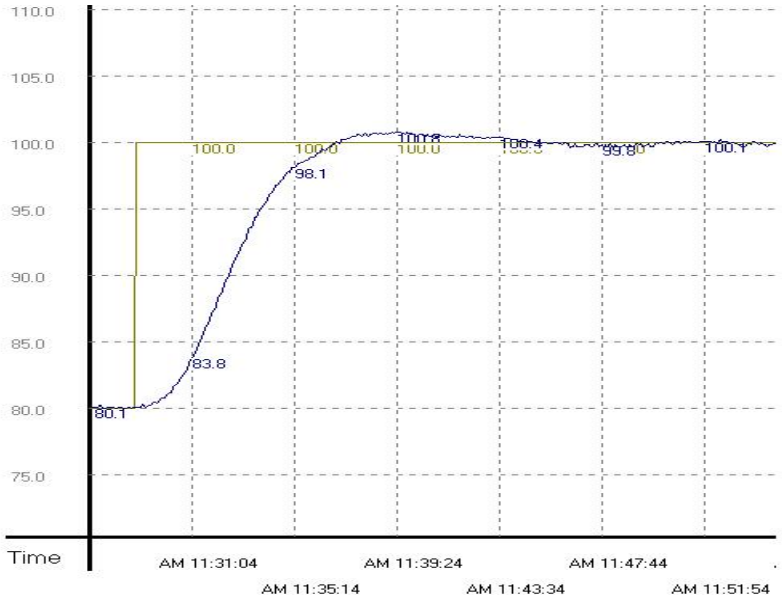


The experiment result of using the adjusted parameter exclusively for temperature control after auto-tuning:



From the figure above, we can see that the temperature control after auto-tuning is working fine and we use only

approximately 20 minutes for the control. Next, we modify the target temperature from 80°C to 100°C and obtain the result below.



From the result above, we can see that when the parameter is 100°C, we can still control the temperature without spending too much time.

API	Mnemonic	Operands	Function	Controllers		
				ES/EX/SS	SA/SX/SC	EH/SV
100	MODRD	(S1) (S2) (n)	Read Modbus Data			

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			MODRD: 7 steps
S_2					*	*							*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

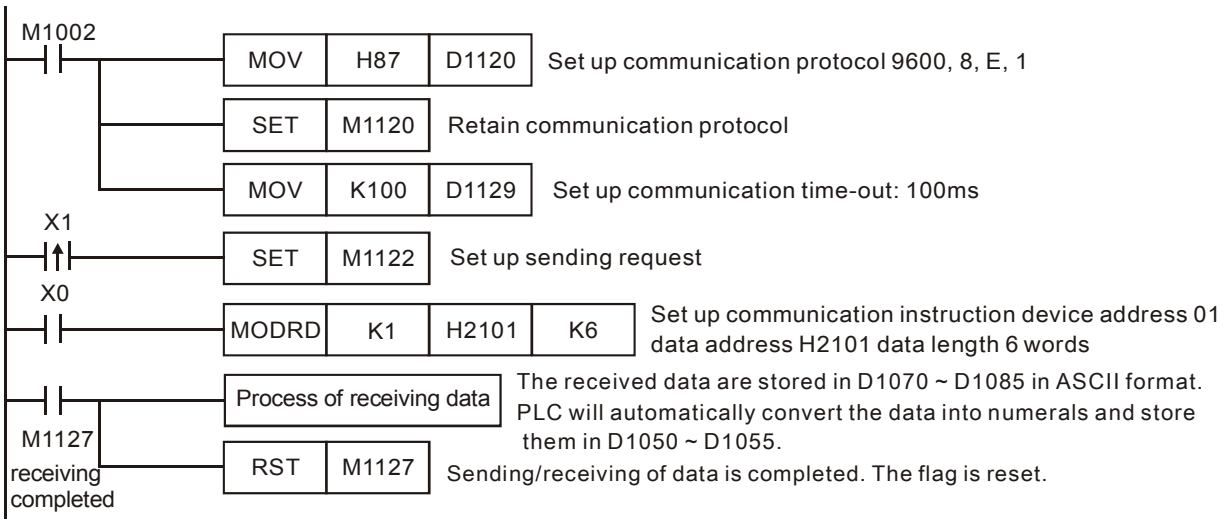
S_1: Address of communication device **S_2**: Address of data to be read **n**: Length of read data

Explanations:

1. Range of **S_1**: K0 ~ K254
2. Range of **n**: K1 ≤ n ≤ K6
3. See the specifications of each model for their range of use.
4. ES/EX/SS series MPU does not support E, F index register modification.
5. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143
6. MODRD is a drive instruction exclusively for peripheral communication equipment in MODBUS ASCII mode /RTU mode. The built-in RS-485 communication ports in Delta VFD drives (except for VFD-A series) are all compatible with MODBUS communication format. MODRD can be used for controlling communication (read data) of Delta drives.
7. If the address of **S_2** is illegal to the designed communication device, the device will respond with an error, PLC will records the error code in D1130 and M1141 will be On.
8. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1085. After receiving the feedback data is completed, PLC will auto-check if all data are correct. If there is an error, M1140 will be On.
9. In ASCII mode, due to that the feedback data are all in ASCII, PLC will convert the feedback data into numerals and store them in D1050 ~ D1055. D1050 ~ D1055 will be invalid in RTU mode.
10. After M1140 or M1141 turn On, the program will send a correct datum to the peripheral equipment. If the feedback datum is correct, M1140 and M1141 will be reset.

Program Example 1:

Communication between PLC and VFD-S series AC motor drives (ASCII Mode, M1143 = Off)



PLC ⇨ VFD-S, PLC sends: "**01 03 2101 0006 D4**"

VFD-S ⇨ PLC , PLC receives: "**01 03 0C 0100 1766 0000 0000 0136 0000 3B**"

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1089 low	'0'	30 H	ADR 1 ADR 0	Address of AC motor drive: ADR (1,0)
D1089 high	'1'	31 H		
D1090 low	'0'	30 H	CMD 1 CMD 0	Instruction code: CMD (1,0)
D1090 high	'3'	33 H		
D1091 low	'2'	32 H	Starting data address	
D1091 high	'1'	31 H		
D1092 low	'0'	30 H		
D1092 high	'1'	31 H		
D1093 low	'0'	30 H	Number of data (counted by words)	
D1093 high	'0'	30 H		
D1094 low	'0'	30 H		
D1094 high	'6'	36 H		
D1095 low	'D'	44 H	LRC CHK 1 LRC CHK 0	Checksum: LRC CHK (0,1)
D1095 high	'4'	34 H		

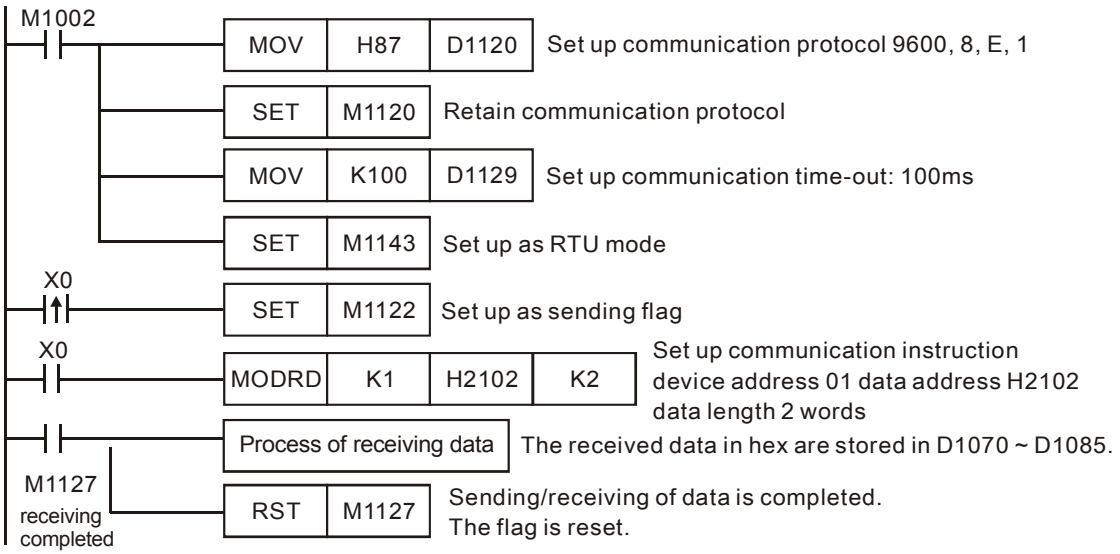
Registers for received data (responding messages)

Register	DATA		Explanation	
D1070 low	'0'	30 H	ADR 1 ADR 0	
D1070 high	'1'	31 H		
D1071 low	'0'	30 H	CMD 1 CMD 0	
D1071 high	'3'	33 H		
D1072 low	'0'	30 H	Number of data (counted by byte)	
D1072 high	'C'	43 H		
D1073 low	'0'	30 H	Content of address 2101 H	PLC automatically convert ASCII codes to numerals and store the numeral in D1050 = 0100 H
D1073 high	'1'	31 H		
D1074 low	'0'	30 H		
D1074 high	'0'	30 H		
D1075 low	'1'	31 H	Content of address 2102 H	PLC automatically convert ASCII codes to numerals and store the numeral in D1051 = 1766 H
D1075 high	'7'	37 H		
D1076 low	'6'	36 H		
D1076 high	'6'	36 H		
D1077 low	'0'	30 H	Content of address 2103 H	PLC automatically convert ASCII codes to numerals and store the numeral in D1052 = 0000 H
D1077 high	'0'	30 H		
D1078 low	'0'	30 H		
D1078 high	'0'	30 H		

Register	DATA		Explanation	
D1079 low	'0'	30 H	Content of address 2104 H	PLC automatically convert ASCII codes to numerals and store the numeral in D1053 = 0000 H
D1079 high	'0'	30 H		
D1080 low	'0'	30 H		
D1080 high	'0'	30 H		
D1081 low	'0'	30 H	Content of address 2105 H	PLC automatically convert ASCII codes to numerals and store the numeral in D1054 = 0136 H
D1081 high	'1'	31 H		
D1082 low	'3'	33 H		
D1082 high	'6'	36 H		
D1083 low	'0'	30 H	Content of address 2106 H	PLC automatically convert ASCII codes to numerals and store the numeral in D1055 = 0000 H
D1083 high	'0'	30 H		
D1084 low	'0'	30 H		
D1084 high	'0'	30 H		
D1085 low	'3'	33 H	LRC CHK 1	
D1085 high	'B'	42 H	LRC CHK 0	

Program Example 2:

Communication between PLC and VFD-S series AC motor drives (RTU Mode, M1143 = On)



PLC ⇨ VFD-S, PLC sends: **01 03 2102 0002 6F F7**

VFD-S ⇨ PLC, PLC receives: **01 03 04 1770 0000 FE 5C**

Registers for sent data (sending messages)

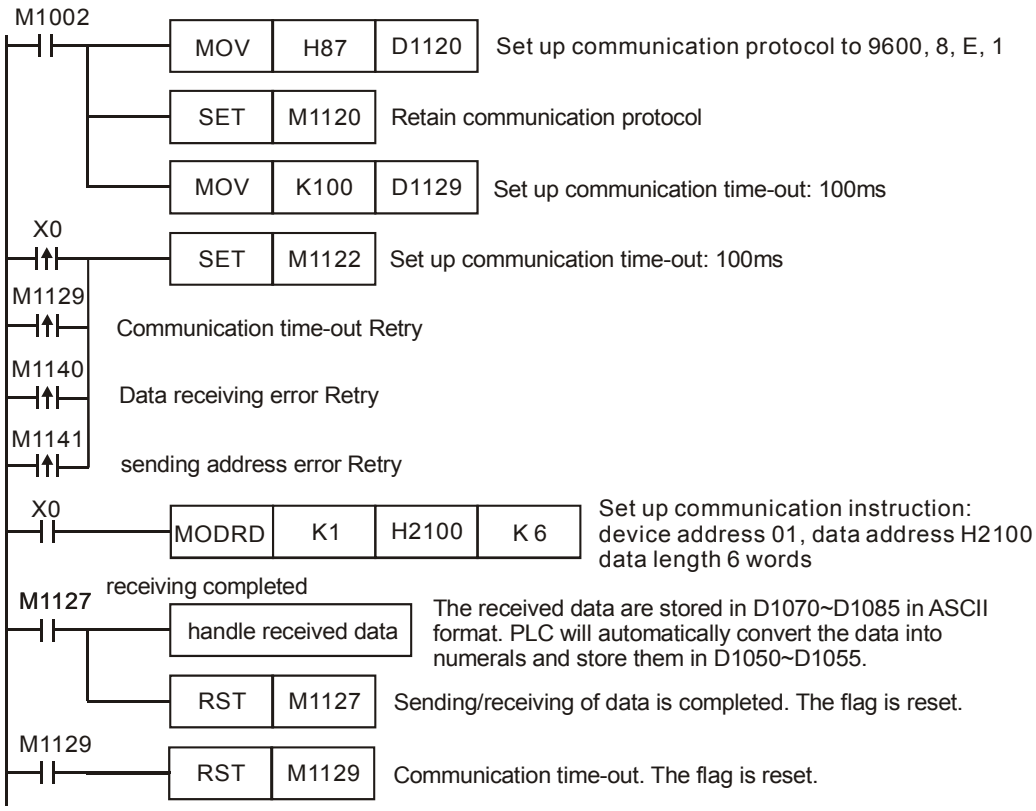
Register	DATA	Explanation
D1089 low	01 H	Address
D1090 low	03 H	Function
D1091 low	21 H	Starting data address
D1092 low	02 H	
D1093 low	00 H	Number of data (counted by words)
D1094 low	02 H	
D1095 low	6F H	CRC CHK Low
D1096 low	F7 H	CRC CHK High

Registers for received data (responding messages)

Register	DATA	Explanation
D1070 low	01 H	Address
D1071 low	03 H	Function
D1072 low	04 H	Number of data (counted by bytes)
D1073 low	17 H	Content of address 2102 H
D1074 low	70 H	
D1075 low	00 H	Content of address 2103 H
D1076 low	00 H	
D1077 low	FE H	CRC CHK Low
D1078 low	5C H	CRC CHK High

Program Example 3:

1. In the communication between PLC and VFD-S series AC motor drive (ASCII Mode, M1143 = Off), retry when communication time-out, data receiving error and sending address error occur.
2. When X0=On, PLC will read the data in VFFD-S data adress H2100 of device 01 and stores the data in ASCII format in D1070 ~ D1085. PLC will automatically convert the data into numerals and stores them in D1050 ~ D1055.
3. M1129 will be On when communication time-out occurs. The program will trigger M1129 and send request to M1122 for reading the data again.
4. M1140 will be On when data receiving error occurs. The program will trigger M1140 and send request to M1122 for reading the data again.
5. M1141 will be On when sending address error occurs. The program will trigger M1141 and send request to M1122 for reading the data again.



Remarks:

1. The activation criteria placed before the three instructions, API 100 MODRD, API 105 RDST, and API 150 MODRW (Function Code H03), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF); otherwise, the data stores in the receiving registers will be incorrect.
2. There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed at a time.

API	Mnemonic	Operands	Function	Controllers
101	MODWR	(S1) (S2) (n)	Write Modbus Data	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MODWR: 7 steps
S_1					*	*							*			
S_2					*	*							*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

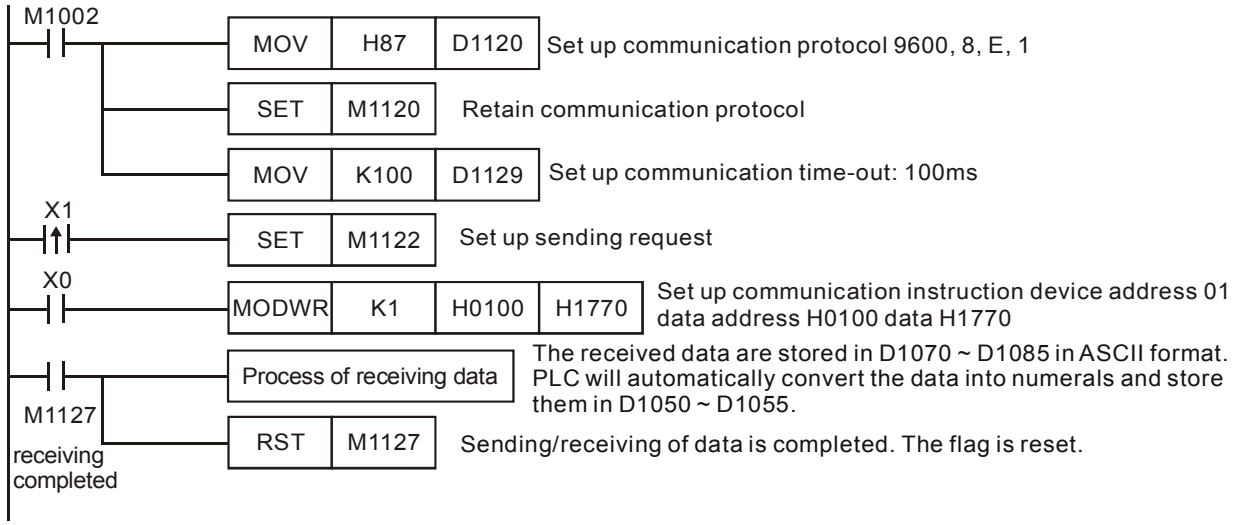
S_1: Address of communication device **S_2**: Address of data to be read **n**: Data to be written

Explanations:

1. Range of **S_1**: K0 ~ K254
2. See the specifications of each model for their range of use.
3. ES/EX/SS series MPU does not support E, F index register modification.
4. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143
5. MODWR is a drive instruction exclusively for peripheral communication equipment in MODBUS ASCII mode/RTU mode. The built-in RS-485 communication ports in Delta VFD drives (except for VFD-A series) are all compatible with MODBUS communication format. MODRD can be used for controlling communication (write data) of Delta drives.
6. If the address of **S_2** is illegal to the designed communication device, the device will respond with an error, PLC will records the error code in D1130 and M1140 will be On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. For error codes, see the user manual of VFD-S.
7. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1076. After receiving the feedback data is completed, PLC will auto-check if all data are correct. If there is an error, M1140 will be On.
8. After M1140 or M1141 turn On, the program will send a correct datum to the peripheral equipment. If the feedback datum is correct, M1140 and M1141 will be reset.

Program Example 1:

Communication between PLC and VFD-S series AC motor drives (ASCII Mode, M1143 = Off)



PLC ⇨ VFD-B, PLC sends: " **01 06 0100 1770 71** "

VFD-B ⇨ PLC, PLC receives: " **01 06 0100 1770 71** "

Registers for sent data (sending messages)

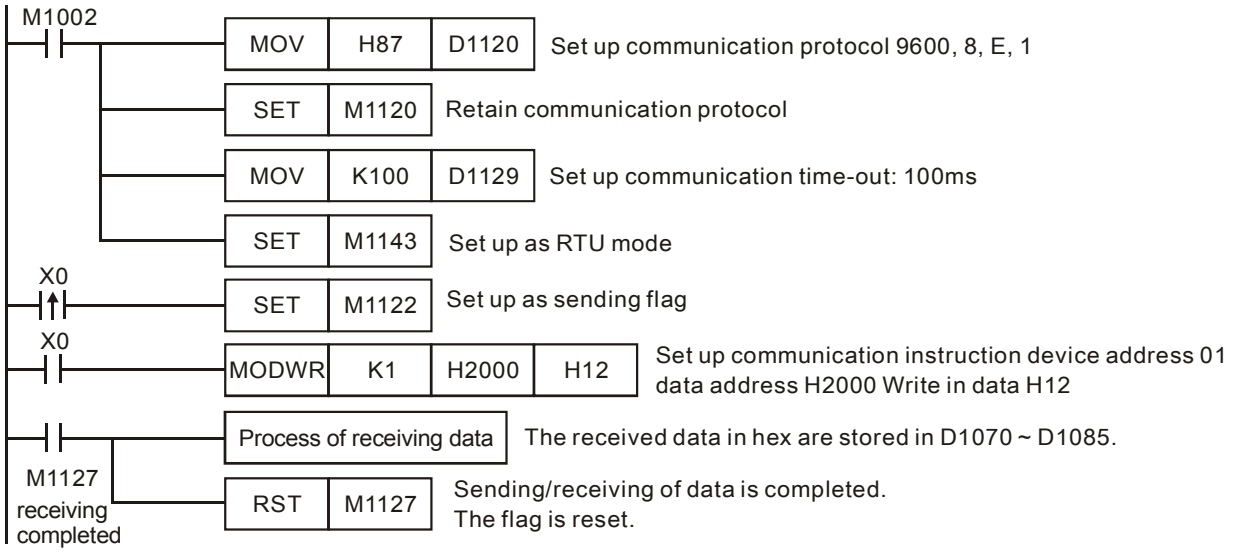
Register	DATA		Explanation	
D1089 low	'0'	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D1089 high	'1'	31 H	ADR 0	
D1090 low	'0'	30 H	CMD 1	Instruction code: CMD (1,0)
D1090 high	'6'	36 H	CMD 0	
D1091 low	'0'	30 H	Data address	
D1091 high	'1'	31 H		
D1092 low	'0'	30 H		
D1092 high	'0'	30 H		
D1093 low	'1'	31 H	Data contents	
D1093 high	'7'	37 H		
D1094 low	'7'	37 H		
D1094 high	'0'	30 H		
D1095 low	'7'	37 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
D1095 high	'1'	31 H	LRC CHK 0	

PLC receiving data register (response messages)

Register	DATA		Explanation
D1070 low	'0'	30 H	ADR 1
D1070 high	'1'	31 H	ADR 0
D1071 low	'0'	30 H	CMD 1
D1071 high	'6'	36 H	CMD 0
D1072 low	'0'	30 H	Data address
D1072 high	'1'	31 H	
D1073 low	'0'	30 H	
D1073 high	'0'	30 H	
D1074 low	'1'	31 H	Data content
D1074 high	'7'	37 H	
D1075 low	'7'	37 H	
D1075 high	'0'	30 H	
D1076 low	'7'	37 H	LRC CHK 1
D1076 high	'1'	31 H	LRC CHK 0

Program Example 2:

Communication between PLC and VFD-S series AC motor drives (RTU Mode, M1143 = On)



PLC ⇨ VFD-S, PLC sends: **01 06 2000 0012 02 07**

VFD-S ⇨ PLC, PLC receives: **01 06 2000 0012 02 07**

Registers for sent data (sending messages)

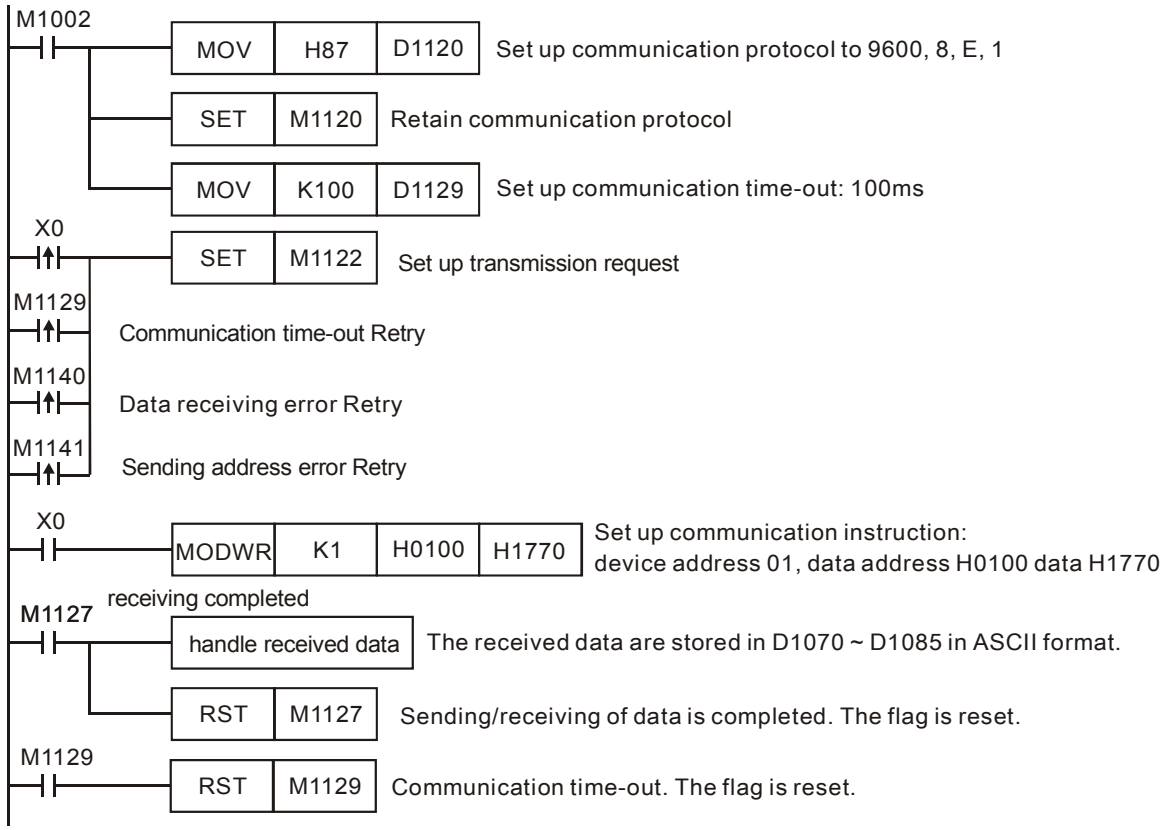
Register	DATA	Explanation
D1089 low	01 H	Address
D1090 low	06 H	Function
D1091 low	20 H	Data address
D1092 low	00 H	
D1093 low	00 H	Data contents
D1094 low	12 H	
D1095 low	02 H	CRC CHK Low
D1096 low	07 H	CRC CHK High

Registers for received data (responding messages)

Register	DATA	Explanation
D1070 low	01 H	Address
D1071 low	06 H	Function
D1072 low	20 H	Data address
D1073 low	00 H	
D1074 low	00 H	Data contents
D1075 low	12 H	
D1076 low	02 H	CRC CHK Low
D1077 low	07 H	CRC CHK High

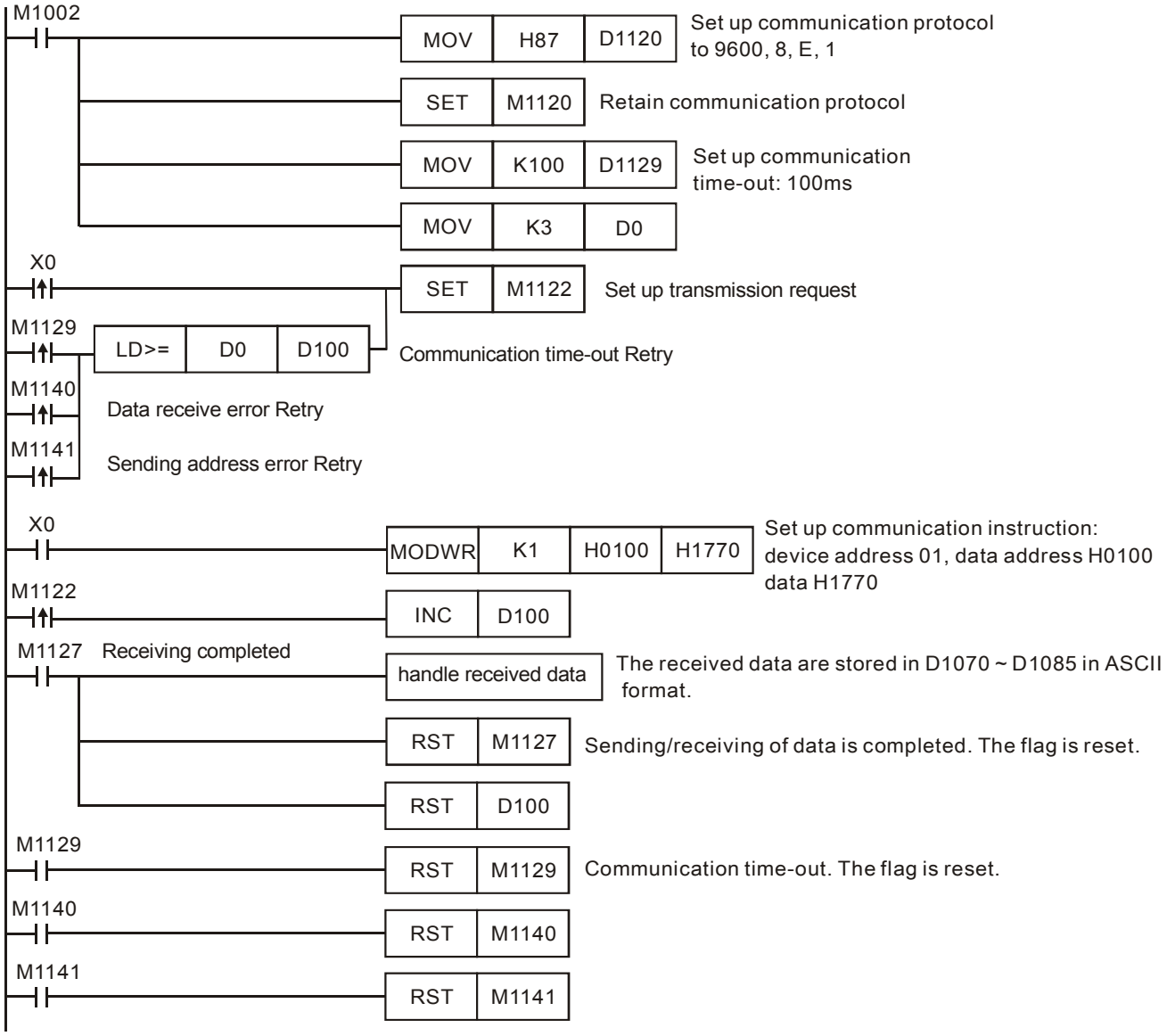
Program Example 3:

1. In the communication between PLC and VFD-S series AC motor drive (ASCII Mode, M1143 = Off), retry when communication time-out, data receiving error and sending address error occur.
2. When X0 = On, PLC will write H1770(K6000) into VFD-S data adress H0100 of device 01.
3. M1129 will be On when communication time-out occurs. The program will trigger M1129 and send request to M1122 for writing the data again.
4. M1140 will be On when data receiving error occurs. The program will trigger M1140 and send request to M1122 for writing the data again.
5. M1141 will be On when sending address error occurs. The program will trigger M1141 and send request to M1122 for writing the data again.



Program Example 4:

1. In the communication between PLC and VFD-S series AC motor drive (ASCII Mode, M1143 = Off), retry when communication time-out, data receiving error and sending address error occur. Times of retry = D0 (default = 3). When communication Retry is successful, the user can return to controlling by triggering criteria.
2. When X0 = On, PLC will write H1770(K6000) into VFD-S data adress H0100 of device 01.
3. M1129 will be On when communication time-out occurs. The program will trigger M1129 and send request to M1122 for writing the data again. Times of Retry = D0 (default = 3)
4. M1140 will be On when data receiving error occurs. The program will trigger M1140 and send request to M1122 for writing the data again. Times of Retry = D0 (default = 3)
5. M1141 will be On when sending address error occurs. The program will trigger M1141 and send request to M1122 for writing the data again. Times of Retry = D0 (default = 3)



Remarks:

1. For the registers for flag settings, see explanations in API 80 RS.
2. The activation criteria placed before the two instructions, API 101 MODWR and API 150 MODRW (Function Code H06, H10), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF) and have to enable sending request M1122 first.
3. There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed at a time.

API	Mnemonic	Operands	Function	Controllers
102	FWD	S_1 S_2 n	Forward Running of VFD-A	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			FWD: 7 steps
S_2					*	*							*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

API	Mnemonic	Operands	Function	Controllers
103	REV	S_1 S_2 n	Reverse Running of VFD-A	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			REV: 7 steps
S_2					*	*							*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

API	Mnemonic	Operands	Function	Controllers
104	STOP	S_1 S_2 n	Stop VFD-A	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			STOP: 7 steps
S_2					*	*							*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Address of communication device **S_2**: Rotation frequency of AC motor drive **n**: Target to be instructed

Explanations:

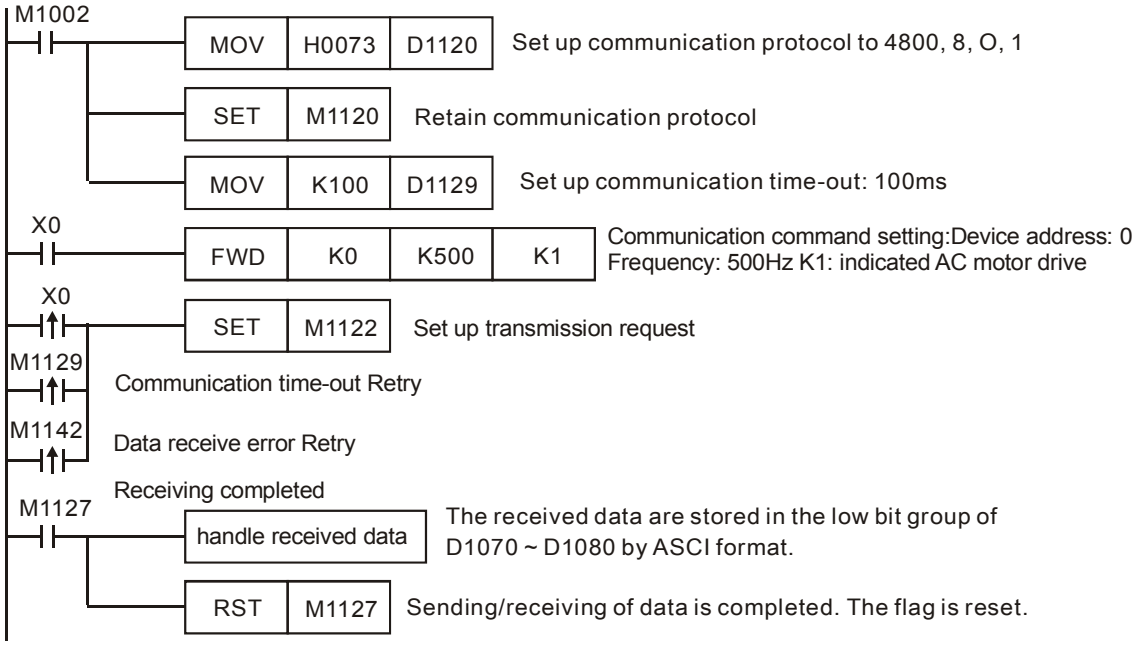
1. Range of **S_1**: K0 ~ K31
2. Range of **n**: K1 or K2
3. See the specifications of each model for their range of use.
4. ES series MPU does not support E, F index register modification.
5. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143
6. FWD/REV/STOP are handy instructions exclusively for Delta VFD-A/H series AC motor drive to perform forward running/reverse running/stop. Be sure to set up communication time-out (D1129) when executing this instruction.
7. **S_2** = operation frequency of AC motor drive. Set frequency in A-series AC motor drive: K0 ~ K4,000 (0.0Hz ~

400.0Hz). Set frequency in H-series: K0 ~ K1,500 (0Hz ~ 1,500Hz).

8. **n** = instructed target. **n**=1: AC motor drive at designated address. **n**=2: all connected AC motor drives.
9. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1080. After receiving the feedback data is completed, PLC will auto-check if all data are correct. If there is an error, M1142 will be On. When **n** = 2, PLC will not receive any data.

Program Example :

Communication between PLC and VFD-A series AC drives, retry for communication time-out and received data error.



PLC ⇨ VFD-A, PLC sends: "**C ♥ ☺ 0001 0500** "

VFD-A ⇨ PLC, PLC sends: "**C ♥ ♠ 0001 0500** "

Registers for sent data (sending messages)

Register	DATA		Explanation
D1089 low	'C'	43 H	Start word of instruction
D1090 low	'♥'	03 H	Checksum
D1091 low	'☺'	01 H	Instructed target
D1092 low	'0'	30 H	Communication address
D1093 low	'0'	30 H	
D1094 low	'0'	30 H	
D1095 low	'1'	31 H	
D1096 low	'0'	30 H	Running instruction
D1097 low	'5'	35 H	
D1098 low	'0'	30 H	
D1099 low	'0'	30 H	

Registers for received data (responding messages)

Register	DATA		Explanation
D1070 low	'C'	43 H	Start word of instruction
D1071 low	'♥'	03 H	Checksum
D1072 low	'♠'	06 H	Reply authorization (correct: 06H, incorrect: 07 H)
D1073 low	'0'	30 H	Communication address
D1074 low	'0'	30 H	
D1075 low	'0'	30 H	
D1076 low	'1'	31 H	
D1077 low	'0'	30 H	Running instruction
D1078 low	'5'	35 H	
D1079 low	'0'	30 H	
D1080 low	'0'	30 H	

Remarks:

There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed at a time.

API	Mnemonic	Operands	Function	Controllers
105	RDST	(S) (n)	Read VFD-A Status	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			RDST: 5 steps
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Address of communicatino device **n**: Target to be instructed

Explanations:

1. Range of **S**: K0 ~ K31
2. Range of **n**: K0 ~ K3
3. See the specifications of each model for their range of use.
4. ES series MPU does not support E, F index register modification.
5. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143
6. **n**: Instructed target (to be read) in AC motor drive
   n=0, frequency
   n=1, output frequency
   n=2, output current
   n=3, running instruction
7. Data sent back (feedback) from AC motor drive (11 bytes, see VFD-A user manual) are stored in the low bytes of D1070 ~ D1080.

   ”Q, S, B, Uu, Nn, ABCD”

Feedback	Explanation	Data storage
Q	Start word: ’Q’ (51H).	D1070 low
S	Checksum code: 03H.	D0171 low
B	Instruction authorization. correct: 06H, incorrect: 07H.	D1072 low
U	Communication address (address: 00~31). ”Uu” = (“00” ~ ”31”) indicated in ASCII format.	D1073 low
U		D1074 low
N	Instructed target (00 ~ 03).”Nn” = (“00 ~ 03”) indicated in ASCII format.	D1075 low
N		D1076 low
A	Instructed data. The content of ”ABCD” differs upon the instructed targets (00 ~ 03). 00 ~ 03 indicate frequency, current and running mode respectively. Please refer to the explanations below for details.	D1077 low
B		D1078 low
C		D1079 low
D		D1080 low
	Nn = “00” Frequency instruction = ABC.D (Hz) Nn = “01” Output instruction = ABC.D (Hz) Nn = “02” Output current = ABC.D (A) PLC will automatically convert the ASCII characters of ”ABCD” into numerals and store the numeral in D1050. For example, assume ”ABCD” = “0600”, PLC will convert ABCD into K0600 (0258 H) and store it in the special register D1050.	

	Nn = “03” Running instruction							
	‘A’ =	‘0’ Stop, ‘1’ Forward running ‘2’ Stop, ‘3’ Reverse running ‘4’ JOG (forward), ‘5’ JOG (forward) ‘6’ JOG (reverse) ‘7’ JOG (reverse) ‘8’ Abnormal						
		ES series PLCs will convert the ASCII characters of ”A” into a numeral and store the numeral in D1051. For example, assume ”A” = “3”, PLC will convert A into K3 and store it in the special register D1051. SA/EH series PLCs will store the numeral in low bytes of D1051.						
	‘B’ =	b7	b6	b5	b4	Source of running instruction		
		0	0	0	0	Digital keypad		
		0	0	0	1	1st Step Speed		
		0	0	1	0	2nd Step Speed		
		0	0	1	1	3rd Step Speed		
		0	1	0	0	4th Step Speed		
		0	1	0	1	5th Step Speed		
		0	1	1	0	6th Step Speed		
		0	1	1	1	7th Step Speed		
		1	0	0	0	JOG frequency		
		1	0	0	1	Analog signal frequency instruction		
		1	0	1	0	RS-485 communication interface		
		1	0	1	1	Up/Down control		
		b3 = 0 No DC braking stop 1 DC braking stop						
		b2 = 0 No DC braking startup 1 DC braking startup						
		b1 = 0 Forward running 1 Reverse running						
		b0 = 0 Stop 1 Running						
		ES series PLCs will store ”B” in special auxiliary relay M1168 (b0) ~ M1175 (b7). SA/EH series PLCs will store “B” (in hex) in the high bytes of special register D1051.						
	“CD” =	“00”	No abnormal record	“10”	OcA			
		“01”	oc	“11”	Ocd			
		“02”	ov	“12”	Ocn			
		“03”	oH	“13”	GFF			
		“04”	oL	“14”	Lv			
		“05”	oL1	“15”	Lv1			
		“06”	EF	“16”	cF2			
		“07”	cF1	“17”	bb			
		“08”	cF3	“18”	oL2			
		“09”	HPF	“19”				
		ES/SA/EH series PLCs will convert the ASCII characters of ”CD” into a numerals and store the numeral in D1052. For example, assume ”CD” = “16”, PLC will convert CD into K16 and store it in the special register D1052.						

Remarks:

1. The activation criteria placed before the three instructions, API 100 MODRD, API 105 RDST and API 150 MODRW (Function Code 03), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF); otherwise, the data stores in the receiving registers will be incorrect.
2. For the registers for flag settings, see explanations in API 80 RS.
3. There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed at a time.

API	Mnemonic	Operands	Function	Controllers
106	RSTEF	(S) (n)	Reset Abnormal VFD-A	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			RSTEF: 5 steps
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Address of communication device **n**: Target to be instructed

Explanations:

1. Range of **S**: K0 ~ K31
2. Range of **n**: K1 or K2
3. See the specifications of each model for their range of use.
4. Flags: See API 80 RS for explanations on M1120 ~ M1131, M1140 ~ M1143
5. RSTEF is a handy communication instruction exclusively for Delta VFD-A series AC motor drives and is used for reset when the AC motor drive operates abnormally.
6. **n**: instructed target. **n**=1: AC motor drive at assigned address. **n**=2: all connected AC motor drives.
7. The feedback (returned) data from the peripheral equipment will be stored in D1070 ~ D1089. If **n** = 2, there will be no feedback data.

Remarks:

1. The activation criteria placed before the three instructions, API 100 MODRD, API 105 RDST and API 150 MODRW (Function Code 03), cannot use rising-edge contacts (LDP, ANDP ORP) and falling-edge contacts (LDF, ANDF, ORF); otherwise, the data stores in the receiving registers will be incorrect.
2. For the registers for flag settings, see explanations in API 80 RS.
3. There is no limitation on the times of using this instruction in the program, but only one instruction is allowed to be executed at a time.

API	Mnemonic		Operands	Function	Controllers
107	LRC	P	(S) (n) (D)	Checksum LRC Mode	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			LRC, LRCP: 7 steps
n					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

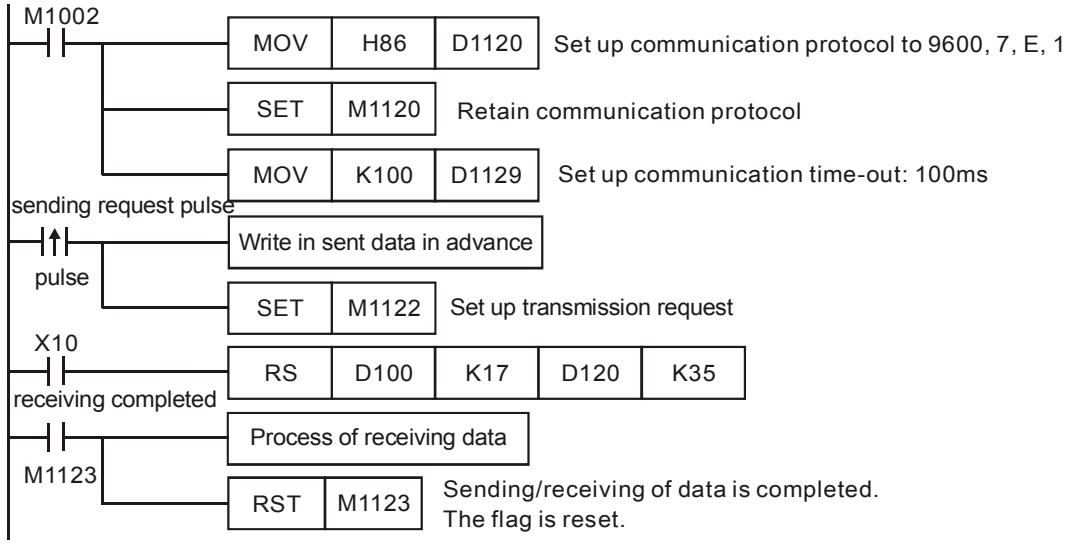
S: Start operation device for ASCII mode checksum **n**: Number of calculated bits **D**: Start device for storing the operation result LRC checksum: See remarks.

Explanations:

1. Range of **n**: K1 ~ K256
2. See the specifications of each model for their range of use.
3. Flag: M1161 (switching between 8/16 bit modes)
4. **n** has to be even. If **n** does not fall within its range, an operation error will occur, the instruction will not be executed, M1067, M1068 = On and D1067 will record the error code H'0E1A.
5. In 16-bit conversion mode: When M1161 = Off, **S** divides its hex data area into higher 8 bits and lower 8 bits and performs LRC checksum operation on each bit. The data will be sent to the higher 8 bits and lower 8 bits in **D**. **n** = the number of calculated bits.
6. In 8-bit conversion mode: When M1161 = On, **S** divides its hex data area into higher 8 bits (invalid data) and lower 8 bits and performs LRC checksum operation on each bit. The data will be sent to the lower 8 bits in **D** and occupy 2 registers. **n** = the number of calculated bits. (All higher bits in **D** are "0".)

Program Example:

When PLC communicates with VFD-S series AC motor drives (In ASCII mode, M1143 = Off), (In 8-bit mode, M1161 = On), the sent data write in advance the 6 data read starting from H2101 of VFD-S.

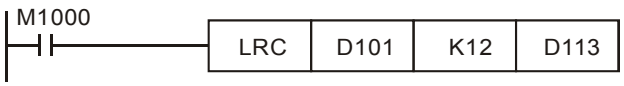


PLC ⇨ VFD-S, PLC sends: **“: 01 03 2101 0006 D4 CR LF ”**

Registers for sent data (sending messages)

Register	DATA		Explanation	
D100 low	‘: ’	3A H	STX	
D101 low	‘0’	30 H	ADR 1 ADR 0	Address of AC motor drive: ADR (1,0)
D102 low	‘1’	31 H		
D103 low	‘0’	30 H	CMD 1 CMD 0	Instruction code: CMD (1,0)
D104 low	‘3’	33 H		
D105 low	‘2’	32 H	Starting data address	
D106 low	‘1’	31 H		
D107 high	‘0’	30 H		
D108 low	‘1’	31 H		
D109 low	‘0’	30 H	Number of data (counted by words)	
D110 low	‘0’	30 H		
D111 low	‘0’	30 H		
D112 low	‘6’	36 H		
D113 low	‘D’	44 H	LRC CHK 1 LRC CHK 0	Error checksum: LRC CHK (0,1)
D114 low	‘4’	34 H		
D115 low	CR	A H	END	
D116 low	LF	D H		

The error checksum LRC CHK (0,1) can be calculated by LRC instruction (in 8-bit mode, M1161 = On).



LRC checksum: 01 H + 03 H + 21 H + 01 H + 00 H + 06 H = 2C H. Obtain 2’s complement, D4H, and store ‘D’(44H) in the lower 8 bits of D113 and ‘4’(34H) in the lower 8 bits of D114.

Remarks:

1. The format of ASCII mode with a communication datum

STX	‘: ’	Start word = ‘: ’ (3AH)
Address Hi	‘ 0 ’	Communication:
Address Lo	‘ 1 ’	8-bit address consists of 2 ASCII codes
Function Hi	‘ 0 ’	Function code:
Function Lo	‘ 3 ’	8-bit function consists of 2 ASCII codes
DATA (n-1)	‘ 2 ’	Data content:
…….	‘ 1 ’	n × 8-bit data consists of 2n ASCII codes
DATA 0	‘ 0 ’	
	‘ 2 ’	
	‘ 0 ’	
	‘ 0 ’	
	‘ 0 ’	
	‘ 2 ’	
LRC CHK Hi	‘ D ’	LRC checksum:
LRC CHK Lo	‘ 7 ’	8-bit checksum consists of 2 ASCII codes
END Hi	CR	End word:
END Lo	LF	END Hi = CR (0DH), END Lo = LF(0AH)

2. LRC checksum: 2’s complement of the summed up value of communication address and data. For example, 01 H + 03 H + 21 H + 02 H + 00 H + 02 H = 29 H. Obtain 2’s complement = D7H.

API	Mnemonic		Operands	Function	Controllers
108	CRC	P	S n D	Checksum CRC Mode	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			CRC, CRCP: 7 steps
n					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

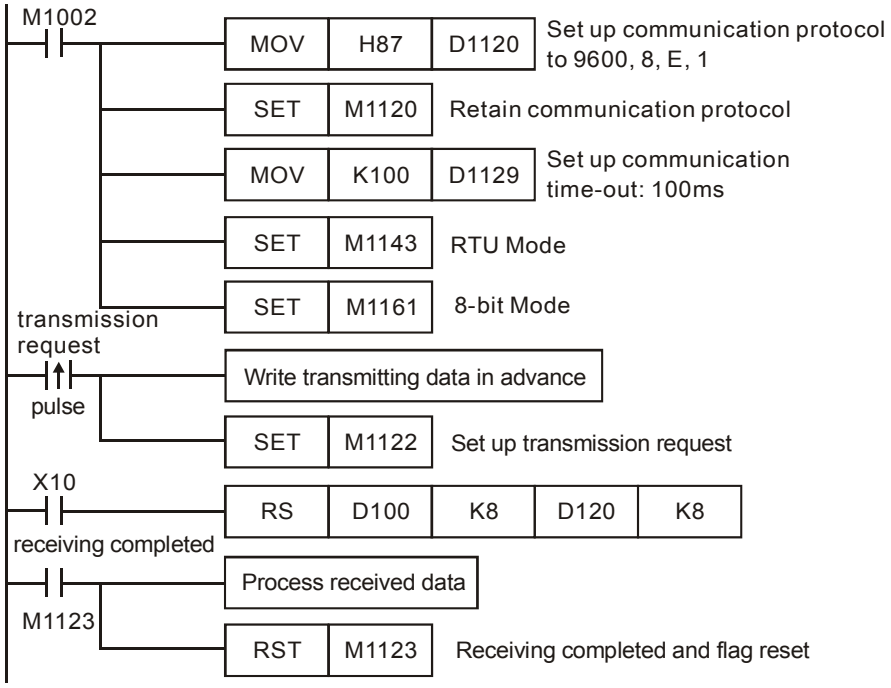
S: Start operation device for RTU mode checksum **n**: Number of calculated bits **D**: Start device for storing the operation result CRC checksum: See remarks.

Explanations:

1. Range of **n**: K1 ~ K256
2. Flags: M1161 (switching between 8/16-bit modes)
3. If **n** does not fall within its range, an operation error will occur, the instruction will not be executed, M1067, M1068 = On and D1067 will record the error code H'0E1A.
4. In 16-bit conversion mode: When M1161 = Off, **S** divides its hex data area into higher 8 bits and lower 8 bits and performs CRC checksum operation on each bit. The data will be sent to the higher 8 bits and lower 8 bits in **D**. **n** = the number of calculated bits.
5. In 8-bit conversion mode: When M1161 = On, **S** divides its hex data area into higher 8 bits (invalid data) and lower 8 bits and performs CRC checksum operation on each bit. The data will be sent to the lower 8 bits in **D** and occupy 2 registers. **n** = the number of calculated bits. (All higher 8 bits in **D** are "0".)

Program Example:

When PLC communicates with VFD-S series AC motor drives (In RTU mode, M1143 = On), (In 16-bit mode, M1161 = On), the sent data write in advance H12 into H2000 of VFD-S.

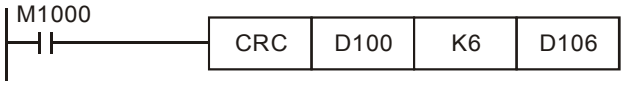


PLC ⇨ VFD-S, PLC sends: **01 06 2000 0012 02 07**

Registers for sent data (sending messages)

Register	DATA	Explanation
D100 low	01 H	Address
D101 low	06 H	Function
D102 low	20 H	Data address
D103 low	00 H	
D104 low	00 H	Data content
D105 low	12 H	
D106 low	02 H	CRC CHK 0
D107 low	07 H	CRC CHK 1

The error checksum CRC CHK (0,1) can be calculated by CRC instruction (in 8-bit mode, M1161 = On).



CRC checksum: 02 H is stored in the lower 8 bits of D106 and 07 H in the lower 8 bits of D107,

Remarks:

1. The format of RTU mode with a communication datum

START	Time interval
Address	Communication address: 8-bit binary
Function	Function code: 8-bit binary
DATA (n-1)	Data content: n × 8-bit data
…….	
DATA 0	
CRC CHK Low	CRC checksum: 16-bit CRC checksum consists of 2 8-bit binaries
CRC CHK High	
END	Time interval

2. CRC checksum starts from Address and ends at Data content.

The operation of CRC checksum:

Step 1: Make the 16-bit register (CRC register) = FFFFH

Step 2: Exclusive OR the first 8-bit byte message instruction and the low-bit 16-bit CRC register. Store the result in CRC register.

Step 3: Shift the CRC register one bit to the right and fill 0 in the higher bit.

Step 4: Check the value that shifts to the right. If it is 0, store the new value from Step 3 into the CRC register, otherwise, Exclusive OR A001H and the CRC register, and store the result in the CRC register.

Step 5: Repeat Step 3 ~ 4 and finish calculating the 8 bits.

Step 6: Repeat Steps 2 ~ 5 for obtaining the next 8-bit message instruction until all the message instructions are calculated. In the end, the obtained CRC register value is the CRC checksum. Be aware that CRC checksum should be placed in the checksum of the message instruction.

API	Mnemonic	Operands	Function	Controllers
109	SWRD P	D	Read Digital Switch	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D								*	*	*	*	*	*	*	*	SWRD, SWRDP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

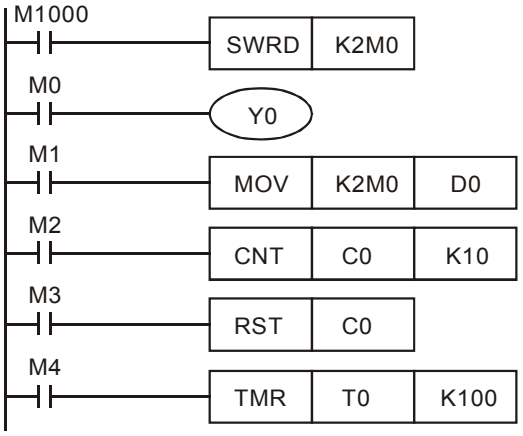
D: Device for storing the read value

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1104 ~ M1111 (status of digital switch)
3. This instruction stores the value read from digital switch function card into **D**.
4. The read value is stored in the low byte in D. Every switch has a corresponding bit.
5. When there is no digital function card inserted, the error message C400 (hex) will appear in grammar check.

Program Example:

1. There are l 8 DIP switches on the digital switch function card. After the switches are read by SWRD instruction, the status of each switch will correspond to M0 ~ M7.



2. The status of M0 ~ M7 can be executed by each contact instruction.
3. The execution of END instruction indicates that the process of input is completed. REF (I/O refresh) instruction will be invalid.
4. When SWRD instruction uses the data in digital switch function card, it can read minimum 4 bits (K1Y*, K1M* or K1S*).

Remarks:

When digital switch function card is inserted, the status of the 8 DIP switches will correspond to M1104 ~ M1111.

API	Mnemonic	Operands	Function	Controllers
110	D ECMP P	(S1) (S2) (D)	Floating Point Compare	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DECMP, DECMPP: 13 steps
S_2					*	*							*			
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

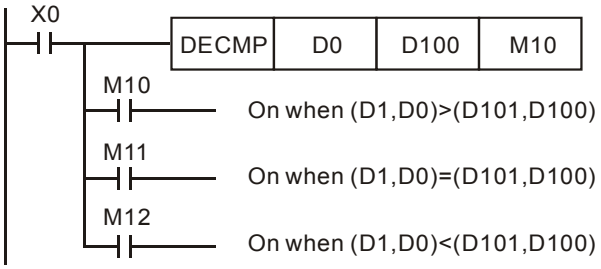
S_1: Binary floating point comparison value 1 **S_2**: Binary floating point comparison value 2 **D**: Comparison result

Explanations:

1. **D** occupies 3 consecutive devices.
2. See the specifications of each model for their range of use.
3. The binary floating point values **S_1** and **S_2** are compared with each other. The comparison result (>, =, <) is stored in **D**.
4. If **S_1** or **S_2** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the comparison.

Program Example:

1. Designated device M10 and M10 ~ M12 are automatically occupied.
2. When X0 = On. DECMP instruction will be executed and one of M10 ~ M12 will be On. When X0 = Off, DECMP instruction will not be executed and M10 ~ M12 will remain their status before X0 = Off.
3. To obtain results ≧, ≦, ≠, serial-parallel M10 ~ M12.
4. Use RST or ZRST instruction to clear the result.



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
111	D	EZCP	P	(S1) (S2) (S) (D)	Floating Point Zone Compare	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DEZCP, DEZCPP: 17 steps
S_2					*	*							*			
S					*	*							*			
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

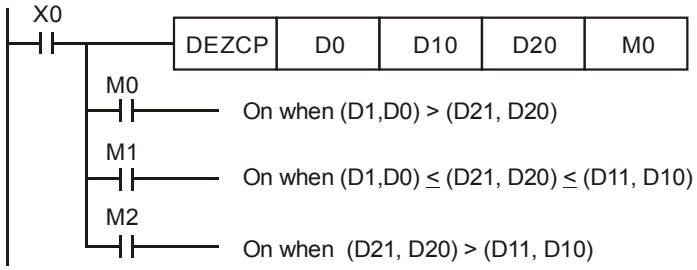
S_1: Lower bound of binary floating point **S_2**: Upper bound of binary floating point **S**: Binary floating point comparison result **D**: Comparison result

Explanations:

1. **D** occupied 3 consecutive devices.
2. **S_1 ≤ S_2**. See the specifications of each model for their range of use.
3. **S** is compared with **S_1** and **S_2** and the result (>, =, <) is stored in **D**.
4. If **S_1** or **S_2** is andesignated constant K or H, the instruction will convert the constant into a binary floating point value before the comparison.
5. When **S_1 > S_2**, **S_1** will be used as upper/lower bound for the comparison.

Program Example:

1. Designated device M0 and M0 ~ M2 are automatically occupied.
2. When X0 = On. DEZCP instruction will be executed and one of M0 ~ M2 will be On. When X0 = Off, EZCP instruction will not be executed and M0 ~ M2 will remain their status before X0 = Off.
3. Use RST or ZRST instruction to clear the result.



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic	Operands	Function	Controllers
112	D MOVR P	(S) (D)	Move Floating Point Data	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S																DMOVR, DMOVRP: 9 steps
D								*	*	*	*	*	*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

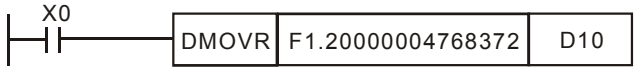
S：Source floating point data **D**：Destination device

Explanations:

1. **S** can only be in floating point (FX.XX).
2. See the specifications of each model for their range of use.
3. This instruction is able to enter floating point values directly in **S**.
4. When the instruction is executed, the content in **S** is moved directly into **D**. When the instruction is not executed, the content in **D** will not be modified.

Program Example:

1. User DMOVR instruction to move 32-bit floating point data.
2. When X0 = Off, the content in (D11、D10) remains unchanged. When X0 = On, the present value F1.20000004768372 will be moved to data registers (D11, D10).



Remarks:

This instruction only supports ES V6.1, SA/SX_V1.1, SV_V1.2, EH_V1.2, EH2/SV_V1.0 and above versions.

API	Mnemonic			Operands	Function	Controllers
116	D	RAD	P	(S) (D)	Angle → Radian	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DRAD, DRADP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

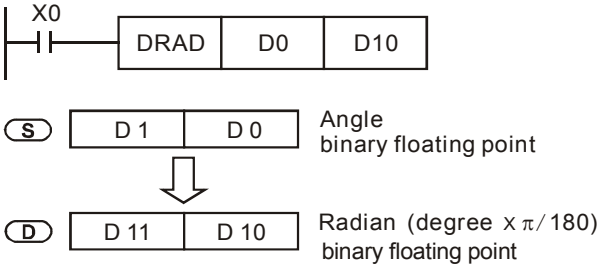
S: Source (angle) **D**: Result (radian)

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. Radian = degree × (π /180)
4. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.
5. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.
6. If the result = 0, the zero flag M1020 = On.

Program Example:

When X0 = On, designate the degree of binary floating point (D1, D0). Convert the angle into radian and store the result in binary floating point in (D11, D10).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
117	D	DEG	P	(S) (D)	Radian → Angle	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DDEG, DDEGP: 9 steps
S					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

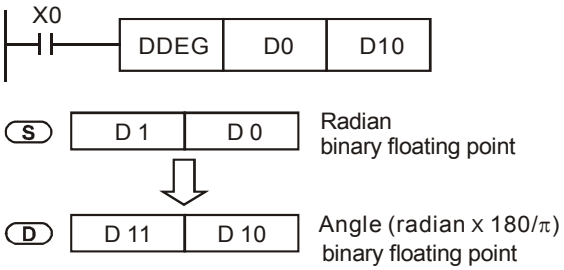
S: Source (radian) **D**: Result (angle)

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. Degree = radian × (180/ π)
4. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.
5. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.
6. If the result = 0, the zero flag M1020 = On.

Program Example:

When X0 = On, designate the angle of binary floating point (D1, D0). Convert the radian into angle and store the result in binary floating point in (D11, D10).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
118	D	EBCD	P	(S) (D)	Float to Scientific Conversion	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			DEBCD, DEBCDP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

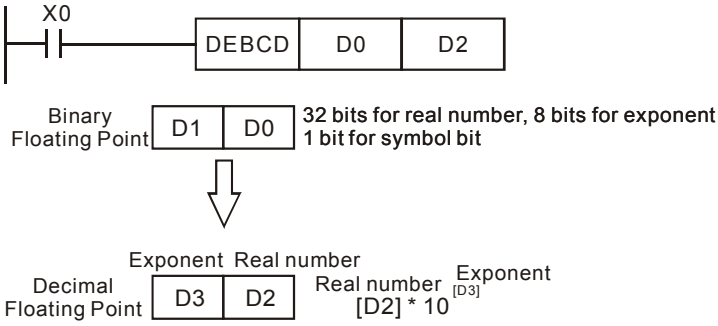
S: Source **D**: Result

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. This instruction converts binary floating point value in the register designated by **S** into decimal floating point value and stores it in the register designated by **D**.
4. PLC conducts floating point operation in binary format. DEBCD instruction is exclusively for converting floating points from binary to decimal.
5. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
6. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
7. If the result = 0, the zero flag M1020 = On.

Program Example:

When X0 = On, the binary floating points in D1 and D0 will be converted into decimal floating points and stored in D3 and D2.



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
119	D	EBIN	P	(S) (D)	Scientific to Float Conversion	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			DEBIN, DEBINP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

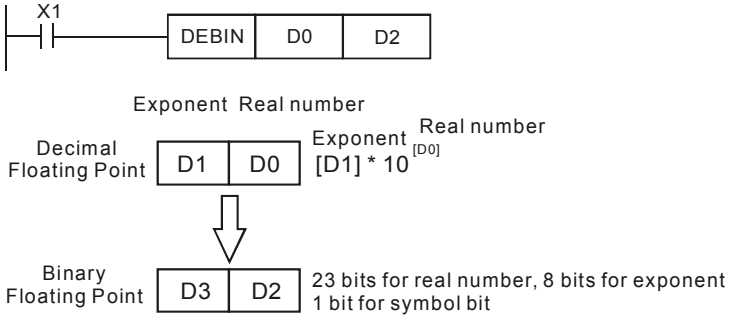
S: Source **D**: Result

Explanations:

1. See the specifications of each model for their range of use.
2. Flag: M1020 (zero flag)
3. This instruction converts decimal floating point value in the register designated by **S** into binary floating point value and stores it in the register designated by **D**.
4. DEBIN instruction is exclusively for converting floating points from decimal to binary.
5. Range of decimal floating point real numbers: -9.999 ~ +9,999. Range of exponants: -41 ~ +35. Range of PLC decimal floating points: $\pm 1{,}175 \times 10^{-41} \sim \pm 3{,}402 \times 10^{+35}$.
6. If the result = 0, the zero flag M1020 = On.

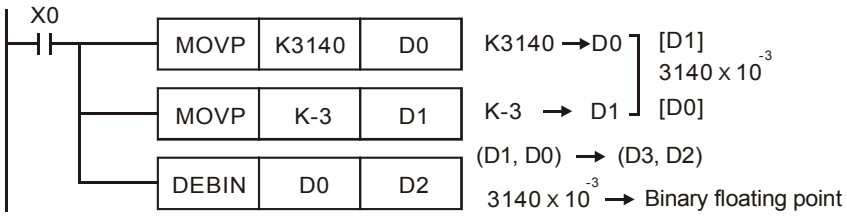
Program Example 1:

When X1 = On, the decimal floating points in D1 and D0 will be converted into binary floating points and stored in D3 and D2.



Program Example 2:

1. Use FLT instruction (API 149) to convert BIN integer into binary floating point before performing floating point operation. The value to be converted must be BIN integer and use DEBIN instruction to convert the floating point into a binary one.
2. When X0 = On, move K3,140 to D0 and K-3 to D1 to generate decimal floating point ($3.14 = 3140 \times 10^{-3}$).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
120	D	EADD	P	(S1) (S2) (D)	Floating Point Addition	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DEADD, DEADDP: 13 steps
S_2					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Summand **S_2**: Addend **D**: Sum

Explanations:

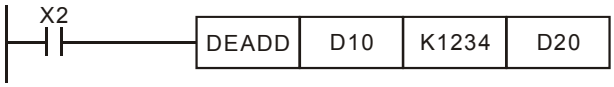
1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. **S_1** + **S_2** = **D**. The floating point value in the register designated by **S_1** and **S_2** are added up and the result is stored in the register designated by **D**. The addition is conducted in binary floating point system.
4. If **S_1** or **S_2** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.
5. **S_1** and **S_2** can designate the same register. In this case, if the "continuous execution" instruction is in use, during the period when the criteria contact in On, the register will be added once in every scan by pulse execution instruction DEADDP.
6. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
7. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
8. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X0 = On, binary floating point (D1, D0) + binary floating point (D3, D2) and the result is stored in (D11, D10).

Program Example 2:

When X2 = On, binary floating point (D11, D10) + K1234 (automatically converted into binary floating point) and the result is stored in (D21, D20).



Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands	Function	Controllers
121	D	ESUB	P	(S1) (S2) (D)	Floating Point Subtraction	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DESUB, DESUBP: 13 steps
S_2					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Minuend **S_2**: Subtrahend **D**: Remainder

Explanations:

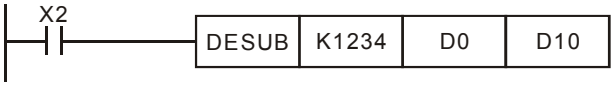
1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. **S_1** − **S_2** = **D**. The floating point value in the register designated by **S_2** is subtracted from the floating point value in the register assigned by **S_1** and the result is stored in the register designated by **D**. The subtraction is conducted in binary floating point system.
4. If **S_1** or **S_2** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.
5. **S_1** and **S_2** can designate the same register. In this case, if the "continuous execution" instruction is in use, during the period when the criteria contact in On, the register will be subtracted once in every scan by pulse execution instruction DESUBP.
6. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
7. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
8. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X0 = On, binary floating point (D1, D0) － binary floating point (D3, D2) and the result is stored in (D11, D10).

Program Example 2:

When X2 = On, K1234 (automatically converted into binary floating point) － binary floating point (D1, D0) and the result is stored in (D11, D10).



Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands	Function	Controllers
122	D	EMUL	P	(S1) (S2) (D)	Floating Point Multiplication	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DEMUL, DEMULP: 13 steps
S_2					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Multiplicand **S_2**: Multiplicator **D**: Product

Explanations:

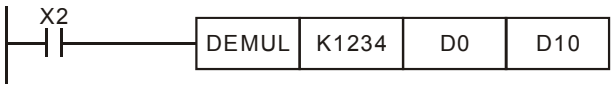
1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. **S_1** × **S_2** = **D**. The floating point value in the register assigned by **S_1** is multiplied with the floating point value in the register designated by **S_2** and the result is stored in the register designated by **D**. The multiplication is conducted in binary floating point system.
4. If **S_1** or **S_2** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.
5. **S_1** and **S_2** can designate the same register. In this case, if the "continuous execution" instruction is in use, during the period when the criteria contact in On, the register will be multiplied once in every scan by pulse execution instruction DEMULP.
6. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
7. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
8. If the result = 0, the zero flag M1020 = On.

Program Example 1:

When X1 = On, binary floating point (D1, D0) × binary floating point (D11, D10) and the result is stored in (D21, D20).

Program Example 2:

When X2 = On, K1234 (automatically converted into binary floating point) × binary floating point (D1, D0) and the result is stored in (D11, D10).



Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands	Function	Controllers
123	D	EDIV	P	(S1) (S2) (D)	Floating Point Division	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DEDIV, DEDIVP: 13 steps
S_2					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

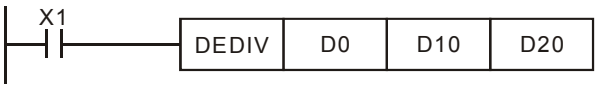
S_1: Dividend **S_2**: Divisor **D**: Quotient and remainder

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. **S_1** ÷ **S_2** = **D**. The floating point value in the register designated by **S_1** is divided by the floating point value in the register assigned by **S_2** and the result is stored in the register designated by **D**. The division is conducted in binary floating point system.
4. If **S_1** or **S_2** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.
5. If **S_2** = 0, operation error will occur, the instruction will not be executed, M1067, M1068 = On and D1067 will recorded the error code H'0E19.
6. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
7. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
8. If the result = 0, the zero flag M1020 = On.

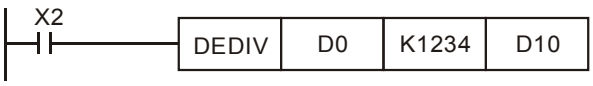
Program Example 1:

When X1 = On, binary floating point (D1, D0) ÷ binary floating point (D11, D10) and the quotient is stored in (D21, D20).



Program Example 2:

When X2 = On, binary floating point (D1, D0) ÷ K1234 (automatically converted into binary floating point) and the result is stored in (D11, D10).



Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands	Function	Controllers
124	D	EXP	P	(S) (D)	Exponent of Binary Floating Point	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DEXP, DEXPP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

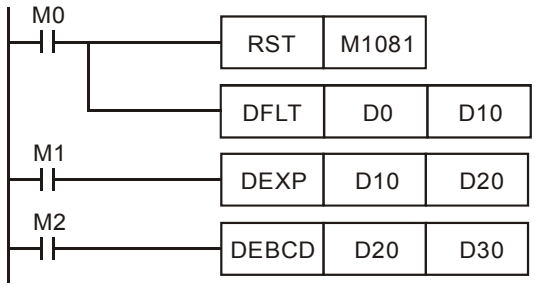
S: Device for operation source **D**: Device for operation result

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. e = 2.71828 as the base and **S** as exponent for EXP operation: EXP〔D +1, D〕= 〔**S** +1, **S**〕
4. Both positive and negative values are valid for **S**. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, **S** should be converted into a floating point value.
5. The content in **D** = e^{S}; e = 2.71828, ^{S} = designated source data
6. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
7. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
8. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) into binary floating point and store it in register (D11, D10).
2. When M1= On, use (D11, D10) as the exponent for EXP operation and store the binary floating point result in register (D21, D20).
3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × $10^{[D31]}$) and store it in register (D31, D30).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
125	D	LN	P	(S) (D)	Natural Logarithm of Binary Floating Point	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DLN, DLNP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

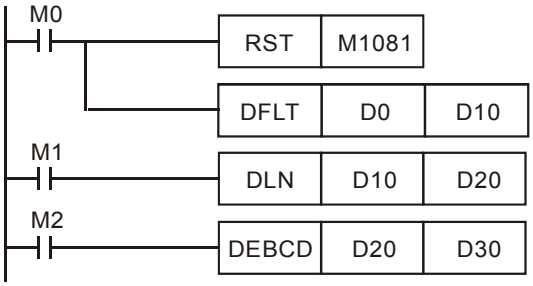
S: Device for operation source **D**: Device for operation result

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. This instruction performs natural logarithm "LN" operation by **S**: LN〔**S** + 1, **S**〕= 〔**D** + 1, **D**〕
4. Only positive values are valid for **S**. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, **S** should be converted into a floating point value.
5. e^{D} = **S** .The content in **D** = ln**S**; **S** = designated source data.
6. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
7. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
8. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) into binary floating point and store it in register (D11, D10).
2. When M1= On, use register (D11, D10) as the real number for LN operation and store the binary floating point result in register (D21, D20).
3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × $10^{[D31]}$) and store it in register (D31, D30).



Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands	Function	Controllers
126	D	LOG	P	(S1) (S2) (D)	Logarithm of Binary Floating Point	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DLOG, DLOGP: 13 steps
S_1					*	*							*			
S_2					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Device for base **S_2**: Device for operation source **D**: Device for operation result

Explanations:

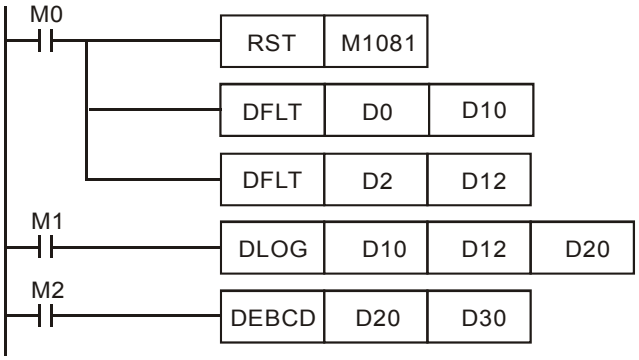
1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. This instruction performs "log" operation of the content in **S_1** and **S_2** and stores the result in **D**.
4. Only positives are valid for the content in **S_1** and **S_2**. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, **S_1** and **S_2** should be converted into floating point values.
5. $\mathbf{S_1}^{D} = \mathbf{S_2}$, **D** = ? → $\mathrm{Log}_{S_1}{}^{S_2} = \mathbf{D}$

   Example: Assume $\mathbf{S_1} = 5$, $\mathbf{S_2} = 125$, $\mathbf{D} = \log_5{}^{125} = ?$

   $\mathbf{S_1}^{D} = \mathbf{S_2} \rightarrow 5^{D} = 125 \rightarrow \mathbf{D} = \log_5{}^{125} = 3$
6. If the absolute value of the result $>$ maximum floating point available, the carry flag M1022 = On.
7. If the absolute value of the result $<$ minimum floating point available, the borrow flag M1021 = On.
8. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) and (D3, D2) into binary floating points and store them in the 32-bit registers (D11, D10) and (D13, D12).
2. When M1= On, perform log operation on the binary floting points in 32-bit registers (D11, D10) and (D13, D12) and store the result in the 32-bit register (D21, D20).
3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × $10^{[D31]}$) and store it in register (D31, D30).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
127	D	ESQR	P	(S) (D)	Floating Point Square Root	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DESQR, DESQRP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

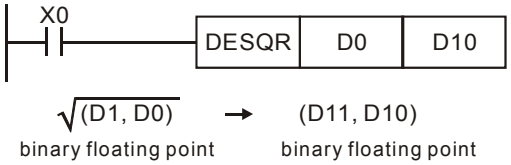
S: Source device **D**: Operation result

Explanations:

1. Range of **S**: ≥ 0
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag); M1067 (operation error)
4. This instruction performs a square root operation on the content in the register designated by **S** and stores the result in the register designated by **D**. The square root operation is performed in floating point system.
5. If **S** is an designated constant K or H, the instruction will convert the constant into a binary floating point value before the operation.
6. If the result of the operation = 0, the zero flag M1020 = On.
7. **S** can only be a positive value. Performing any square root operation on a negative value will result in an "operation error" and this instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code H'0E1B.

Program Example 1:

When M0 = On, calculate the square root of the binary floating point (D1, D0) and store the result in register (D11, D10).



Program Example 2:

When M2 = On, calculate the square root of K1,234 (automatically converted into binary floating point) and store the result in register (D11, D10).

Remarks:

For floating point operations, see "5.3 Handling of Numeric Values".

API	Mnemonic			Operands	Function	Controllers
128	D	POW	P	(S1) (S2) (D)	Floating Point Power Operation	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DPOW, DPOWP: 13 steps
S_2					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

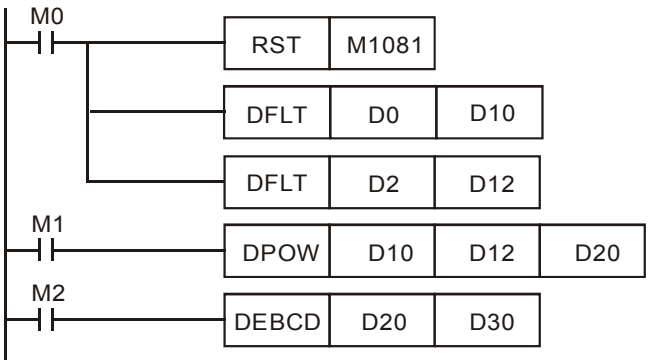
S_1: Device for base. **S_2**: Device for exponent. **D**: Device for operation result

Explanations:

1. See the specifications of each model for their range of use.
2. This instruction performs power multiplication of binary floating point **S_1** and **S_2** and stores the result in **D**.
   D = POW〔**S_1** + 1, **S_1**〕^〔**S_2** + 1, **S_2**〕
3. Only positives are valid for the content in **S_1**. Both positives and negatives are valid for the content in **S_2**. When designating **D** registers, the data should be 32-bit and the operation should be performed in floating point system. Therefore, **S_1** and **S_2** should be converted into floating point values.
   Example: When $\mathbf{S_1}^{S2}$ = **D**, **D** = ?
   Assume **S_1** = 5, **S_2** = 3, **D** = 5^3 =125
4. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
5. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
6. If the result = 0, the zero flag M1020 = On.

Program Example:

1. When M0 = On, convert (D1, D0) and (D3, D2) into binary floating points and store them in the 32-bit registers (D11, D10) and (D13, D12).
2. When M1= On, perform POW operation on the binary floting points in 32-bit registers (D11, D10) and (D13, D12) and store the result in the 32-bit register (D21, D20).
3. When M2 = On, convert the binary floating point (D21, D20) into decimal floating point (D30 × $10^{[D31]}$) and store it in register (D31, D30).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers	
129	D	INT	P	(S) (D)	Float to Integer	ES/EX/SS	SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			INT, INTP: 5 steps DINT, DINTP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

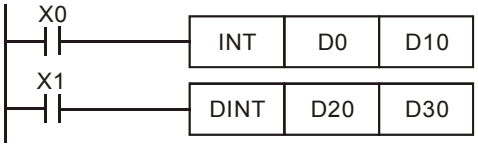
S: Source device **D**: Converted result

Explanations:

1. **S** occupies 2 consecutive devices. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. The binary floating point value of the register designated by **S** is converted to BIN integer and stored in the register designated by **D**. The decimal of BIN integer is left out.
4. This instruction is the inverse operation of API 49 FLT instruction.
5. If the converstion result = 0, the zero flag M1020 = On
   If there is any decimal left out, the borrow flag M1021 = On.
   If the result exceeds the range listed below, the carry flag M1022 = On.
   16-bit instruction: -32,768 ~ 32,767
   32-bit instruction: -2,147,483,648 ~ 2,147,483,647

Program Example:

1. When X0 = On, the binary floating point (D1, D0) will be converted into BIN integer and the result will be stored in (D10). The decimal of BIN integer will be left out.
2. When X1 = On, the binary floating point (D21, D20) will be converted into BIN integer and the result will be stored in (D31, D30). The decimal of BIN integer will be left out.



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers	
130	D	SIN	P	(S) (D)	Sine	ES/EX/SS	SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DSIN, DSINP: 9 steps
D													*			

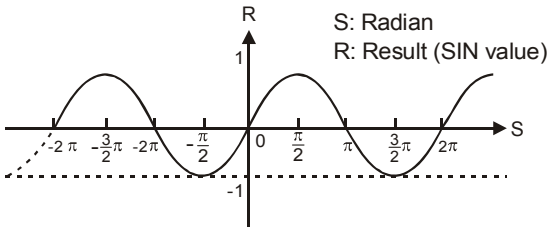
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value **D**: SIN result

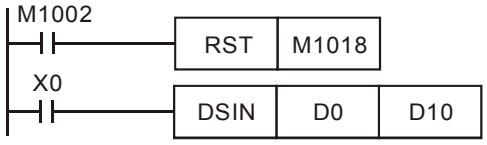
Explanations:

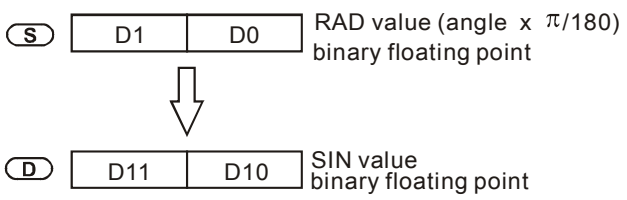
1. 0° ≤ **S** < 360°. See the specifications of each model for their range of use.
2. Flags: M1018 (angle or radian); M1020 (zero flag)
3. **S** can be an angle or radian, decided by M1018.
4. When M1018 = Off, the program will be in radian mode and the RAD value = angle × π /180
5. When M1018 = On, the program will be in angle mode and the range of angle should be "0° ≤ angle < 360°"
6. If the result = On, M1020 = On.
7. The SIN value obtained by **S** is calculated and stored in the register designated by **D**. The figure below offers the relation between radian and the result.



Program Example 1:

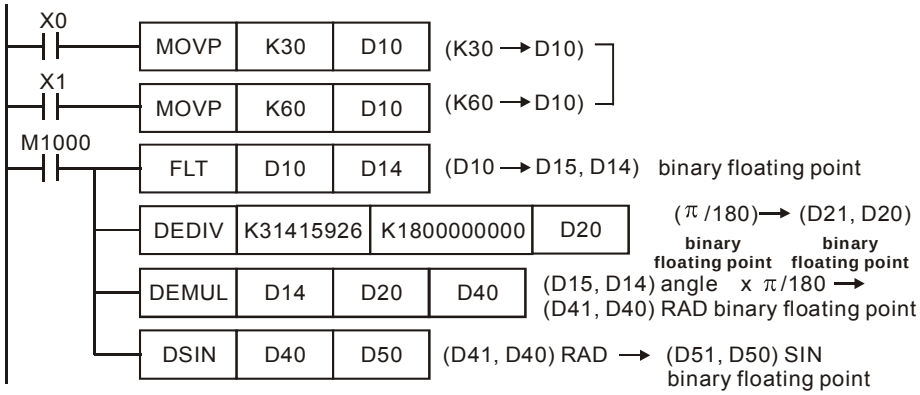
When M1018 = Off, the program is in radian mode. When X0 = On, use the RAD value of binary floating point (D1, D0) and obtain its SIN value. The binary floating point result will be stored in (D11, D10).





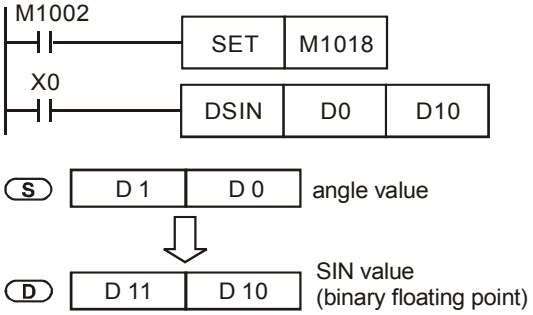
Program Example 2:

When M1018 = Off, the program is in radian mode. Input terminals X0 and X1 select the angle. The angles are converted into RAD value for calculating the SIN value.



Program Example 3:

When M1018 = On, the program is in angle mode. When X0 = On, use the angle of (D1, D0) to obtain SIN value and store the binary floating point result in (D11, D10). (0° ≤ angle < 360°)



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers	
131	D	COS	P	(S) (D)	Cosine	ES/EX/SS	SA/SX/SC / EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DCOS, DCOSP: 9 steps
D													*			

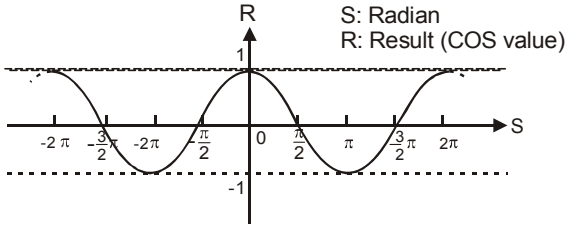
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value **D**: COS result

Explanations:

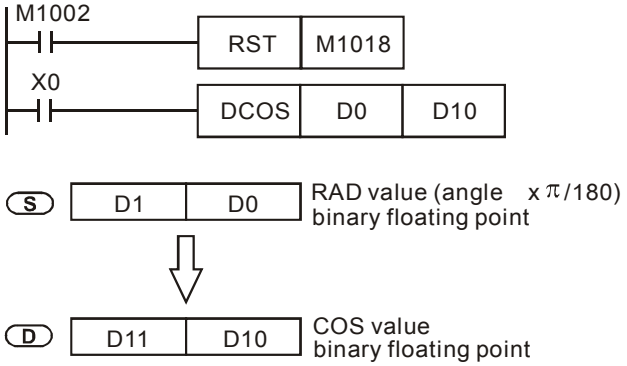
1. 0° ≤ **S** < 360°. See the specifications of each model for their range of use.
2. Flags: M1018 (angle or radian); M1020 (zero flag)
3. **S** can be an angle or radian, decided by M1018.
4. When M1018 = Off, the program will be in radian mode and the RAD value = angle × π /180
5. When M1018 = On, the program will be in angle mode and the range of angle should be "0° ≤ angle < 360°"
6. If the result = On, M1020 = On.
7. The COS value obtained by **S** is calculated and stored in the register designated by **D**. The figure below offers the relation between radian and the result.



8. Switch between radian and angle by M1018: When M1018 = Off, **S** will be a RAD value; when M1018 = On, **S** will be an angle (0° ~ 360°).

Program Example 1:

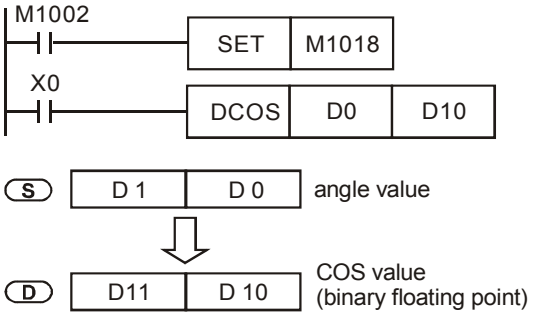
When M1002 = Off, the program is in radian mode. When X0 = On, use the RAD value of binary floating point (D1, D0) and obtain its COS value. The binary floating point result will be stored in (D11, D10).



Program Example 2:

When M1018 = On, the program is in angle mode. When X0 = On, use the angle of (D1, D0) to obtain COS value and

store the binary floating point result in (D11, D10). (0° ≤ angle < 360°)



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
132	D	TAN	P	(S) (D)	Tangent	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DTAN, DTANP: 9 steps
S					*	*							*			
D													*			

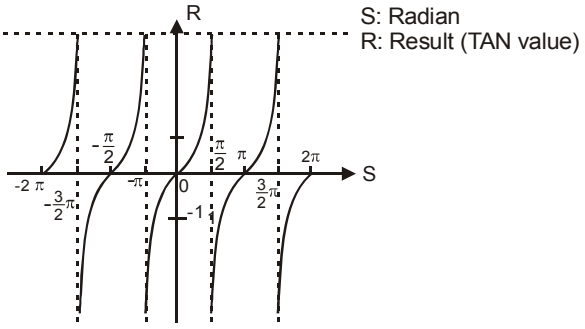
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value **D**: TAN result

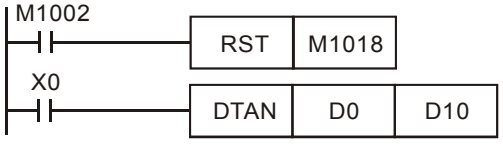
Explanations:

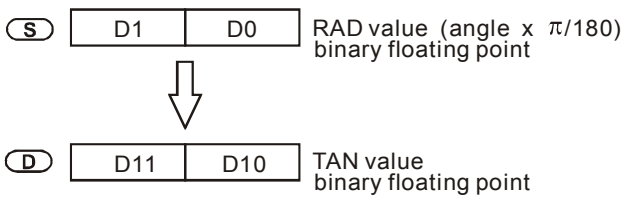
1. 0° ≤ **S** < 360°. See the specifications of each model for their range of use.
2. Flags: M1018 (angle or radian); M1020 (zero flag)
3. **S** can be an angle or radian, decided by M1018.
4. When M1018 = Off, the program will be in radian mode and the RAD value = angle × π /180
5. When M1018 = On, the program will be in angle mode and the range of angle should be "0° ≤ angle < 360°"
6. If the result = On, M1020 = On.
7. The TAN value obtained by **S** is calculated and stored in the register designated by **D**. The figure below offers the relation between radian and the result.



Program Example 1:

When M1018 = Off, the program is in radian mode. When X0 = On, use the RAD value of binary floating point (D1, D0) and obtain its TAN value. The binary floating point result will be stored in (D11, D10).

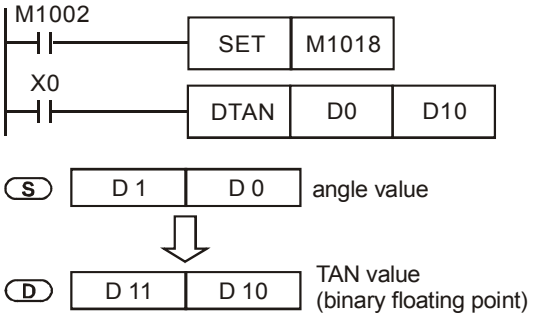




Program Example 2:

When M1018 = On, the program is in angle mode. When X0 = On, use the angle of (D1, D0) to obtain TAN value and

store the binary floating point result in (D11, D10). (0° ≤ angle < 360°)



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
133	D	ASIN	P	(S) (D)	Arc Sine	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DASIN, DASINP: 9 steps
D													*			

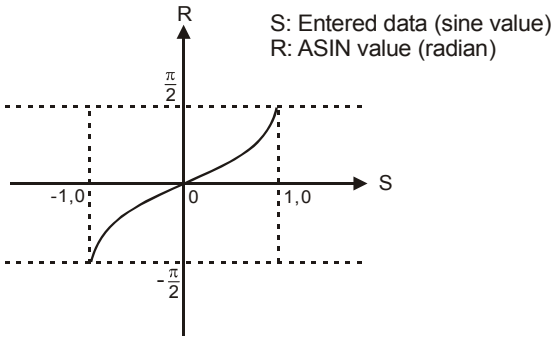
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value (binary floating point) **D**: ASIN result

Explanations:

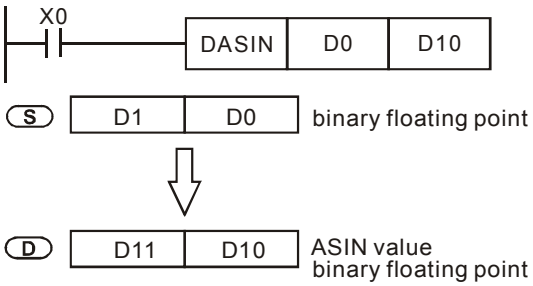
1. See the specifications of each model for their range of use.
2. Flag: M1020 (zero flag)
3. ASIN value=$\sin^{-1}$. The figure below offers the relation between the entered sin value and the result.



4. The decimal floating point of the SIN value designated by **S** should be within -1.0 ~ +1.0. If the value falls without the range, M1067 and M1068 will be On without performing any action.
5. If the result = 0, M1020 = On.

Program Example:

When X0 = On, obtain the ASIN value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
134	D	ACOS	P	(S) (D)	Arc Cosine	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DACOS, DACOSP: 9 steps
D													*			

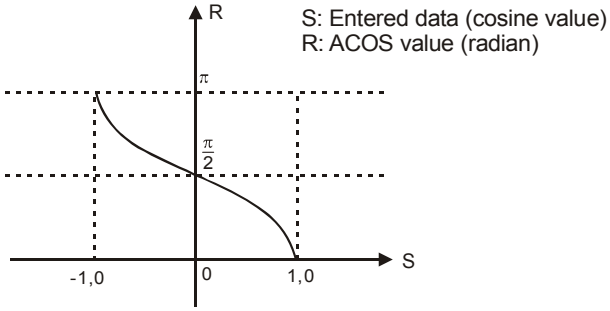
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value (binary floating point) **D**: ACOS result

Explanations:

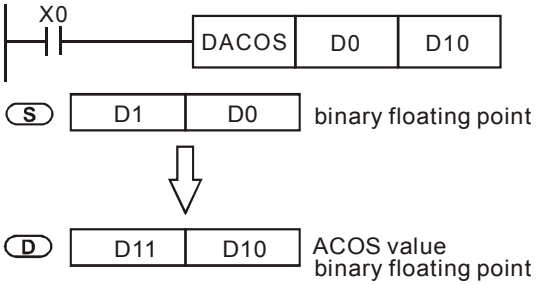
1. See the specifications of each model for their range of use.
2. Flag: M1020 (zero flag)
3. ACOS value=$\cos^{-1}$. The figure below offers the relation between the entered cos value and the result.



4. The decimal floating point of the COS value designated by **S** should be within -1.0 ~ +1.0. If the value falls without the range, M1067 and M1068 will be On without performing any action.
5. If the result = 0, M1020 = On.

Program Example:

When X0 = On, obtain the ACOS value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
135	D	ATAN	P	(S) (D)	Arc Tangent	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DATAN, DATANP: 9 steps
S					*	*							*			
D													*			

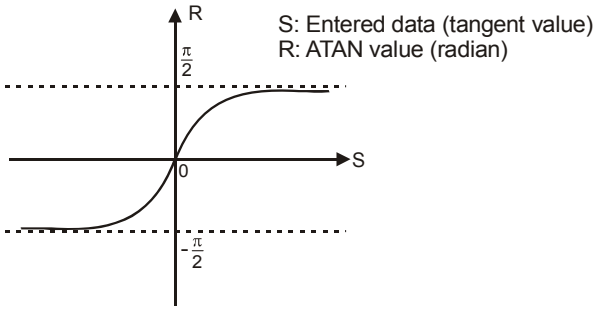
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Source value (binary floating point) **D**: ATAN value

Explanations:

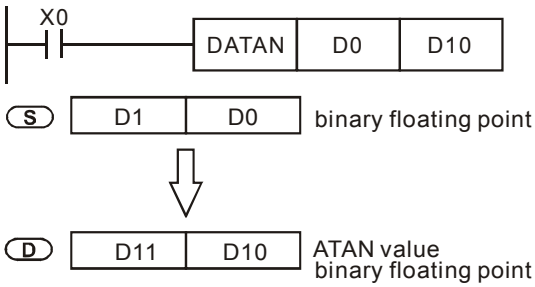
1. See the specifications of each model for their range of use.
2. Flag: M1020 (zero flag)
3. ATAN value=$\tan^{-1}$. The figure below offers the relation between the entered tan value and the result.



4. If the result =0, M1020 = On.

Program Example:

When X0 = On, obtain the ATAN value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).



Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
136	D	SINH	P	(S) (D)	Hyperbolic Sine	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DSINH, DSINHP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

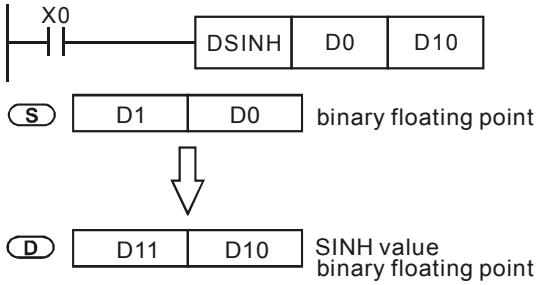
S: Source value (binary floating point) **D**: SINH value

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. SINH value=$(e^{s}-e^{-s})/2$. The result is stored in **D**.

Program Example:

1. When X0 = On, obtain the SINH value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).



2. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
3. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
4. If the result = 0, the zero flag M1020 = On.

Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic	Operands	Function	Controllers
137	D COSH P	(S) (D)	Hyperbolic Cosine	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DCOSH, DCOSHP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

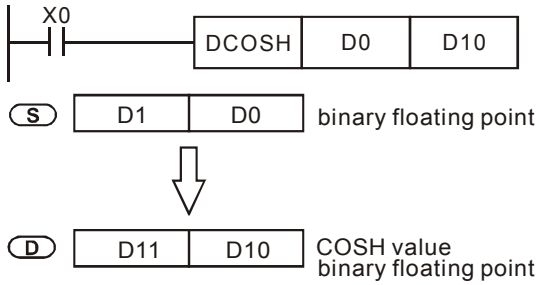
S: Source value (binary floating point) **D**: COSH value

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. COSH value=$(e^{s}+e^{-s})/2$. The result is stored in **D**.

Program Example:

1. When X0 = On, obtain the COSH value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).



5. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
6. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
7. If the result = 0, the zero flag M1020 = On.

Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic			Operands	Function	Controllers
138	D	TANH	P	(S) (D)	Hyperbolic Tangent	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DTANH, DTANHP: 9 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

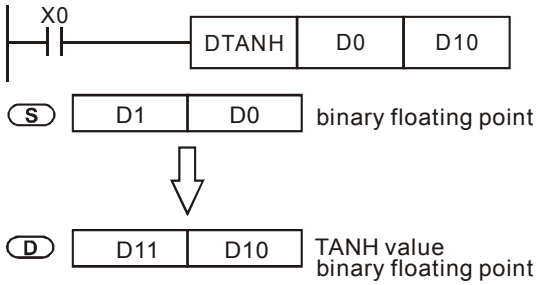
S: Source value (binary floating point) **D**: TANH result

Explanations:

1. See the specifications of each model for their range of use.
2. Flags: M1020 (zero flag); M1021 (borrow flag); M1022 (carry flag)
3. TANH value=$(e^{s}-e^{-s})/(e^{s}+e^{-s})$. The result is stored in **D**.

Program Example:

1. When X0 = On, obtain the TANH value of binary floating point (D1, D0) and store the binary floating point result in (D11, D10).



2. If the absolute value of the result ＞ maximum floating point available, the carry flag M1022 = On.
3. If the absolute value of the result ＜ minimum floating point available, the borrow flag M1021 = On.
4. If the result = 0, the zero flag M1020 = On.

Remarks:

For floating point operations, see “5.3 Handling of Numeric Values”.

API	Mnemonic	Operands	Function	Controllers
143	DELAY	S	Delay Instruction	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*							*			DELAY, DELAYP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

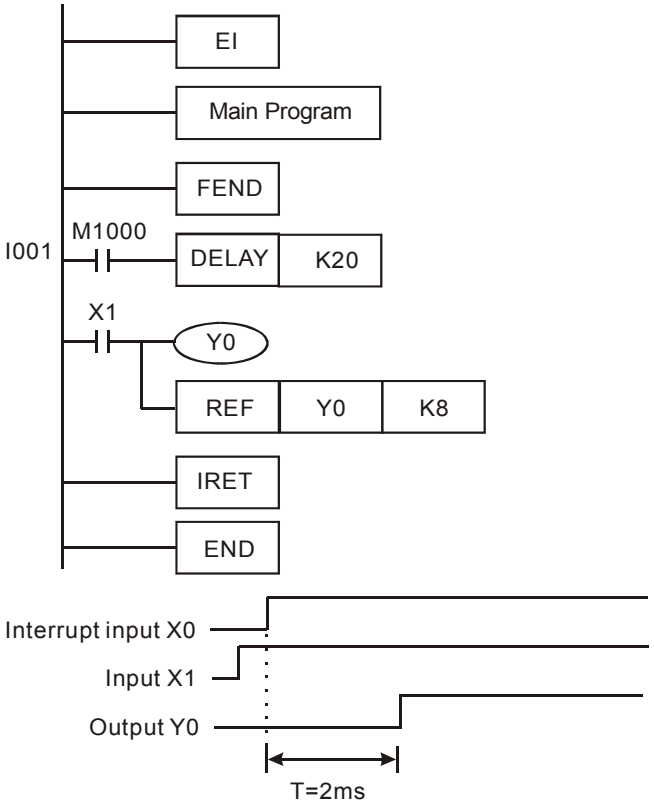
S: delay time (unit: 0.1ms)

Explanations:

1. Range of **S**: K1 ~ K1,000. See the specifications of each model for their range of use.
2. After DELAY instruction is executed, the program after DELAY in every scan period will execute delay outputs according to the delay time designated by the user.

Program Example:

When X0 goes from Off to On and generates an external interruption, the interruption subroutine will execute DELAY for 2ms before executing the next step, X1 = On and Y0 = On.



Remarks:

1. User can define the delay time based on their needs.
2. The delay time may increase due do the influences from communication, high-speed counters and high-speed pulse output instructions.
3. The delay time of designated external output (transistor or relay) will increase due to the delay on the transistor

or relay itself. See 2.3 for more information.

API	Mnemonic	Operands	Function	Controllers
144	GPWM	(S1) (S2) (D)	General PWM Output	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1													*			GPWM: 7 steps
S_2													*			
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

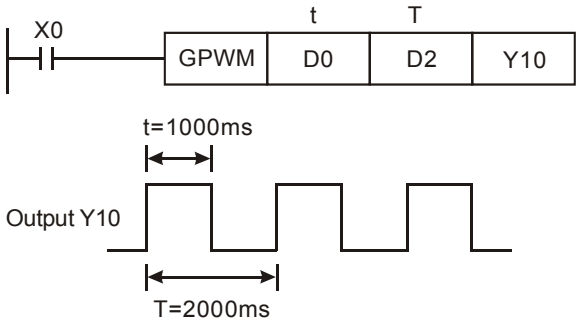
S_1: Width of output pulse **S_2**: Pulse output cycle **D**: Pulse output device

Explanations:

1. **S_2** occupies 3 consecutive devices.
2. **S_1 ≤ S_2**. See the specifications of each model for their range of use.
3. Range of **S_1**: t = 0 ~ 32,767ms.
4. Range of **S_2**: t = 1 ~ 32,767ms.
5. **S_2** +1 and **S_2** +2 are parameters for the system. Do not occupy them.
6. Pulse output devices **D**: Y, M, S.
7. When being executed, GPWM instruction designates **S_1** and **S_2** and that pulses output will be from device **D**.
8. When **S_1** ≤ 0, there will be no pulse output. When **S_1** ≥ **S_2**, the pulse output device will keep being On.
9. **S_1** and **S_2** can be modified when GPWM instruction is being executed.

Program Example:

When X0 = On, D0 = K1,000, D2 = K2,000, and Y10 will output the pulse illustrated below. When X0 = Off, Y10 output will be Off.



Explanations:

1. This instruction counts by the scan cycle; therefore the maximum offset will be one PLC scan cycle. **S_1**, **S_2** and (**S_2** - **S_1**) should > PLC scan cycle; otherwise, errors will occur during GPWM outputs.
2. Please note that placing this instruction in a subroutine or interruption will cause inaccurate GPWM outputs.

API	Mnemonic	Operands	Function	Controllers
145	FTC	(S1) (S2) (S3) (D)	Fuzzy Temperature Control	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			FTC: 9 steps
S_2					*	*							*			
S_3													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

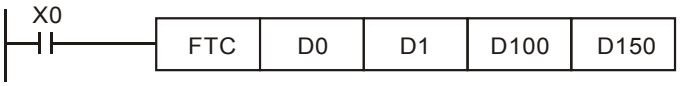
S_1: Set value (SV) **S_2**: Present value (PV) **S_3**: Parameter (sampling time) **D**: Output value (MV)

Explanations:

1. Range of **S_1** : 1 ~ 5000 (shown as 0.1°C ~ 500°C). Unit: 0.1°. If (**S_3** +1) is set as K0, the range will be 0.1°C ~ 500°C.
2. Range of **S_2** : 1 ~ 5000 (shown as 0.1°C ~ 500°C). Unit: 0.1°. If (**S_3** +1) is set as bit0 = 0, the range will be 0.1°C ~ 500°C. Therefore, when the user obtain an A/D value from the temperature sensor, the value has to be converted into a value between 1 ~ 5,000 by four arithmetic operation instructions.
3. If **S_3** < K1, the instruction will not be executed. If **S_3** > K200, S3 will adopt K200. **S_3** will occupy 7 consecutive devices.
4. See the specifications of each model for their range of use.
5. Settings of parameter **S_3** +1: bit0 = 0 ->°C; bit1 = 0 ->°F; bit1 = 0 -> no filter function; bit1 = 1 -> with filter function; bit2 ~ bit5 -> 4 kinds of heating environments; bit6 ~ bit15 -> reserved. See remarks for more information.
6. **D** is the value between 0 ~ sampling time × 100. When using this instruction, the user has to adopt other instructions according to the types of the heater. For example, FTC can be used with GPWM for output pulse control. “Sampling time × 100” is the cycle of GPWM pulse output; MV is the width of GPWM pulse. See program example 1.
7. There is no limit on the times of using FTC instruction, but Do not repeatedly use a designated operand in case an error may occur..

Program Example:

1. Set up the parameter before executing FTC instruction.
2. When X0 = On, the instruction will be executed and and result will be stored in D150. When X0 = Off, the instruction will not be executed and the previous data remain unchanged.

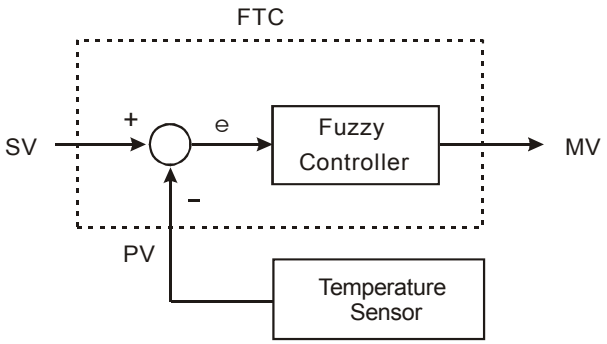


Remarks:

1. Setting of S_3:

Device No.	Function	Range	Explanation
S_3	Sampling time (T_S) (unit: 100ms)	1 ~ 200 (unit: 100ms)	If T_S is less than a scan time, PID instruction will be executed for a scan time. If T_S= 0, PID instruction will not be enabled. The minimum T_S must be greater than a scan time.
S_3 +1	b0: temperature unit b1: filter function b2 ~ b5: heating environnment b6 ~ b15: reserved	b0 =0 means °C b0 =1 means °F	When the value exceeds the upper bound, use upper bound.
		b1=0 means without fileter function b1=1 means with filter function	When without filter function, PV = currently measured value. When with filter function, PV = (currently measured value + previous PV)/2
		b2=1	Slow heating environment
		b3=1	General heating environment
		b4=1	Fast heating environment
		b5=1	High-speed heating environment
S_3 +2 ~ S_3 +6	Parameters for system use only. Do not use them.		

2. Control Diagram:



3. Notes and suggestion:

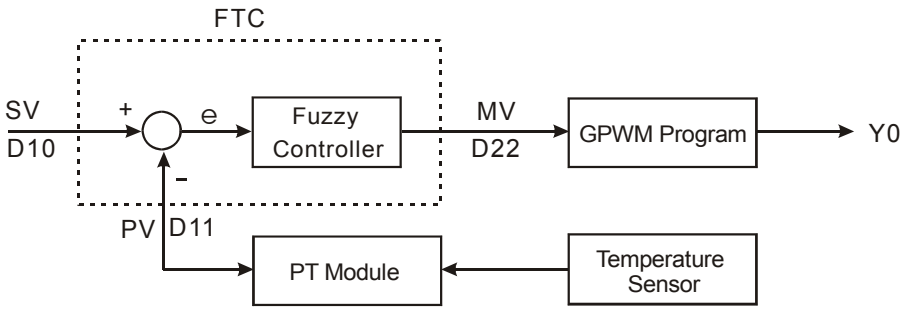
   It is recommended that the sampling time be set to 2 times more than the sampling time of the temperature sensor for better temperature control.

   bit2 ~ bit5 of S_3+1 are for the control speed. If the user does not set up the parameter, FTC will automatically activate “general heating environment”. When the user finds that the control is too slow to reach SV, select “slow

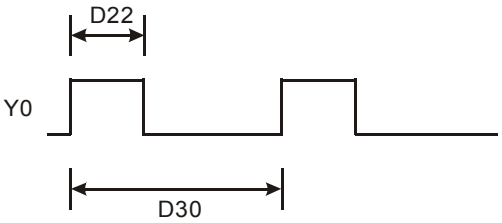
heating environment" to enhance the speed to reach SV. On the contrary, when the user finds that the control is too fast or with too many fluctuations, select "fast heating environment" to slow down the control speed.

When bit2 ~ bit5 of S_3+1 are all set as 1 or more than 1 environments are designated, FTC instruction will check from bit2 to bit 5 in order and enable the function that has been set as 1. The parameter can be modified during the control.

4. **Example 1:** control diagram

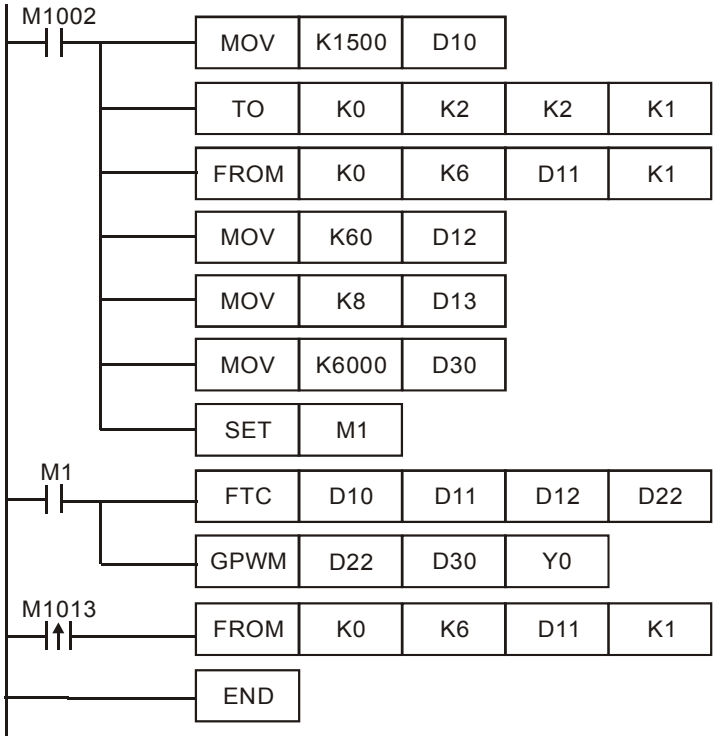


Output D22 (MV) of FTC instruction is the input D22 of GPWM instruction, as the duty cycle of ajustable pulses. D30 is the fixed cycle time of pulses. See below for the timing diagram of Y0 output.

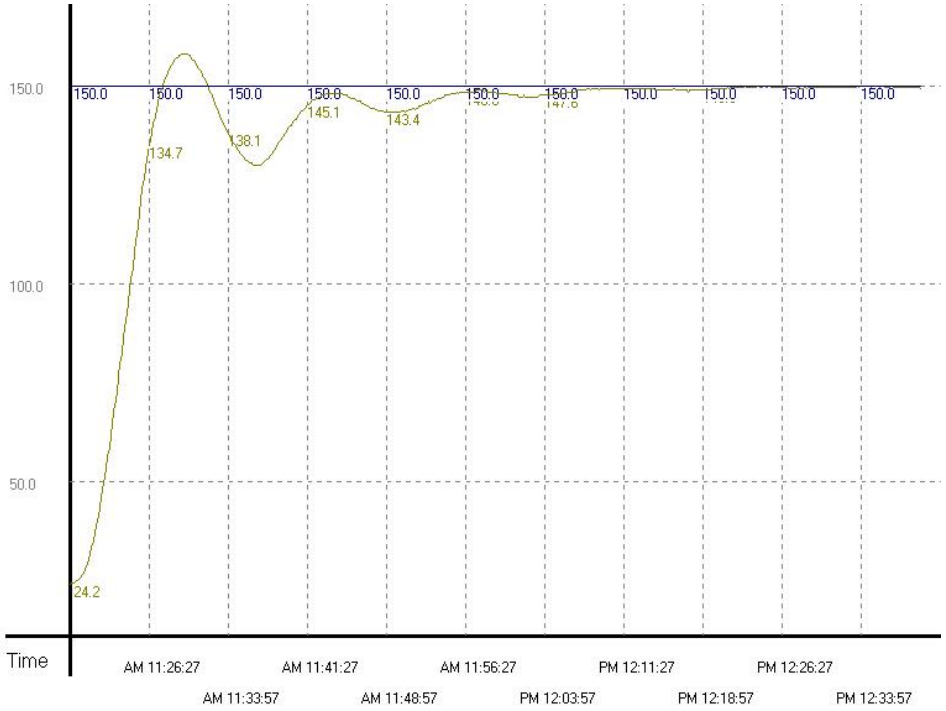


Assume parameter settings: D10 = K1,500 (target temperature), D12 = K60 (sampling time: 6 secs.), D13 = K8 (bit3=1), D30 = K6,000 (=D12*100)

The example control program is indicated as:

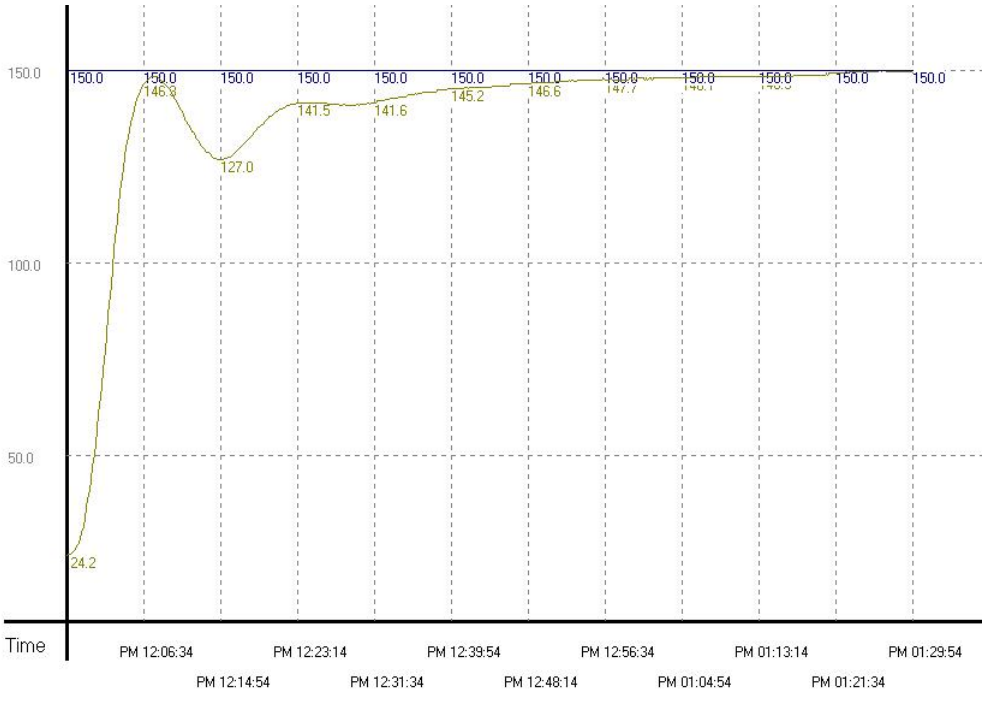


Experiment in an oven which can be heated up to 250°C. See below for the records of target and present temperatures. As shown in the diagram below, we can see that after 48 minutes, the temperature is able to reach the target temperature with ±1°C inaccuracy and exceed approx. 10°C of the target temperature.



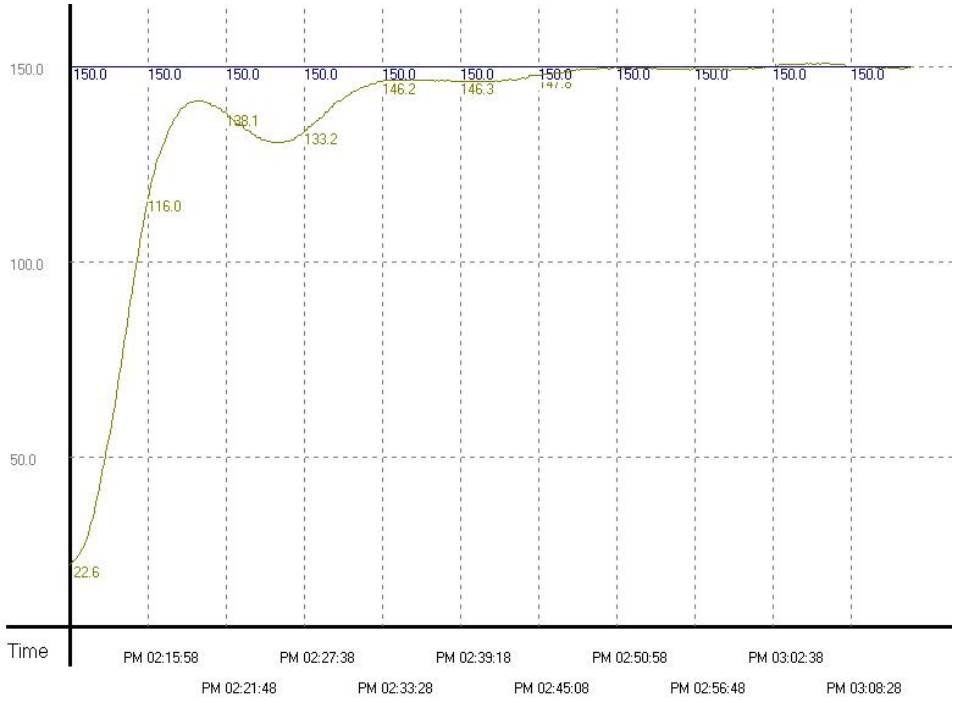
Example 2: Due to that the temperature once exceeds the target temperature, we modify the heating environment into “fast heating environment” (D13 = K16). The results are shown in the diagram below.

From the diagram below, we see that though the temperature no longer exceeds the target temperature, it still needs to take more than 1 hour and 15 minutes to reach the target temperature with ±1°C inaccuracy. It seems that we have chosen the right environment, but the sampling time is too long, resulting in the extension of heating time.



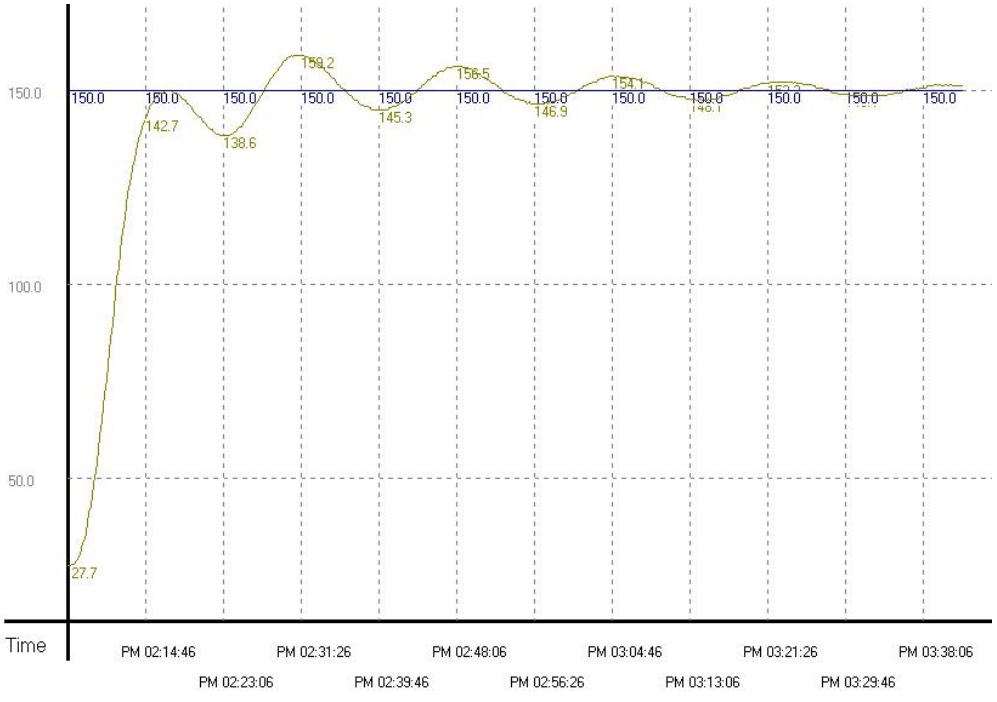
Example 3: To speed up the speed to reach the target temperature, we correct the sampling time as 4 seconds (D12 = K40, D30 = K4,000). The results are shown in the diagram below.

From the diagram below, we see that the overall control time has been shortened as 37 minutes. Therefore, we find out that modifying the sampling time can speed up the time for reaching the target temperature.



Example 4: To see if we can reach the target temperature faster, we modify the sampling time frim example 3 into 2 seconds (D12 = K20, D30 = K2,000). The results are shown in the diagram below.

From the diagram below, we see that the sampling time that is too short will cause the control system to become too sensitive and lead to up and down fluctuations.



API	Mnemonic	Operands	Function	Controllers
146	CVM	(S1) (S2) (D)	Valve Control	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1													*			CVM: 7 steps
S_2					*	*							*			
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

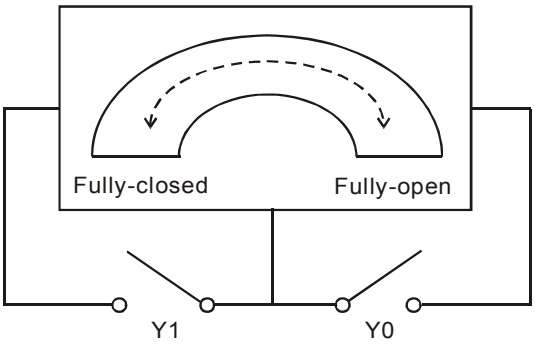
S_1: Target time of valve (absolute position) **S_2**: Time from fully-closed to fully-open of valve (destination) **D**: Output device

Explanations:

1. **S_1** occupies 3 consecutive registers when in use. **S_1** + 0 are for the user to store the designated value; **S_1** + 1 (the current position of the valve) and **S_1** + 2 are for storing the parameters recorded in the instruction and please DO NOT use and alter these two registers.
2. **D** occupies 2 consecutive output devices when in use. **D** + 0 is the "open" contact and **D** + 1 is the "close" contact.
3. This instruction only supports EH2/SV and does not support EH.
4. The unit of time: 0.1 second. When the scan time of the program exceeds 0.1 second, DO NOT use this instruction to adjust the position of the valve.
5. Frequency of the output device: 10Hz.
6. When the time of **S_1** + 0 $>$ the fully-opened time set in **S_2**, **D** + 0 will keep being On and **D** + 1 being Off. When the time of **S_1** + 0 $<$ 0, **D** + 0 will keep being Off and **D** + 1 being On.
7. When the instruction is enabled, the instruction will start to control the valve from "0" time position. Therefore, if the user cannot be sure whether the valve is at "0" before executing the instruction, please designate **S_1** + 0 as less than 0 and execute the instruction for **S_2** (time) before sending in the correct target control time.

Program Example 1:

1. The control valve



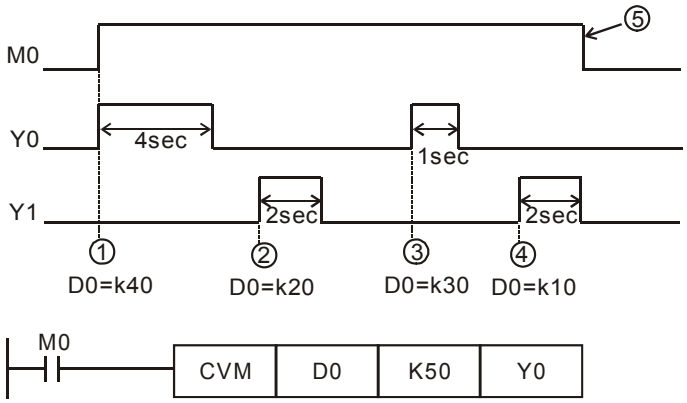
2. Definitions of the control valve:
   a) When Y0 and Y1 = Off: No valve action

b) When Y0 = On and Y1 = Off: Valve “open”

c) When Y0 = Off and Y1 = On: Valve “closed”

d) When Y0 and Y1 = On: The action is prohibited.

3. Timing diagram and program of the control:



4. Control phases:

1) Phase ①: When M0 = On, D0 = K40 refers to the valve shall be open (Y0 = On, Y1 = Off) till the position of 4 seconds.

2) Phase ②: Change the position of the valve and D0 = K20. Due to that the previous position was at 4 seconds, the valve shall be closed (Y0 = Off, Y1 = On) for 2 seconds, moving the valve to the position of 2 seconds.

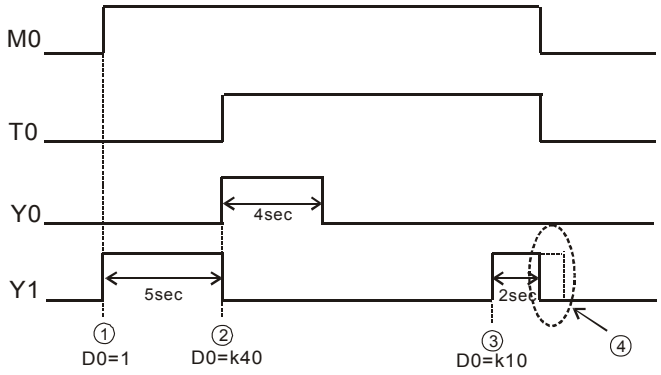
3) Phase ③: Change the position of the valve and D0 = K30. Due to that the previous position was at 2 seconds, the valve shall be open (Y0 = On, Y1 = Off) for 1 second, moving the valve to the position of 3 seconds.

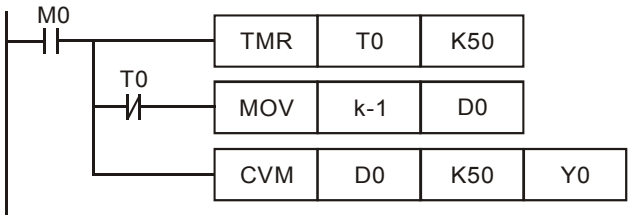
4) Phase ④: Change the position of the valve and D0 = K10. Due to that the previous position was at 2 seconds, the valve shall be closed (Y0 = Off, Y1 = On) for 2 seconds, moving the valve to the position of 1 second.

5) Phase ⑤: Switch off X0 and no actions at the valve (Y0 = Off, Y1 = Off).

Program Example 2:

1. Timing diagram and program of the control:





2. Control phases:
   1) Phase ①: When M0 = On, due to that we are not sure about there the valve is, set D0 = K-1 to deliberately close the valve (Y0 = Off, Y1 = On) for 5 seconds and make sure the valve is at the position of 0 second before moving on to the next step.
   2) Phase ②: When T0 = On, allow D0 = K40 to start is action. Open the valve (Y0 = On, Y1 = Off) for 4 seconds, moving the valve to the position of 4 seconds.
   3) Phase ③: Change the position of the valve and D0 = K10. Due to that the previous position was at 4 seconds, the valve shall be closed (Y0 = Off, Y1 = On) for 3 seconds, moving the valve to the position of 1 second.
   4) Phase ④: Switch off M0 and the valve will no longer move (Y0 = Off, Y1 = Off).

API	Mnemonic			Operands	Function	Controllers	
147	D	SWAP	P	(S)	Byte Swap	ES/EX/SS	SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S								*	*	*	*	*	*	*	*	SWAP, SWAPP: 3 steps DSWAP, DSWAPP: 5 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

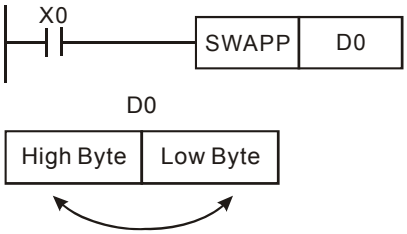
S: Device for swapping 8 high/low byte.

Explanations:

1. If **D** is used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. As 16-bit instruction: the contents in the 8 high bytes and 8 low bytes are swapped.
4. As 32-bit instruction: the 8 high bytes and 8 low bytes in the two registers swap with each other respectively.
5. This instruction adopts pulse execution instructions (SWAPP, DSWAPP).

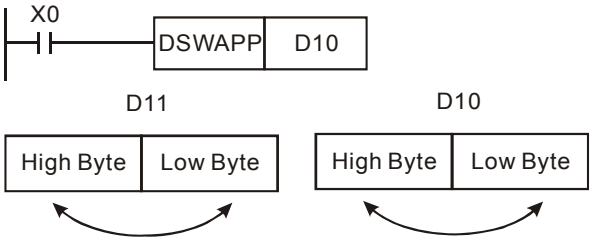
Program Example 1:

When X0 = On, the high 8 bytes and low 8 bytes in D0 will swap with each other.



Program Example 2:

When X0 = On, the high 8 bytes and low 8 bytes in D11 will swap with each other and the high 8 bytes and low 8 bytes in D10 will swap with each other.



API	Mnemonic	Operands	Function	Controllers
148	D MEMR P	(m) (D) (n)	Read File Register	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
m					*	*							*			MEMR, MEMRP: 7 steps
D													*			DMEMR, DMEMRP: 13 steps
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

m: Address in the file register to be read **D**: Device for storing the read data (starting from the designated D)

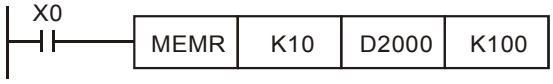
n: Number of data read at a time

Explanations:

1. Range of **m**: K0 ~ K1,599 (SA/SX/SC); K0 ~ K9,999 (EH/EH2/SV)
2. Range of **D**: D2000 ~ D4999 (SA/SX/SC); D2000 ~ D9999 (EH/EH2/SV)
3. Range of **n**: For 16-bit instruction K1 ~ K1,600 (SA/SX/SC), K1 ~ K8,000 (EH/EH2/SV); For 32-bit instruction K1 ~ K800 (SA/SX/SC), K1 ~ K4,000 (EH/EH2/SV)
4. See the specifications of each model for their range of use.
5. Flag: M1101. See explanations below.
6. SA/SX/SC/EH/EH2/SV uses this instruction to read the data in file registers and store them into data registers.
7. SA/SX/SC offers 1,600 16-bit file registers.
8. **m** and **n** of SA/SX/SC do not suppot E and F index register modification.
9. EH/EH2/SV offers 10,000 16-bit file registers.
10. If **m**, **D** and **n** fall without their range, operation error will occur. M1067, M1068 = On and D1067 will record the error code H'0E1A.

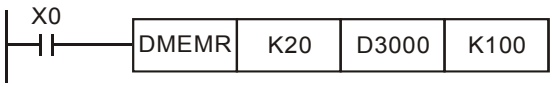
Program Example 1:

1. The 16-bit instruction MEMR reads 100 data at address 10 in the file register and store the read data in register D starting from D2000.
2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the previously read data will remain unchanged.



Program Example 2:

1. The 32-bit instruction DMEMR reads 100 data at address 20 in the file register and store the read data in register D starting from D3000.
2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the previously read data will remain unchanged.



API	Mnemonic			Operands	Function	Controllers	
149	D	MEMW	P	(S) (m) (n)	Write File Register	ES/EX/SS	SA/SX/SC / EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			MEMW, MEMWP: 7 steps
m					*	*							*			DMEMW, DMEMWP: 13 steps
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Device for storing the written data (starting from the designated D) **m**: Address in the file register to be written

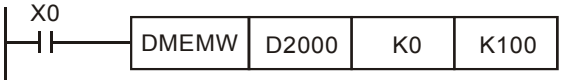
n: Number of data to be written at a time

Explanations:

1. Range of **S**: D2000 ~ D4999 (SA/SX/SC); D2000 ~ D9999 (EH/EH2/SV)
2. Range of **m**: K0 ~ K1,599 (SA/SX/SC); K0 ~ K9,999 (EH/EH2/SV)
3. Range of **n**: For 16-bit instruction K1 ~ K1,600 (SA/SX/SC), K1 ~ K8,000 (EH/EH2/SV); For 32-bit instruction K1 ~ K800 (SA/SX/SC), K1 ~ K4,000 (EH/EH2/SV)
4. See the specifications of each model for their range of use.
5. Flag: M1101. See explanations below.
6. SA/SX/SC/EH/EH2/SV uses this instruction to read the data in data registers and write them into file registers.
7. SA/SX/SC offers 1,600 16-bit file registers.
8. **m** and **n** of SA/SX/SC do not suppot E and F index register modification.
9. EH/EH2/SV offers 10,000 16-bit file registers.
10. If **S**, **m** and **n** fall without their range, operation error will occur. M1067, M1068 = On and D1067 will record the error code H'0E1A.

Program Example:

1. When X0 = On, the 32-bit instruction DMEMW writes 100 32-bit data starting from D2001 and D2000 into address 0 ~ 199 in the file register.
2. When X0 = On, the instruction will be executed. When X0 = Off, the instruction will not be executed and the previously data written in will remain unchanged.



File Register:

1. EH/EH2/SV: When the PLC is powered, it will decide whether to automatically send the data in the file register to the designated data register by M1101 (whether to enable the function of file register), D1101 (start address in file register K0 ~ K9,999), D1102 (number of data to be read in file register K1 ~ k8,000), and D1103 (device for storing read data, starting from designated D, K2,000 ~ K9,999).
2. In EH/EH2/SV, the reading of data from file register to data register D will not be executed if D1101 < 0, D1101

> K9,999, D1103 < K2,000 or D1103 > K9,999.

3. SA/SX/SC: When the PLC is powered, it will decide whether to automatically send the data in the file register to the designated data register by M1101 (whether to enable the function of file register), D1101 (start address in file register K0 ~ K1,599), D1102 (number of data to be read in file register K1 ~ k1,600), and D1103 (device for storing read data, starting from designated D, K2,000 ~ K4,999).
4. In SA/SX/SC, the reading of data from file register to data register D will not be executed if D1101 < 0, D1101 > K1,599, D1103 < K2,000 or D1103 > K4,999.
5. When the reading of data from file register to data register D starts, PLC will stop the reading if the address of file register or data register exceed their range.
6. In PLC program, only API 148 MEMR and API 149 MEMW can be used to read or write the file register. See 2.8.3 for more information on file registers.
7. File registers do not have actual addresses in it. Reading and writing of file registers can only be done through API 148 MEMR, API 149 MEMW or peripheral devices HPP and WPLSoft.
8. If the address in the file register to be read exceeds its range, the read value will be 0.
9. Special relays of file register and other relevant special registers:

Flag	Function
M1101	Whether to enable the function of file register; latched; default = off

Special D	Function
D1101	Start address in file register. SA/SX/SC: K0 ~ K1,599; EH/EH2/SV: K0 ~ K9,999; latched; default = 0
D1102	Number of data to be read in file register. SA/SX/SC: K1 ~ K1,600; EH/EH2/SV: K1 ~ K8,000; latched; default = 0
D1103	Device for storing read data, starting from designated D. SA/SX/SC: K2,000 ~ K4,999; EH/EH2/SV: K2,000 ~ K9,999; latched; default = 2,000

MEMO

API	Mnemonic	Operands	Function	Controllers
150	MODRW	(S1) (S2) (S3) (S) (n)	Read/Write MODBUS Data	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			MODRW: 11 steps
S_2					*	*							*			
S_3					*	*							*			
S													*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Address of communication device **S_2**: Function code **S_3**: Device address of data to be read/written

S: Register for storing read/written data (source or destination) **n**: Length of read/written data

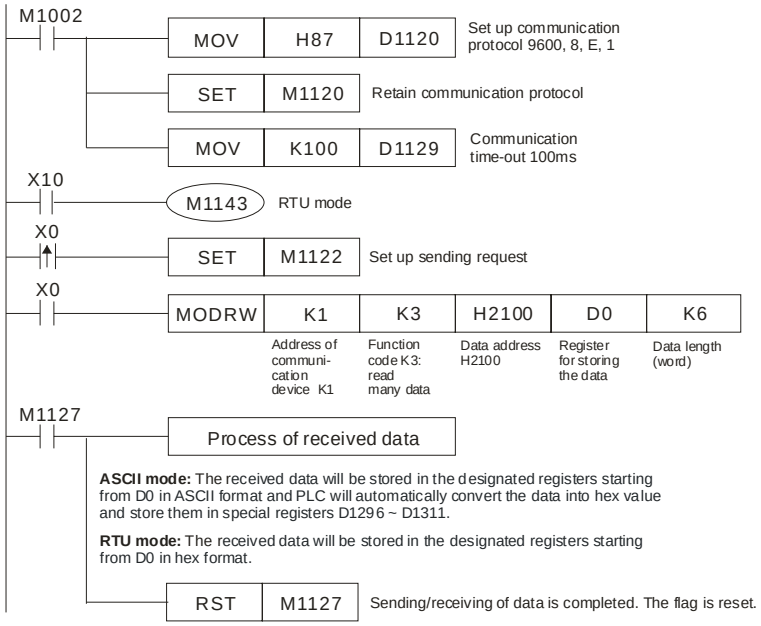
Explanations:

1. The content of **S_2** shall only be: K3(H3), K6(H6), K16(H10).
2. ES/EX/SS V.4.9 (and above) support the continuous execution instruction (MODRW). Other versions do not support this instruction.
3. ES/EX/SS series MPU does not support E, F index register modification.
4. Flags: M1120 ~ M1131, M1140 ~ M1143. See remarks for more details.
5. Range of **S_1**: K0 ~ K254
6. **S_2**: Funcation code. For example, H03 is for AC motor drive or DVP-PLC to read many data; H06 is for AC motor drive or DVP-PLC to write a single data; H10 is for AC motor drive or DVP-PLC to write many data. Only these function codes are available currently; other function codes are still not executable. See program examples for more information.
7. **S_3**: Device address of data to be read/written. The device address inside the communication device. If the address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.
8. **S**: Register for storing read/written data. The user sets up a register and stores the data to be written in the register in advance. The register can be register for storing the read data.
9. **n**: Length of read/written data. For ES/SA, when M1143 = Off (in ASCII mode), the range of length: K1 ~ K8 (word); when M1143 = On (in RTU mode), the range of length: K1 ~ K16 (word). For EH/EH2/SV, the range of length: K1 ~ K16 (word).
10. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example 1:

1. Function code K3(H3): For reading many data in register
   When PLC is connected to VFD-S AC motor drive: M1143 = Off, in ASCII mode
   When PLC is connected to VFD-S AC motor drive: M1143 = On, in RTU mode

2. When in ASCII mode, the received data will be stored in the designated registers starting from D0 in ASCII format and PLC will automatically convert the data into hex value and store them in special registers D1296 ~ D1311. When the conversion into hex value starts, M1131 will be On and turn Off when the conversion is completed.
3. If necessary, the user can move the hex values stored in D1296 ~ D1131 to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1296 ~ D1311.
4. When in RTU mode, the received data will be stored in the designated registers starting from D0 in hex format.
5. When In ASCII mode or RTU mode, PLC will store the data to be sent in D1256 ~ D1295. If necessary, the user can move the data to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1256 ~ D1295.
6. The data sent back from AC motor drive are stored in the registers designated by the user. After the transmission is completed, PLC will auto-check if the received data are incorrect. M1140 will be On if there is an error.
7. If the device address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.
8. After M1140 = On or M1141 = On, PLC will send another correct datum to AC motor drive. If the data sent back from AC motor drive is correct, M1140 and M1141 will be reset.



9. ASCII Mode: When PLC is connected to VFD-S AC motor drive.

   PLC ⇨ VFD-S, PLC sends: **"01 03 2100 0006 D5"**

   VFD-S ⇨ PLC, PLC receives: **"01 03 0C 0100 1766 0000 0000 0136 0000 3B"**

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1256 Low	‘0’	30 H	ADR 1 ADR 0	Address of AC motor drive: ADR (1,0)
D1256 High	‘1’	31 H		
D1257 Low	‘0’	30 H	CMD 1 CMD 0	Instruction code: CMD (1,0)
D1257 High	‘3’	33 H		
D1258 Low	‘2’	32 H	Starting Data Address	
D1258 High	‘1’	31 H		
D1259 Low	‘0’	30 H		
D1259 High	‘0’	30 H		
D1260 Low	‘0’	30 H	Number of Data (counted by words)	
D1260 High	‘0’	30 H		
D1261 Low	‘0’	30 H		
D1261 High	‘6’	36 H		
D1262 Low	‘D’	44 H	LRC CHK 1 LRC CHK 0	Error checksum: LRC CHK (0,1)
D1262 High	‘5’	35 H		

Registers for received data D0 (responding messages)

Register	DATA		Explanation	
D0 Low	‘0’	30 H	ADR 1 ADR 0	
D0 High	‘1’	31 H		
D1 Low	‘0’	30 H	CMD 1 CMD 0	
D1 High	‘3’	33 H		
D2 Low	‘0’	30 H	Number of Data (counted by byte)	
D2 High	‘C’	43 H		
D3 Low	‘0’	30 H	Content of address 2100H	PLC automatically convert ASCII codes to numerals and store the numeral in D1296 = H0100
D3 High	‘1’	31 H		
D4 Low	‘0’	30 H		
D4 High	‘0’	30 H		
D5 Low	‘1’	31 H	Content of address 2101H	PLC automatically convert ASCII codes to numerals and store the numeral in D1297 = H1766
D5 High	‘7’	37 H		
D6 Low	‘6’	36 H		
D6 High	‘6’	36 H		
D7 Low	‘0’	30 H	Content of address 2102H	PLC automatically convert ASCII codes to numerals and store the numeral in D1298 = H0000
D7 High	‘0’	30 H		
D8 Low	‘0’	30 H		
D8 High	‘0’	30 H		
D9 Low	‘0’	30 H	Content of address 2103H	PLC automatically convert ASCII codes to numerals and store the numeral in D1299 = H0000
D9 High	‘0’	30 H		
D10 Low	‘0’	30 H		
D10 High	‘0’	30 H		
D11 Low	‘0’	30 H	Content of address 2104H	PLC automatically convert ASCII codes to numerals and store the numeral in D1300 = H0136
D11 High	‘1’	31 H		
D12 Low	‘3’	33 H		
D12 High	‘6’	36 H		
D13 Low	‘0’	30 H	Content of address 2105H	PLC automatically convert ASCII codes to numerals and store the numeral in D1301 = H0000
D13 High	‘0’	30 H		
D14 Low	‘0’	30 H		
D14 High	‘0’	30 H		
D15 Low	‘3’	33 H	LRC CHK 1 LRC CHK 0	
D15 High	‘B’	42 H		

10. RTU Mode: When PLC is connected to VFD-S AC motor drive

    PLC ⇨ VFD-S, PLC sends: **"01 03 2100 0006 CF F4"**

    VFD-S ⇨ PLC, PLC receives: **"01 03 0C 0000 0503 0BB8 0BB8 0000 012D 8E C5"**

Registers for sent data (sending messages)

Register	DATA	Explanation
D1256 Low	01 H	Address
D1257 Low	03 H	Function
D1258 Low	21 H	Starting Data Address
D1259 Low	00 H	
D1260 Low	00 H	Number of Data (counted by words)
D1261 Low	06 H	
D1262 Low	CF H	CRC CHK Low
D1263 Low	F4 H	CRC CHK High

Registers for received data D0 (responding messages)

Register	DATA	Explanation	
D0 Low	01 H	Address	
D1 Low	03 H	Function	
D2 Low	0C H	Number of Data (byte)	
D3 Low	00 H	Content of address 2100H	PLC automatically convert ASCII codes to numerals and store the numeral in D1296 = H0000
D4 Low	00 H		
D5 Low	05 H	Content of address 2101H	PLC automatically convert ASCII codes to numerals and store the numeral in D1297 = H0503
D6 Low	03 H		
D7 Low	0B H	Content of address 2102H	PLC automatically convert ASCII codes to numerals and store the numeral in D1298 = H0BB8
D8 Low	B8 H		
D9 Low	0B H	Content of address 2103H	PLC automatically convert ASCII codes to numerals and store the numeral in D1299 = H0BB8
D10 Low	B8 H		
D11 Low	00 H	Content of address 2104H	PLC automatically convert ASCII codes to numerals and store the numeral in D1300 = H0000
D12 Low	00 H		
D13 Low	01 H	Content of address 2105H	PLC automatically convert ASCII codes to numerals and store the numeral in D1301 = H012D
D14 Low	2D H		
D15 Low	8E H	CRC CHK Low	
D16 Low	C5 H	CRC CHK High	

Program Example 2:

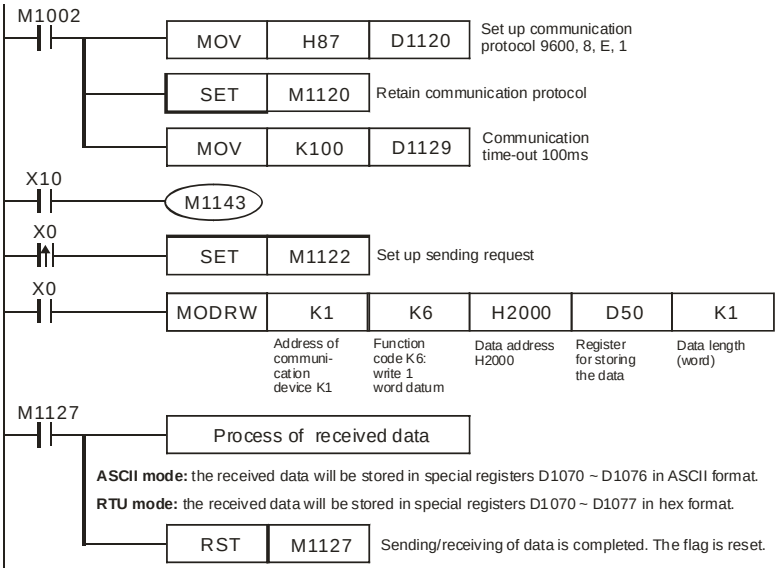
1. Function code K6(H6): For writing a word data to register

   When PLC is connected to VFD-S AC motor drive: M1143 = Off, in ASCII mode

   When PLC is connected to VFD-S AC motor drive: M1143 = On, in RTU mode

2. When in ASCII mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1076.
3. When in RTU mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1077.
4. When In ASCII mode or RTU mode, PLC will store the data to be sent in D1256 ~ D1295. If necessary, the user can move the data to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1256 ~ D1295.

5. After receiving the data sent back from AC motor drive is completed, PLC will auto-check if the received data are incorrect. M1140 will be On if there is an error.
6. If the device address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.
7. After M1140 = On or M1141 = On, PLC will send another correct datum to AC motor drive. If the data sent back from AC motor drive is correct, M1140 and M1141 will be reset.



8. ASCII Mode: When PLC is connected to VFD-S AC motor drive.

   PLC ⇨ VFD-S, PLC sends: **"01 06 0100 1770 71"**

   VFD-S ⇨ PLC, PLC receives: **"01 06 0100 1770 71"**

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1256 Low	'0'	30 H	ADR 1 ADR 0	Address of AC motor drive: ADR (1,0)
D1256 High	'1'	31 H		
D1257 Low	'0'	30 H	CMD 1 CMD 0	Instruction code: CMD (1,0)
D1257 High	'6'	36 H		
D1258 Low	'0'	30 H	Data Address	
D1258 High	'1'	31 H		
D1259 Low	'0'	30 H		
D1259 High	'0'	30 H		
D1260 Low	'1'	31 H	Data content	The content of register D50 (H1770 = K6,000)
D1260 High	'7'	37 H		
D1261 Low	'7'	37 H		
D1261 High	'0'	30 H		
D1262 Low	'7'	37 H	LRC CHK 1 LRC CHK 0	LRC CHK (0,1) is error check
D1262 High	'1'	31 H		

Registers for received data (responding messages)

Register	DATA		Explanation
D1070 Low	‘0’	30 H	ADR 1
D1070 High	‘1’	31 H	ADR 0
D1071 Low	‘0’	30 H	CMD 1
D1071 High	‘6’	36 H	CMD 0
D1072 Low	‘0’	30 H	Data Address
D1072 High	‘1’	31 H	
D1073 Low	‘0’	30 H	
D1073 High	‘0’	30 H	
D1074 Low	‘1’	31 H	Data content
D1074 High	‘7’	37 H	
D1075 Low	‘7’	37 H	
D1075 High	‘0’	30 H	
D1076 Low	‘7’	37 H	LRC CHK 1
D1076 High	‘1’	31 H	LRC CHK 0

9. RTU Mode: When PLC is connected to VFD-S AC motor drive

   PLC ⇨ VFD-S, PLC sends: **“01 06 2000 0012 02 07”**

   VFD-S ⇨ PLC, PLC receives: **“01 06 2000 0012 02 07”**

Registers for sent data (sending message)

Register	DATA	Explanation	
D1256 Low	01 H	Address	
D1257 Low	06 H	Function	
D1258 Low	20 H	Data Address	
D1259 Low	00 H		
D1260 Low	00 H	Data content	The content of register D50 (H12)
D1261 Low	12 H		
D1262 Low	02 H	CRC CHK Low	
D1263 Low	07 H	CRC CHK High	

Registers for received data (responding message)

Register	DATA	Explanation
D1070 Low	01 H	Address
D1071 Low	06 H	Function
D1072 Low	20 H	Data Address
D1073 Low	00 H	
D1074 Low	00 H	Data content
D1075 Low	12 H	
D1076 Low	02 H	CRC CHK Low
D1077 Low	07 H	CRC CHK High

Program Example 3:

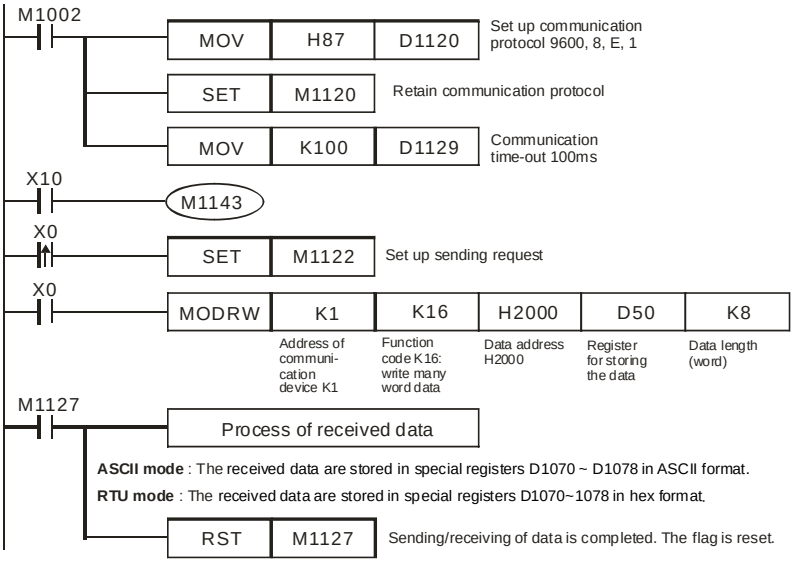
1. Function code K16(H10): For writing many word data into register.

   When PLC is connected to VFD-S AC motor drive: M1143 = Off, in ASCII mode

   When PLC is connected to VFD-S AC motor drive: M1143 = On, in RTU mode

2. When in ASCII mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1076.

3. When in RTU mode, the user stores the data to be written in the designated register D50 in hex format. The data sent back from AC motor drive are stored in D1070 ~ D1077.
4. When In ASCII mode or RTU mode, PLC will store the data to be sent in D1256 ~ D1295. If necessary, the user can move the data to other general registers by using MOV, DMOV or BMOV instruction. Other instructions of ES/EX/SS do not function on the data in D1256 ~ D1295.
5. After receiving the data sent back from AC motor drive is completed, PLC will auto-check if the received data are incorrect. M1140 will be On if there is an error.
6. If the device address is illegal to a designated communication device, the communication device will respond with an error message and PLC will store the error code in D1130 and M1141 = On. For example, if 8000H is illegal to VFD-S, M1141 will be On and D1130 = 2. See user manual of VFD-S for error codes.
7. After M1140 = On or M1141 = On, PLC will send another correct datum to AC motor drive. If the data sent back from AC motor drive is correct, M1140 and M1141 will be reset.



8. ASCII Mode: When PLC is connected to VFD-S AC motor drive.
   PLC ⇨ VFD-S, PLC sends: "**01 10 2000 0002 04 0012 1770 30**"
   VFD-S ⇨ PLC, PLC receives: "**01 10 2000 0002 CD**"

Registers for sent data (sending messages)

Register	DATA		Explanation	
D1256 Low	'0'	30 H	ADR 1	Address of AC motor drive: ADR (1,0)
D1256 High	'1'	31 H	ADR 0	
D1257 Low	'1'	31 H	CMD 1	Instruction code: CMD (1,0)
D1257 High	'0'	30 H	CMD 0	
D1258 Low	'2'	32 H	Data Address	
D1258 High	'0'	30 H		
D1259 Low	'0'	30 H		
D1259 High	'0'	30 H		
D1260 Low	'0'	30 H	Number of Registers	
D1260 High	'0'	30 H		
D1261 Low	'0'	30 H		
D1261 High	'2'	32 H		

Register	DATA		Explanation	
D1262 Low	‘0’	30 H	Byte Count	
D1262 High	‘4’	34 H		
D1263 Low	‘0’	30 H	Data contents 1	The content of register D50 (H12)
D1263 High	‘0’	30 H		
D1264 Low	‘1’	31 H		
D1264 High	‘2’	32 H		
D1265 Low	‘1’	31 H	Data contents 2	The content of register D51 (H1770 = K6,000)
D1265 High	‘7’	37 H		
D1266 Low	‘7’	37 H		
D1266 High	‘0’	30 H		
D1267 Low	‘3’	33 H	LRC CHK 1	Error checksum: LRC CHK (0,1)
D1267 High	‘0’	30 H	LRC CHK 0	

Registers for received data (responding messages)

Register	DATA		Explanation
D1070 Low	‘0’	30 H	ADR 1
D1070 High	‘1’	31 H	ADR 0
D1071 Low	‘1’	31 H	CMD 1
D1071 High	‘0’	30 H	CMD 0
D1072 Low	‘2’	32 H	Data Address
D1072 High	‘0’	30 H	
D1073 Low	‘0’	30 H	
D1073 High	‘0’	30 H	
D1074 Low	‘0’	30 H	Number of Registers
D1074 High	‘0’	30 H	
D1075 Low	‘0’	30 H	
D1075 High	‘2’	32 H	
D1076 Low	‘C’	43 H	LRC CHK 1
D1076 High	‘D’	44 H	LRC CHK 0

9. RTU Mode: When PLC is connected to VFD-S AC motor drives

PLC ⇨ VFD-S, PLC sends: “**01 10 2000 0002 04 0012 1770 C4 7F**”

VFD-S ⇨ PLC, PLC receives: “**01 10 2000 0002 4A 08**”

Registers for send data (sending messages)

Register	DATA	Explanation	
D1256 Low	01 H	Address	
D1257 Low	10 H	Function	
D1258 Low	20 H	Data Address	
D1259 Low	00 H		
D1260 Low	00 H	Number of Registers	
D1261 Low	02 H		
D1262 Low	04 H	Byte Count	
D1263 Low	00 H	Data content 1	The content of register D50 (H12)
D1264 Low	12 H		
D1265 Low	17 H	Data content 2	The content of register D51 (H1770 = K6,000)
D1266 Low	70 H		
D1267 Low	C4 H	CRC CHK Low	
D1268 Low	7F H	CRC CHK High	

Registers for received data (responding messages)

Register	DATA	Explanation
D1070 Low	01 H	Address
D1071 Low	10 H	Function
D1072 Low	20 H	Data Address
D1073 Low	00 H	
D1074 Low	00 H	Number of Registers
D1075 Low	02 H	
D1076 Low	4A H	CRC CHK Low
D1077 Low	08 H	CRC CHK High

Remarks:

1. The activation condition placed before MODRD, RDST and MODRW instructions cannot use rising-edge or falling-edge contacts; otherwise the data stored in the registers for received data will encounter errors.
2. Flags and special registers for MODRW instruction in RS-485 communication. (For details, see API 80 RS).

Flags	Function
M1120	For retaining communication setups. After the setup is made, changes in D1120 will be invalid.
M1121	When Off, RS-485 is sending data.
M1122	Sending request
M1123	Receiving is completed
M1124	Waiting for receiving data
M1125	Disable receiving status
M1126	Selecting STX/ETX system
M1127	Sending/receiving data through MODRD / RDST / MODRW instructions is completed.
M1128	Sending data…/receiving data…
M1129	Receiving data time-out
M1130	User/system defined STX/ETX
M1131	On when MODRD / MODWR / MODRW is converting data to hex
M1140	MODRD / MODWR / MODRW data receiving error
M1141	MODRD / MODWR / MODRW parameter error
M1142	VFD-A handy instruction data receiving error
M1143	ASCII/RTU mode selection (used with MODRD/MODWR/MODRW) (Off = ASCII mode; On = RTU mode)
D1070 ~ D1085	When the built-in RS-485 communication instruction is executed and sends out data, the receiving end will respond with a message and the message will be stored in D1070 ~ D1085. The user can check the registers for the messages.
D1120	RS-485 communication protocol
D1121	PLC communication address (saving PLC communication address; latched)
D1122	Remaining words of the sent data
D1123	Remaining words of the received data
D1124	Start text definition (STX)
D1125	Definition of end text 1 (ETX1)
D1126	Definition of end text 2 (ETX2)
D1129	Abnormal communication time-out. Unit: ms
D1130	Records of error codes sent back from MODBUS
D1256 ~ D1295	When the built-in RS-485 communication instruction MODRW is executed, the sent out data will be stored in D1256 ~ D1295. The user can check whether the instruction is correct by the contents in the registers.
D1296 ~ D1311	PLC will automatically convert the ASCII data stored in the register designated by the user into hex format.

API	Mnemonic	Operands	Function	Controllers
151	PWD	(S) (D)	Detection of Input Pulse Width	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*															PWD: 5 steps
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

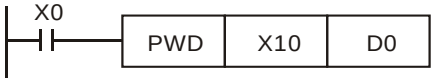
S: Source device **D**: Destination device for storing the detected result

Explanations:

1. Range of **S**: X10 ~ X17
2. Range of **D**: D0 ~ D999, occupying 2 consecutive devices. Can only be used once in the program.
3. PWD instruction is for detecting the time span of output signals from X10 ~ X17; the valid frequency range is 1 ~1KHz. When M1169 = On, the instruction will detect the time span of the continuous rising edge and falling edge of the input signals (time unit: 100us). When M1169 = On, the instruction will detect the time span of 2 continuous rising edges of the input signals (time unit: 1us). It cannot designate the same X10 ~ X17 as does DCNT and ZRN instructions.
4. **D** occupies two continuous devices. The longest detectable time is 21,474.83647 seconds, about 357.9139 minutes or 5.9652 hours.
5. There is no limitation on the times of using this instruction. However, only one instruction can be executed at a time.

Program Example:

When X0 = On, record the time span of X10 = On and store it in D1 and D0.



API	Mnemonic	Operands	Function	Controllers
152	RTMU	(D) (n)	Start of the Measurement of Execution Time of I Interruption	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D					*	*							*			RTMU: 5 steps
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: Device for storing the measuring time (unit: 1us) **n**: Measurement time base. Parameter range: K10 ~ K500 (time unit: 1us)

Explanations:

1. Range of **D**: K0 ~ K9
2. Range of **n**: K10 ~ K500
3. The designated special D registers (D1156 ~ D1165) can measure up to 10 interruption subroutines. For example, when **D** = K5, the designated D register will be D1161.
4. When RTMU is executed, if the **D** and **n** entered by the user are legal, interruption of the timer will be enabled and the counting starts and the special D designated by **D** is cleared as 0. When RTMD is executed, interruption of the timer is disabled and the calculated time will be assigned to special D designated by RTMD.
5. With API 153 RTMD, RTMU can measure the execution time of "I" interruption service subroutine, which can be reference for dealing with the high-speed response when the user is at the initial stage of developing the program.

API	Mnemonic	Operands	Function	Controllers
153	RTMD	(D)	End of the Measurement of the Execution Time of I Interruption	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D					*	*							*			RTMD: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

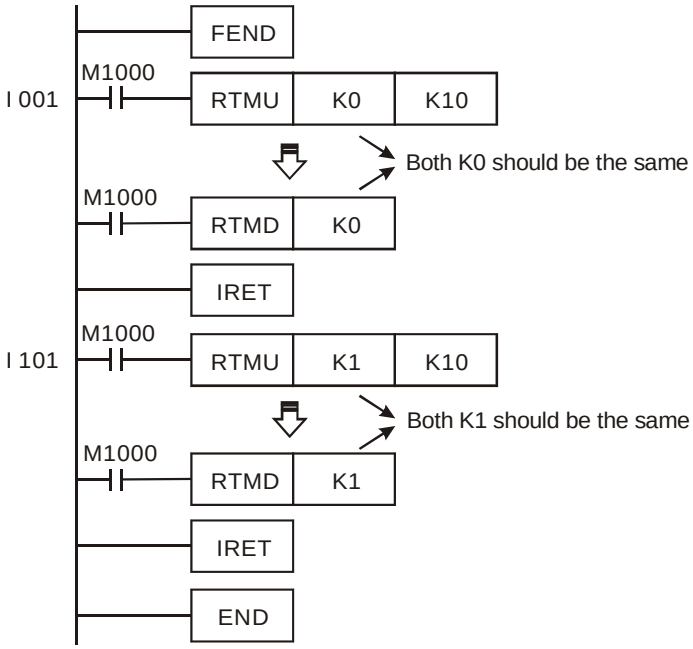
D: Device to store the measuring time (unit: 1us).

Explanations:

1. Range of **D**: K0 ~ K9. The No. of **D** has to be the same as that designated by **D** in API 152 RTMU; otherwise the result of the measurement may be unexpectable.

Program Example:

When X0 goes from Off to On, the program will enter I001 interruption subroutine. RTMU will activate an 8-bit timer (unit: 10us) and RTMD (when D = K0) will shut down the timer and store the time in the timer in special D registers (D1156 ~ D1165, designated by K0 ~ K9).



Remarks:

1. We suggest you remove this instruction after you finish developing your PLC program.
2. Due to the lower priority of the interruption enabled by RTMU, when RTMU is enabled, other high-speed pulse input counting or high-speed pulse output may result in failure to trigger the timer.
3. If you activate RTMU but do not activate RTMD before the end of the interruption, the interruption will not be shut down.
4. RTMU instruction activates 1 timer interruption in PLC. Therefore, if many RTMU or RTMD are executed at the same time, confusion in the timer may occur. Please be aware of the situation.

API	Mnemonic		Operands	Function	Controllers
154	RAND	P	(S1) (S2) (D)	Random Number	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	RAND, RANDP: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

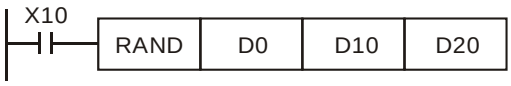
S_1: Lower bound of the random number **S_2**: Upper bound of the random number **D**: The random number produced

Explanations:

1. **$S_1 \leqq S_2$**; K0 $\leqq$ S_1 , $S_2 \leqq$K32,767
2. See the specifications of each model for their range of use.
3. Entering $S_1 > S_2$ will result in operation error. The instruction will not be executed at this time, M1067, M1068 = On and D1067 records the error code 0E1A (hex).

Program Example:

When X10 = On, RAND will produce the random number between the lower bound D0 and upper bound D10 and store the result in D20.



API	Mnemonic	Operands	Function	Controllers
155	D ABSR	S D1 D2	Read the Absolute Position from a Servo Motor	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S	*	*	*	*												DABSR: 13 steps
D_1		*	*	*												
D_2								*	*	*	*	*	*	*		

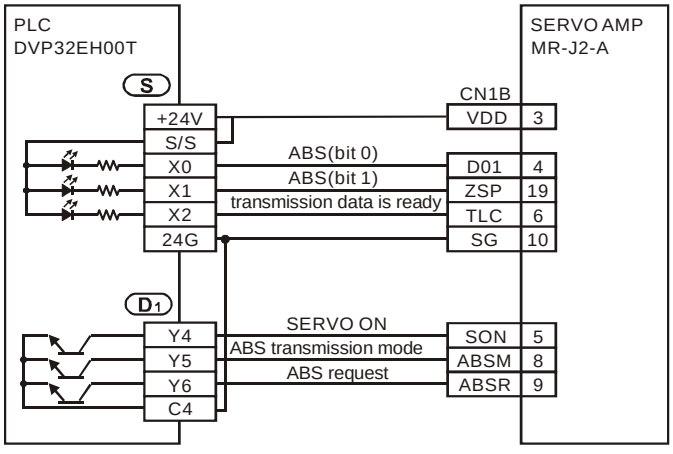
PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Input signal from Servo (occupies 3 consecutive devices) **D_1**: Control signal for controlling Servo (occupies 3 consecutive devices) **D_2**: Absolute position data (32-bit) read from Servo

Explanations:

1. Operand S and D_1 of SA series MPU do not support E, F index register modification.
2. See the specifications of each model for their range of use.
3. This instruction can only be used once in the program.
4. Flag: see remarks for more details.
5. This instruction reads the absolute position (ABS) of MITSUBISHI MR-J2 servo drive (with absolute position check function).
6. **S** will occupy 3 consecutive devices, **S**, **S** +1, and **S** +2. **S** and **S** +1 are connected to the absolute position (bit 0, bit 1) on the servo for data transmitting. **S** +2 is connected to Servo for transmitting data ready flag. See the wiring example below for more details.
7. **D_1** will occupy 3 consecutive devices, **D_1**, **D_1** + 1, **D_1** + 2. **D_1** is connected to SERVO On (SON) of Servo. **D_1**+1 is connected to ABS transmisstion mode of Servo and **D_1**+2 is connected to ABS request signal. See the wiring example below for more details.



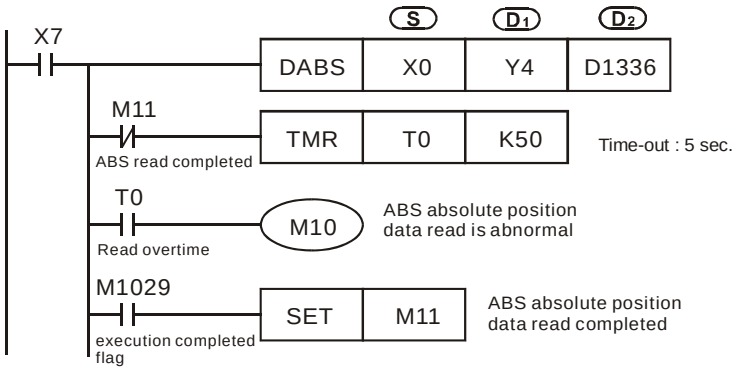
8. **D_2** will occupy 2 consecutive devices **D_2** and **D_2** + 1. **D_2** is the lower 16 bits and **D_2** + 1 is the higher 16 bits. The absolute position data should be written into the present value registers (D1337, D1336) of CH0 pulse (Y0, Y1) or the present value registers (D1339, D1338) of CH1 pulse (Y2, Y3) in EH series MPU; therefore, we suggest you designate the two corresponding registers. If you designate other devices as the registers, you still have to

transmit the data to D1337 and D1336 of CH0 or D1339 and D1338 of CH1. In addition, the absolute position data should be written into the present value registers (D1348, D1349) of CH0 pulse (Y10) or the present value registers (D1350, D1351) of CH1 pulse (Y11) in SC series MPU; therefore, we suggest you designate the two corresponding registers. If you designate other devices as the registers, you still have to transmit the data to D1348 and D1349 of CH0 or D1350 and D1351 of CH1.

9. When DABSR instruction starts to read, after finishing reading the absolute position of SERVO, flag M1029 will be On. The user has to reset the flag.
10. When driving the DABSR command, please specify normally open contact. If the drive contact of DABSR command turns Off when DABSR command read starts, the execution of absolute current value read will be interrupted and result in incorrect data. Please be careful and notice that.

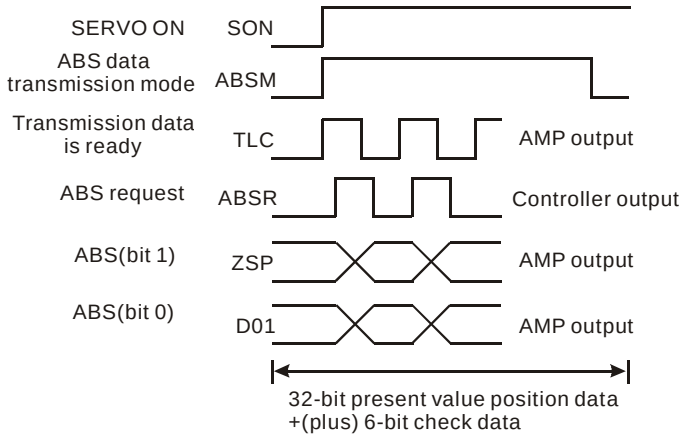
Program Example:

1. When X7 = On, the 32-bit absolute position data read from Servo will be stored in the present value registers (D1337, D1336) of CH0 pulse in EH MPU. At the same time, the timer T10 is enabled and starts to count for 5 seconds. If the reading of the absolute position is not completed after 5 seconds, M10 will be On, indicating that the reading of absolute position encounters abnormality.
2. When enabling the connection to the system, please synchronize the power input of DVP-PLC EH/EH2/SV and SERVO AMP or activate the power of SERVO AMP earlier than DVP-PLC.



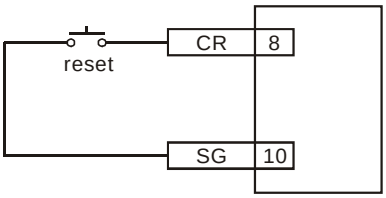
Remarks:

1. If the instruction is interrupted when PLC is still reading the absolute position of SERVO, an ALARM message (ALE5) will occur in SERVO.
2. Timing chart of DABSR instruction reading absolute position:
   a) When DABSR instruction starts to execute, it will drive SERBVO On (SON) and ABS transmittion mode for output.
   b) By “transmission is ready” and “ABS request” signals, you can confirm the transmission and reciept of both sides as well as processing the transmission of the 32-bit present position data plus the 6-bit check data.
   c) The data are transmitted by ABS (bit0, bit1).



3. This instruction is applicable to the Servo motor equipped with absolute positioning function, e.g. Mitsubishi MR-J2-A Servo drive.
4. Select one of the following methods for the initial reading of present absolute position.
   a) Complete zero point return by using reset signal function to execute API 156 ZRN instruction.
   b) After using JOG or manual operation to adjust the zero point position, input a reset signal in SERVO AMP. See the figure of external switch below for whether to use DVP-PLC for output. For the wiring of DVP-PLC and Mitsubishi MR-H2-□A, see remarks of API 159 DRVA instruction.

Ex: Mitsubishi MR-J2-□A



5. Flags explanation:

M1010： (For EH/EH2/SV series MPU) When M1010 is On, CH0 (Y0, Y1) and CH1 (Y2, Y3) will output pulses while END instruction is being executed. When the output starts, M1010 will automatically turn Off.

M1029： (For EH/EH2/SV series MPU) When the first group CH0 (Y0, Y1) pulse output or the execution of other relevant instructions are completed, M1029 will turn On.

M1030： (For EH/EH2/SV series MPU) When the second group CH1 (Y2, Y3) pulse output is completed, M1030 will turn on.

M1102： (For SC series MPU) When the first group CH0 (Y10) pulse output is completed, M1102 will turn On.

M1103： (For SC series MPU) When the second group CH1 (Y11) pulse output is completed, M1103 will turn On.

M1258： (For EH/EH2/SV series MPU) When M1258 is On, CH0 (Y0, Y1) will output reverse pulses.

M1259： (For EH/EH2/SV series MPU) When M1259 is On, CH1 (Y2, Y3) will output reverse pulses.

M1305： (For EH/EH2/SV series MPU) PLSV, DPLSV, DRVI, DDRVI, DRVA, DDRVA instructions for CH0 (Y1, Y2) reverse running.

M1306： (For EH/EH2/SV series MPU) PLSV, DPLSV, DRVI, DDRVI, DRVA, DDRVA instructions for CH1 (Y2, Y3) reverse running.

M1334： (For EH series MPU) When M1334 = On, CH0 (Y0, Y1) pulse output will pause.
(For EH2/SV series MPU) When M1334 = On, CH0 (Y0, Y1) pulse output will stop.
(For SC series MPU) When M1334 = On, the DDRVI and DDRVA execution criteria will stop and CH0 (Y10) pulse output will stop immediately without deceleration.

M1335： (For EH series MPU) When M1335 = On, CH1 (Y2, Y3) pulse output will pause.
(For EH2/SV series MPU) When M1335 = On, CH1 (Y2, Y3) pulse output will stop.
(For SC series MPU) When M1335 = On, DDRVI and DDRVA execution criteria will stop and CH1 (Y11) pulse output will stop immediately without deceleration.

M1520： (For EH2/SV series MPU) When M1520 = On, CH2 (Y4, Y5) pulse output will stop.

M1521： (For EH2/SV series MPU) When M1521 = On, CH3 (Y6, Y7) pulse output will stop.

M1336： (For EH/EH2/SV series MPU) CH0 (Y0, Y1) pulse output indication flag

M1337： (For EH/EH2/SV series MPU) CH1 (Y2, Y3) pulse output indication flag

M1346： (For EH/EH2/SV series MPU) ZRN instruction for “enabling CLEAR output signal” flag

6. Special registers:

D1337, D1336：
1. (For EH/EH2/SV series MPU) Registers for the first group (Y0, Y1) output pulse present value of position control instructions (API 156 ZRN, API 157 PLSV, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1337 is for high word; D1336 is for low word.
2. (For EH/EH2/SV series MPU) Registers for storing the current number of output pulses of the first group (Y0, Y1) output of pulse output instructions (API 57 PLSY, API 59 PLSR). D1337 is for high word; D1336 is for low word.

D1338, D1339：
1. (For EH/EH2/SV series MPU) Registers for the second group (Y2, Y3) output pulse present value of position control instructions (API 156 ZRN, API 157 PLSV, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1339 is for high word; D1338 is for low word.
2. (For EH/EH2/SV series MPU) Registers for storing the current number of output pulses of the second group (Y2, Y3) output of pulse output instructions (API 57 PLSY, API 59 PLSR). D1339 is for high word; D1338 is for low word.

D1340 (D1352)： For setting up the frequencies of the first acceleration segment and the last deceleration segment when the position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA) are executing CH0 (CH1) outputs.

Range of setting:

For EH/EH2/SV series MPU, the speed has to be higher than 10Hz. Frequency lower than 10Hz or higher than maximum output frequency will be output by 10Hz. The default setting in EH/EH2/SV series MPU is 200Hz. For SC series MPU, the speed has to be 100 ~ 100KHz. Frequency lower than 100Hz will be output by 100Hz and frequency higher than

100KHz will be output by 100KHz. The default setting in SC series MPU is 100Hz.

Note: During the control of the stepping motor, please consider the resonance and the limitation on the start frequency when you set up the speed.

D1341, D1342： (For EH/EH2/SV series MPU) For setting up the maximum speed when the position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA) are being executed. D1342 is for high word; D1341 is for low word.

Range of setting: 200KHz fixed.

D1343 (D1353)： For setting up the time of the first acceleration segment and the last deceleration segment when the position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA) are executing CH0 (CH1) outputs.

Range of setting:

For EH/EH2/SV series MPU, the acceleration/deceleration time has to be longer than 10ms. The time shorter than 10ms or longer than 10,000ms will be output by 10ms. The default setting in EH/EH2/SV series MPU is 100ms. For SC series MPU, the time has to be 50 ~ 20,000ms. The time shorter than 50ms will be regarded as 50ms.

Note: During the control of the stepping motor, please consider the resonance and the limitation on the start frequency when you set up the speed.

D1348, D1349： (For SC series MPU) Registers for the first group (Y0, Y1) output pulse present value of position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1349 is for high word; D1348 is for low word.

D1350, D1351： (For SC series MPU) Registers for the second group (Y11) output pulse present value of position control instructions (API 156 ZRN, API 158 DRVI, API 159 DRVA). The present value increases or decreases according to the corresponding rotation direction. D1351 is for high word; D1350 is for low word.

API	Mnemonic	Operands	Function	Controllers
156	D ZRN	S_1 S_2 S_3 D	Zero Return	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	ZRN: 9 steps DZRN: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
S_3	*	*	*	*												
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Zero return speed **S_2**: Creep speed **S_3**: Near p oint signal (DOG) **D**: Pulse output device (please use transistor output module)

Explanations:

1. When **S_1** and **S_2** are used in device F, only 16-bit instruction is applicable.
2. **S_1** and **S_2** of SC series MPU only support device K, H and D. **S_3** of SC series MPU only supports device X10 and X11.
3. Flag: see remarks of API 155 ABSR and API 158 DDRVI for more details.
4. **S_1** is the starting speed of zero return operation. For EH/EH2/SV series MPU, the 16-bit instruction can designate the range of the speed, which is 10 ~ 32,767Hz and the range designated by the 32-bit instruction is 10 ~ 200,000Hz. If the designated speed is slower than 10Hz, the zero return will operate at 10Hz and when the designated speed is faster than 200KHz, the zero return will operate at 200KHz. For SC series MPU, the 32-bit instruction can designate the range of speed , which is 100 ~ 100,000Hz. If the designated speed is slower than 100Hz, the zero return will operate at 100Hz, and when the designated speed is faster than 100KHz, the zero return will operate at 100KHz.
5. **S_2** is the designated low speed after the near point signal (DOG) is On. EH/EH2/SV series MPU can designate the range of **S_2**, which is 10 ~ 32,767Hz and SC series MPU can designate the range 100 ~ 100,000Hz.
6. **S_3** is the designated near point signal (DOG) input (input from A contact). In EH/EH2/SV series MPU, if devices other than the external output device (X10 ~ X17), e.g. X, Y, M, S are designated, they will be affected by the scan period, resulting in dispersion of the zero point. In addition, please note that the MPU cannot designate the same input points X10 ~ X17 as those designated by DCNT and PWD instructions. SC series MPU can only designate X10 and X11 and cannot designate the same input points as those designated by DCNT instruction.
7. EH series MPU has two groups of A/B phase pulse output, CH0 (Y0, Y1) and CH1 (Y2, Y3); EH2/SV series MPU has four groups of A/B phase pulse output, CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks for the setup methods.
8. Zero return output device in different models

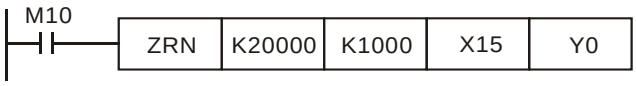
Model	SC MPU	EH MPU	EH2/SV MPU
Zero return output	Y10, Y11	Y0, Y2	Y0, Y2, Y4, Y6

9. When executing API 158 DRVI (releative positioning) or API 159 DRVA (absolute positioning), PLC will automatically store the increasing or decreasing forward/reverse pulses in the present value registers. For

EH/EH2/SV series MPU, Y0: D1337, D1336; Y2: D1339, D1338, Y4: D1376, D1375; Y6: D1378, D1377. For SC series MPU, Y10: D1348, D1349; Y11: D1350, D1351. In this way, you can keep track of the position of the machine at any time. However, due to that the data will be lost when the power of the PLC is switched off, you have to enter the zero point position of the machine when executing zero return for the first time.

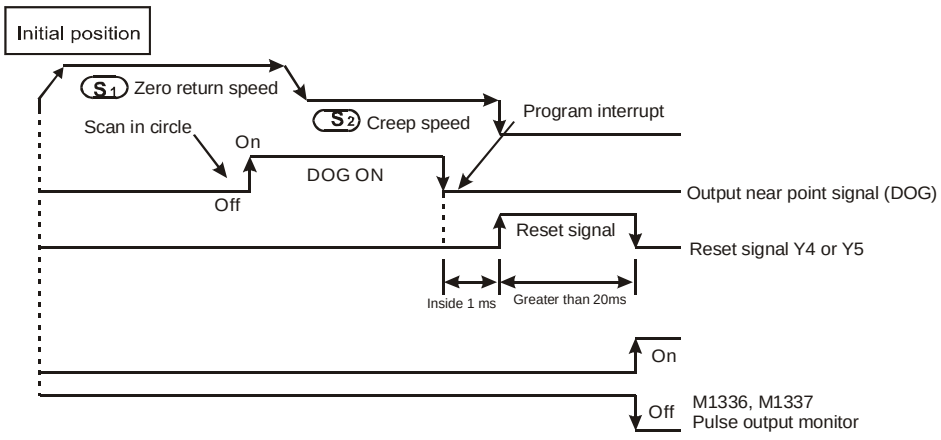
Program Example:

When M10= On, Y0 output pulses start to operate zero return at the frequency of 20KHz. When the zero return meets DOG X15 = On, Y0 output pulses will start to operate by creep speed 1KHz until X15 is Off.



Remarks:

1. Timing chart of the reset signal output for EH/EH2/SV series MPU. (SC series MPU does not support this function.)
   a) When the reset signal flag M1346 = On, after zero return is completed, the PLC can send the reset signal to the servo drive and the signal will last for approximately 20ms. After 20ms, the reset signal will return to Off again.
   b) Output devices for reset signals of EH series MPU:
      CH0 (Y0, Y1) reset output device (Y4)
      CH1 (Y2, Y3) reset output device (Y5)
   c) Output devices for reset signals of EH2/SV series MPU:
      CH0 (Y0, Y1) reset output device (Y10)
      CH1 (Y2, Y3) reset output device (Y11)
      CH2 (Y4, Y5) reset output device (Y12)
      CH3 (Y6, Y7) reset output device (Y13)



Note: The designated devices, X, Y, M, and S, other than the external input devices X10 ~ X17 will be affected by the scan period, 2 times of the scan period at worst.

2. The zero return operation:
   a) When ZRN instruction is executed, the frequency of the first acceleration segment of CH0 (CH1) in EH/EH2/SV series MPU is set by D1340 (D1352). In SC series MPU, CH0 (CH1) will set creep speed as the

start frequency. The acceleration time of EH/EH2/SV/SC series MPU is set by D1343 (D1353). S1 will start to move when the acceleration reaches the zero return speed.

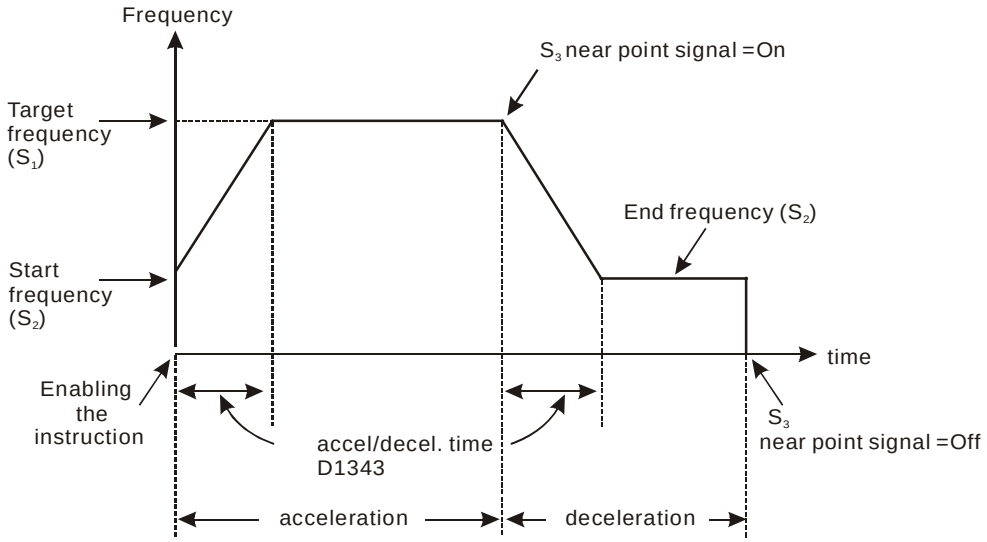
b) When the DOG signal goes from Off to On, the zero return speed will decelerate to **S_2** by the time set in D1343 (D1353).

c) When the DOG signal goes from On to Off and the pulse output stops, 0 will be written in the present value registers (D1337, D1336) of CH0 pulses, D1339 and D1338 of CH1 pulses, D1375 and D1376 of CH2 pulses, and D1377 and D1378 of CH3 pulses in EH/EH2/SV series MPU; 0 will also be written in D1348 and D1349 of Y10 (CH0) pulses or D1350 and D1351 of Y11 (CH1) pulses in SC series MPU.

d) When the DOG signal goes from On to Off and the reset signal flag M1346 = On, Y4 (CH0) or Y5 (CH1) in EH series MPU will output a reset signal; Y10 (CH0), Y11 (CH1) , Y12 (CH2) and Y13 (CH3) in EH2/SV series MPU will output a reset signal.

e) For EH/EH2/SV series MPU, when the pulse output is completed and M1029, M1030, M1036 and M1037 are enabled, indication flag M1336 sent by CH0 pulses, M1337 by CH1, M1522 by CH2 and M1523 by CH3 will be Off. For SC series MPU, when the pulse output is completed, M1102 and M1103 will be enabled.

f) Due to that ZRN (DZRN) instruction cannot locate the position of DOG, the zero return can only be done towards a single direction. In the zero return operation of EH/EH2/SV series MPU, D1337 and D1336 (present value registers) of CH0 pulses or D1339 and D1338 of CH1 pulses are decreasing. In the zero return operation of SC series MPU, D1348 and D1349 of CH0 pulses or D1350 and D1351 of CH1 pulses are also decreasing.



g) ZRN (DZRN) instruction is applicable to servo motor with absolute positioning function, e.g. Mitsubishi MR-J2-A servo drive. Even when the power is switched off, the current position can still be recorded. In addition, the current position of servo drive can be read by API 155 DABSR of EH/EH2/SV/SV series MPU; therefore only one zero return operation is required and no zero return has to be done after the power is switched off.

h) When the drive contact of ZRN instruction is On, CH0 (CH1) will read the acceleration/deceleration time set in D1343 (D1353) and accelerate to the zero return speed, waiting for the DOG and decelerate to creep speed. When the DOG is Off, the pulse output will stop immediately.

i) For SC series MPU, many ZRN instructions can be compiled in the program but only one instruction can be executed when the PLC program is being executed. For example, provided there is already an instruction enabling Y10 output, other instructions enabling also Y10 output will not be executed. The principle of the instruction execution is “first come, first executed”.

j) For SC series MPU, when you designate Y10 as the output device, you can choose either X10 or X11 for DOG input in the “acceleration to deceleration” segment. In other words, when designating Y11 as the output device, you can also choose either X10 or X11 for DOG input.

k) For SC series MPU, due to that this instruction does not compare between the number of output pulses, the DOG input (from Off to On) will therefore become the trigger of acceleration converting to deceleration. The “On” time of DOG has to be longer than 10us; otherwise the signal may be regarded as useless interference.

l) For SC series MPU, when the execution of the instrucion enters the deceleration segment and the output frequency reaches creep speed (end frequency), the output will stop when DOG goes from On to Off.

m) For SC series MPU, the current accumulated number of pulses of Y10 is stored in D1348 and D1349 and that of Y11 is stored in D1350 and D1351. Then the program operates from STOP to RUN or from RUN to STOP, the contents will not be cleared to 0.

n) For SC series MPU, M1102 = On indicates the end of Y10 pulse output; M1103 = On indicates the end of Y11 pulse output.

o) For SC series MPU, after the instruction is executed, all parameters cannot be modified unless the execution of the instruction stops.

p) For SC series MPU, when the execution of the stops, all outputs will stop immediately no matter what type of the output it is.

API	Mnemonic		Operands	Function	Controllers
157	D	PLSV	(S) (D1) (D2)	Adjustable Speed Pulse Output	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	PLSV: 7 steps DPLSV: 13 steps
D_1		*														
D_2		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

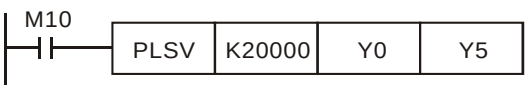
S: Pulse output frequency **D_1**: Pulse output device (please use transistor output module) **D_2**: Output device for the signal of rotation direction

Explanations:

1. See remarks for the setting range of **S**, **D_1** and **D_2**.
2. Flag: see remarks of API 155 ABSR and API 158 DDRVI for more details.
3. **S** is the designated pulse output frequency. The 16-bit instruction can designate its range 0 ~ +32,767Hz, 0 ~ -32,768Hz. The ranges designated by 32-bit instruction are 0 ~ +200,000Hz and 0 ~ -200,000Hz. “+/-” signs indicate forward/backward directions. During the pulse output, the frequency can be changed, but not the frequencies of different directions.
4. **D_1** is the pulse output device. EH series MPU can designate Y0 and Y2 and EH2/SV series MPU can designate Y0, Y2, Y4 and Y6.
5. The operation of **D_2** corresponds to the “+” or “-“ of **S**. When **S** is “+”, **D_2** will be On; when **S** is “-“, **D_2** will be Off.
6. PLSV instruction does not have settings for acceleration and deceleration. Please use API 67 RAMP for the acceleration and deceleration of pulse output frequency.
7. During the pulse output executed by PLSV instruction, the drive contact turning Off will result in the immediate stop of the output without going through a deceleration.
8. When the absolute value of the input frequency during the execution of DPLSV is bigger than 200KHz, the output will operate at 200KHz.
9. For EH/EH2/SV series MPU, D1222, D1223, D1383 and D1384 are the time differences sent between the direction setup signal and pulse output points of CH0, CH1, CH2 and CH3.
10. For EH/EH2/SV series MPU, M1305, M1306, M1532 and M1533 are the flags of the direction signals of CH0, CH1, CH2 and CH3. When S is “+”, the output will operate towards a forward direction and the flag will go Off. When S is “-“, the output will operate towards a backward direction and the flag will go On.

Program Example:

When M10 = On, Y0 will output pulses at 20KHz. Y5 = On indicates forward pulses.



API	Mnemonic		Operands	Function	Controllers
158	D	DRVI	(S1) (S2) (D1) (D2)	Drive to Increment	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	DRVI: 9 steps DDRVI: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D_1		*														
D_2		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Number of output pulses (relative designation) **S_2**: Pulse output frequency **D_1**: Pulse output device (please use transistor output module) **D_2**: Output device for the signal of rotation direction

Explanations:

1. See remarks for the setting range of **S_1**, **S_2**, **D_1** and **D_2**.
2. **S_1** and **S_2** of SC series MPU only support device K, H and D.
3. Flag: see remarks for more details.
4. **S_1** is the number of output pulses (relative designation). For EH/EH2/SV series MPU, the 16-bit instruction can designate the range -32,768 ~ +32,767. The range designated by 32-bit instruction is -2,147,483,648 ~ +2,147,483,647. For SC series MPU, the 32-bit instruction can designate the range -2,147,483,648 ~ +2,147,483,647. “+/-” signs indicate forward/backward directions.
5. **S_2** is the designated pulse output frequency. For EH/EH2/SV series MPU, the 16-bit instruction can designate its range 10 ~ 32,767Hz. The range designated by 32-bit instruction is 10 ~ 200,000Hz. For SC series MPU, the 32-bit instruction can designate the range 100 ~ 100,000Hz.
6. EH series MPU has two groups of A/B phase pulse output, CH0 (Y0, Y1) and CH1 (Y2, Y3). EH2/SV series MPU has four groups of A/B phase pulse output, CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks for the setup methods.
7. Pulse output device **D_1** in different models

Model	SC MPU	EH MPU	EH2/SV MPU
Pulse output end	Y10, Y11	Y0, Y2	Y0, Y2, Y4, Y6

8. The operation of **D_2** corresponds to the “+” or “-“ of **S_1**. When **S_1** is “+”, **D_2** will be On; when **S_1** is “-“, **D_2** will be Off. **D_2** will not be Off immediately after the pulse output is over; it will be Off only when the drive contact of the instruction turns Off.
9. For EH/EH2/SV series MPU, **S_1** is
   - The 32-bit data stored in the present value registers D1337 (high word) and D1336 (low word) of CH0 (Y0, Y1).
   - The 32-bit data stored in the present value registers D1339 (high word) and D1338 (low word) of CH1 (Y2, Y3).
   - The 32-bit data stored in the present value registers D1376 (high word) and D1375 (low word) of CH2 (Y4, Y5).
   - The 32-bit data stored in the present value registers D1378 (high word) and D1377 (low word) of CH3 (Y5,

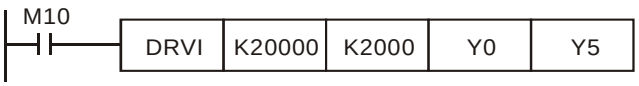
Y6).

- When in backward direction, the content in the present value register will decrease.

10. For SC series MPU, $\mathbf{S_1}$ is the 32-bit data stored in the present value registers D1348 (low word) and D1349 (high word) of CH0 (Y10) or the 32-bit data stored in the present value registers D1350 (low word) and D1351 (high word) of CH1 (Y11). When in backward direction, the content in the present value register will decrease. When the program goes from STOP to RUN or from RUN to STOP, the content in the present value register will remain unchanged.
11. When DRVI instruction is executing pulse output, you cannot change the content of all operands. The changes will be valid next time when DRVI instruction is enabled.
12. For EH/EH2/SV series MPU, when the drive contact of DRVI instruction is Off, even the indication flag M1336 sent by CH0 pulses, M1337 sent by CH1 pulses, M1522 sent by CH2 pulses and M1523 sent by CH3 pulses are “On”, DRVI instruction will not be driven again.
13. When the absolute value of the input frequency of DDRVI insturction in EH/EH2/SV series MPU is larger than 200KHz, the output will be operated at 200KHz. When the absolute value of the input frequency is smaller than 10Hz, the output will be operated at 10Hz.
14. D1343 (D1353) is for setting up the time of the first acceleartion segment and last deceleration segment of CH0 (CH1). The acceleration and deceleration time of EH/EH2/SV series MPU shall not be shorter than 10ms. The output will be operated for 10ms if the time is shorter than 10ms or longer than 10,000ms (default setting = 100ms). The time range for SC series MPU is 50 ~ 20,000ms. The output will be operated for 20,000ms or 50ms if the time set is longer than 20,000ms or shorter than 50ms.
15. D1340 (D1352) is for setting up the start/end frequency of Y10 (Y11). If $\mathbf{S_2}$ is less than or equals start/end frequency, the pulse output frequency will be executed by the start/end frequency.
16. For EH/EH2/SV series MPU, M1305 (M1306) is the direction signal of CH0 (CH1). When $\mathbf{S_1}$ is a positive number, the output will be operated in a forward direction and M1305 (M1306) will be Off. When $\mathbf{S_1}$ is a negative number, the output will be operated in a backward direction and M1305 (M1306) will be On.

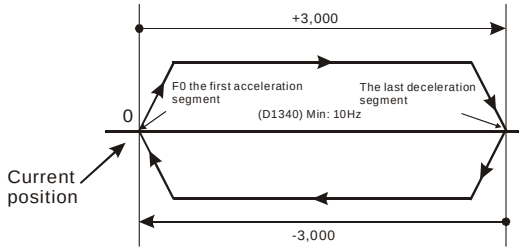
Program Example:

When M10= On, Y0 will output 20,000 pulses (relative designation) at 2KHz. Y5 = On indicates the pulses are executed in forward direction.

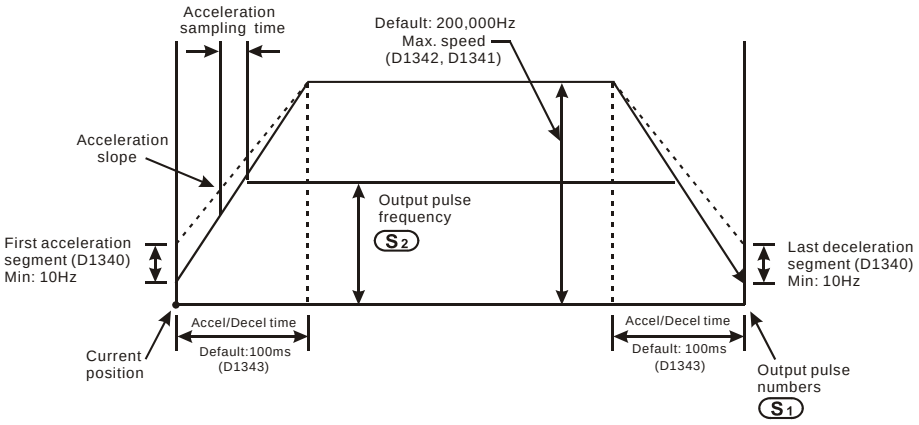


Remarks:

1. Explanations on EH/EH2/SV series MPU:
   a) Relative position control: Designating the traveling distance starting from the current position by “+/ -“ signs; also known as a relative driving method.

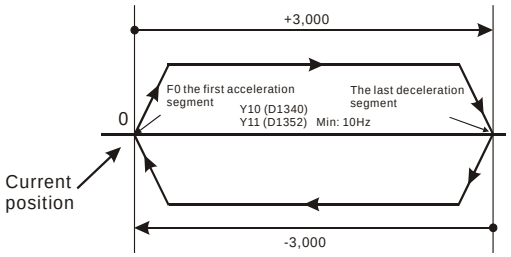


b) Settings of relative positioning and the acceleration/deceleration speed:

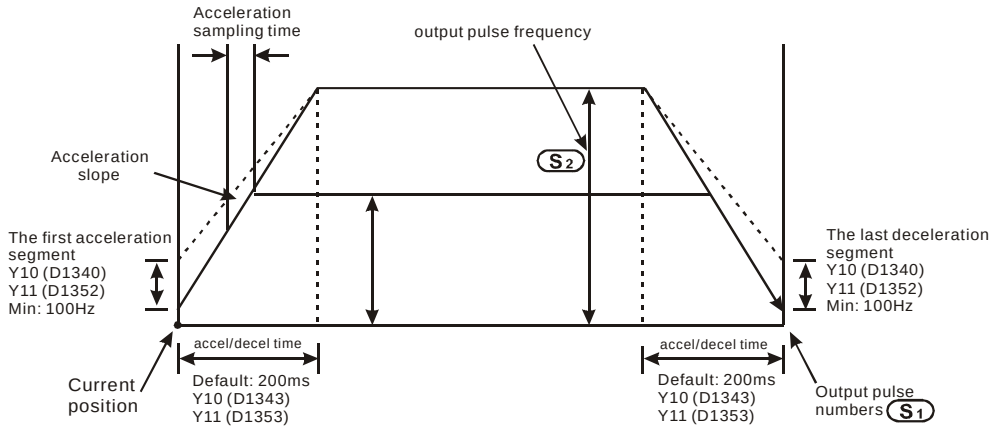


2. Explanations on SC series MPU:

a) Relative position control: Designating the traveling distance starting from the current position by "+/ -" signs; also known as a relative driving method.



b) Settings of relative positioning and the acceleration/deceleration speed: D1343 (D1353) is for settings of the time of the first acceleration segment and last deceleration segment of Y10 (Y11). D1340 (D1352) is for settings of start/end frequency of Y10 (Y11).



c) Many DRVI instructions can be compiled synchronously in the program, but only one instruction can be activated whenever the PLC executes the program. For example, if Y10 output has already been activated by an instruction, other instructions that are also used to activate Y10 output will not be excecuted. Therefore, the principle of the instruction activation sequence is “first activated, first executed”.

d) When Y10 is activated by DDRVI instruction, the output function of Y10 will be disabled until DDRVI is OFF. The same rule applies to Y11.

e) Once the instruction is activated, all other parameters cannot be modified until the instruction is disabled.

f) When the instruction is disabled but the output has not yet completed:

M1334 = On indicates that Y10 will stop output immediately.

M1334 = Off indicates that Y10 will decelerate according to the deceleration time till it reaches end frequency and stop the pulse output.

M1335 corresponds to Y11 output and applies the same rule.

3. Flags for SC series MPU:

M1102： M1102 = On after Y10 pulse output is completed.

M1103： M1103 = On after Y11 pulse outout is completed.

M1334： Y10 pulse output stops immediately without deceleration when the pulse output instruction is disabled.

M1335： Y11 pulse output stops immediately without deceleration when the pulse output instruction is disabled.

4. Special registers for SC series MPU:

D1348： Low word of the current number of Y10 output pulses.

D1349： High word of the current number of Y10 output pulses.

D1350： Low word of the current number of Y11 output pulses.

D1351： High word of the current number of Y11 output pulses..

D1340： Settings of the first start frequency and the last end frequency of Y10 output pulses.

D1352： Settings of the first start frequency and the last end frequency of Y11 output pulses.

D1343： Settings of the acceleration/deceleration time of Y10 output pulses.

D1353： Settings of the acceleration/deceleration time of Y11 output pulses.

5. Flags for EH/EH2/SV series MPU:

M1010： For EH/EH2/SV, when M1010 = On, CH0, CH1, CH2 and CH3 will output pulses when END instruction is being executed. M1010 will be Off automatically when the output starts.

M1029： For EH/EH2/SV, M1029 = On after CH0 pulse output is completed.

M1030： For EH/EH2/SV, M1030 = On after CH1 pulse output is completed.

M1036： For EH2/SV, M1036 = On after CH2 pulse output is completed.

M1037： For EH2/SV, M1037 = On after CH3 pulse output is completed.

M1305： For EH/EH2/SV, direction signal of CH0.

M1306： For EH/EH2/SV, direction signal of CH1.

M1334： For EH, CH0 pulse output pauses.

For EH2/SV, CH0 pulse output stops.

M1335： For EH, CH1 pulse output pauses.

For EH2/SV, CH1 pulse output stops.

M1336： For EH/EH2/SV, “CH0 sends out pulses” indication.

M1337： For EH/EH2/SV, “CH1 sends out pulses” indication.

M1520： For EH2/SV, CH2 pulse output stops.

M1521： For EH2/SV, CH3 pulse output stops.

M1522： For EH2/SV, “CH2 sends out pulses” indication.

M1523： For EH2/SV, “CH3 sends out pulses” indication.

M1534： For EH2/SV, designated deceleration time of CH0 (should be used with D1348).

M1535： For EH2/SV, designated deceleration time of CH1 (should be used with D1349).

M1536： For EH2/SV, designated deceleration time of CH2 (should be used with D1350).

M1537： For EH2/SV, designated deceleration time of CH3 (should be used with D1351).

M1532： For EH2/SV, direction signal of CH2.

M1533： For EH2/SV, direction signal of CH3.

6. Special registers for EH/EH2/SV series MPU:

D1220： For EH/EH2/SV, phase setting of CH0 (Y0, Y1): D1220 determines the phase by the last two bits; other bits are invalid.

1. K0: Y0 output
2. K1: Y0, Y1 AB-phase output; A ahead of B.
3. K2: Y0, Y1 AB-phase output; B ahead of A.
4. K3: Y1 output

D1221： For EH/EH2/SV, phase setting of CH1 (Y2, Y3): D1221 determines the phase by the last two bits; other bits are invalid.

1. K0: Y2 output
2. K1: Y2, Y3 AB-phase output; A ahead of B.
3. K2: Y2, Y3 AB-phase output; B ahead of A.
4. K3: Y3 output

D1222： For EH/EH2/SV, the time difference between the direction signal and pulse output sent by CH0.

D1223： For EH/EH2/SV, the time difference between the direction signal and pulse output sent by CH1.

D1229： For EH2/SV, phase setting of CH2 (Y4, Y5): D1229 determines the phase by the last two bits; other bits are invalid.

1. K0: Y4 output
2. K1: Y4, Y5 AB-phase output; A ahead of B.
3. K2: Y4, Y5 AB-phase output; B ahead of A.
4. K3: Y5 output

D1230： For EH2/SV, phase setting of CH3 (Y6, Y7): D1230 determines the phase by the last two bits; other bits are invalid.

1. K0: Y6 output
2. K1: Y6, Y7 AB-phase output; A ahead of B.
3. K2: Y6, Y7 AB-phase output; B ahead of A.
4. K3: Y7 output

D1336： For EH/EH2/SV, low word of the current number of output pulses from CH0.

D1337： For EH/EH2/SV, high word of the current number of output pulses from CH0.

D1338： For EH/EH2/SV, low word of the current number of output pulses from CH1.

D1339： For EH/EH2/SV, high word of the current number of output pulses from CH1.

D1340： For EH/EH2/SV, settings of the first start frequency and the last end frequency of CH0.

D1343： For EH/EH2/SV, settings of acceleration/deceleration time for CH0 pulse output.

D1352： For EH/EH2/SV, settings of the first start frequency and the last end frequency of CH1.

D1353： For EH/EH2/SV, settings of acceleration/deceleration time for CH1 pulse output.

D1375： For EH2/SV, low word of the current number of output pulses from CH2.

D1376： For EH2/SV, high word of the current number of output pulses from CH2.

D1377： For EH2/SV, low word of the current number of output pulses from CH3.

D1378： For EH2/SV, high word of the current number of output pulses from CH3.

D1379： For EH2/SV, settings of the first start frequency and the last end frequency of CH2.

D1380： For EH2/SV, settings of the first start frequency and the last end frequency of CH3.

D1348： For EH2/SV, deceleration time for CH0 pulse output when M1534 = On.

D1349： For EH2/SV, deceleration time for CH1 pulse output when M1535 = On.

D1350： For EH2/SV, deceleration time for CH2 pulse output when M1536 = On.

D1351： For EH2/SV, deceleration time for CH3 pulse output when M1537 = On.

D1381： For EH2/SV, settings of acceleration/deceleration time for CH2 pulse output.

D1382： For EH2/SV, settings of acceleration/deceleration time for CH3 pulse output.

D1383： For EH2/SV, the time difference between the direction signal and pulse output sent by CH2.

D1384： For EH2/SV, the time difference between the direction signal and pulse output sent by CH3.

API	Mnemonic		Operands	Function	Controllers
159	D	DRVA	(S1) (S2) (D1) (D2)	Drive to Absolute	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	DRVA: 9 steps DDRVA: 17 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D_1		*														
D_2		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Number of output pulses (absolute designation) **S_2**: Pulse output frequency **D_1**: Pulse output device (please use transistor output module) **D_2**: Output device for the signal of rotation direction

Explanations:

1. See remarks for the setting range of **S_1**, **S_2**, **D_1** and **D_2**.
2. **S_1** and **S_2** of SC series MPU only support device K, H and D.
3. Flag: see remarks of API 158 DRVI for more details.
4. **S_1** is the number of output pulses (absolute designation). For EH/EH2/SV series MPU, the 16-bit instruction can designate the range -32,768 ~ +32,767. The range designated by 32-bit instruction is -2,147,483,648 ~ +2,147,483,647. For SC series MPU, the 32-bit instruction can designate the range -2,147,483,648 ~ +2,147,483,647. “+/-” signs indicate forward/backward directions.
5. **S_2** is the designated pulse output frequency. For EH/EH2/SV series MPU, the 16-bit instruction can designate its range 10 ~ 32,767Hz. The range designated by 32-bit instruction is 10 ~ 200,000Hz. For SC series MPU, the 32-bit instruction can designate the range 100 ~ 100,000Hz.
6. EH series MPU has two groups of A/B phase pulse output, CH0 (Y0, Y1) and CH1 (Y2, Y3). EH2/SV series MPU has four groups of A/B phase pulse output, CH0 (Y0, Y1), CH1 (Y2, Y3), CH2 (Y4, Y5) and CH3 (Y6, Y7). See remarks for the setup methods.
7. Pulse output device **D_1** in different models

Model	SC MPU	EH MPU	EH2/SV MPU
Pulse output end	Y10, Y11	Y0, Y2	Y0, Y2, Y4, Y6

8. When **S_1** is larger than the current relative position, **D_2** will be Off; when **S_1** is smaller than the current relative position, **D_2** will be On. **D_2** will not be Off immediately after the pulse output is over; it will be Off only when the drive contact of the instruction turns Off.
9. For EH/EH2/SV series MPU, **S_1** is
   - The 32-bit data stored in the present value registers D1337 (high word) and D1336 (low word) of CH0 (Y0, Y1).
   - The 32-bit data stored in the present value registers D1339 (high word) and D1338 (low word) of CH1 (Y2, Y3).
   - The 32-bit data stored in the present value registers D1376 (high word) and D1375 (low word) of CH2 (Y4, Y5).
   - The 32-bit data stored in the present value registers D1378 (high word) and D1377 (low word) of CH3 (Y5,

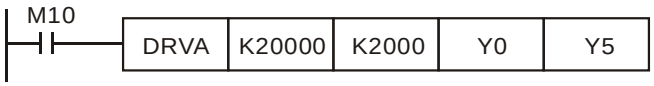
Y6).

When in backward direction, the content in the present value register will decrease.

10. For SC series MPU, S_1 is the 32-bit data stored in the present value registers D1348 (low word) and D1349 (high word) of CH0 (Y10) or the 32-bit data stored in the present value registers D1350 (low word) and D1351 (high word) of CH1 (Y11). When in backward direction, the content in the present value register will decrease. When the program goes from STOP to RUN or from RUN to STOP, the content in the present value register will remain unchanged.
11. For EH/EH2/SV series MPU, when DRVA instruction is executing pulse output, you cannot change the content of all operands. The changes will be valid next time when DRVA instruction is enabled.
12. For EH/EH2/SV series MPU, when the drive contact of DRVA instruction is Off, the pulse output will decelerate to stop and M1029 and M1030 will be enabled. For SC series MPU, the pulse output will decelerate to stop and M1102 and M1103 will be enabled.
13. For EH/EH2/SV series MPU, when the drive contact of DRVA instruction is Off, even the indication flag M1336 sent by CH0 pulses or M1337 sent by CH1 pulses are “On”, DRVA instruction will not be driven again.
14. When the absolute value of the input frequency of DRVA and DDRVA insturctions in EH/EH2/SV series MPU is larger than 200KHz, the output will be operated at 200KHz. When the absolute value of the input frequency is smaller than 10Hz, the output will be operated at 10Hz.
15. D1343 (D1353) is for setting up the time of the first acceleartion segment and last deceleration segment of CH0 (CH1). The acceleration and deceleration time of EH/EH2/SV series MPU shall not be shorter than 10ms. The output will be operated for 10ms if the time is shorter than 10ms or for 100ms (default) if the time is longer than 10,000ms. The time range for SC series MPU is 50 ~ 20,000ms. The output will be operated for 20,000ms or 50ms if the time set is longer than 20,000ms or shorter than 50ms.
16. For EH/EH2/SV series MPU, M1305 (M1306) is the direction signal of CH0 (CH1). When S_1 is a positive number, the output will be operated in a forward direction and M1305 (M1306) will be Off. When S_1 is a negative number, the output will be operated in a backward direction and M1305 (M1306) will be On.
17. D1340 (D1352) is for setting up the start/end frequency of Y10 (Y11). If S_2 is less than or equals start/end frequency, the pulse output frequency will be executed by the start/end frequency.

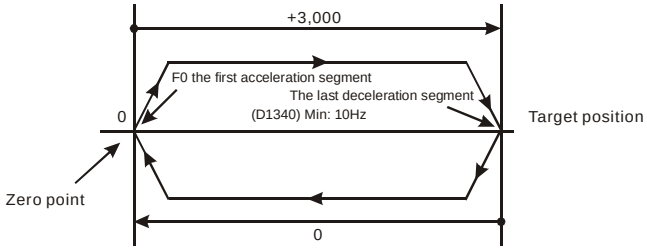
Program Example:

When M10= On, Y0 will output 20,000 pulses (absolute designation) at 2KHz. Y5 = On indicates the pulses are executed in forward direction.

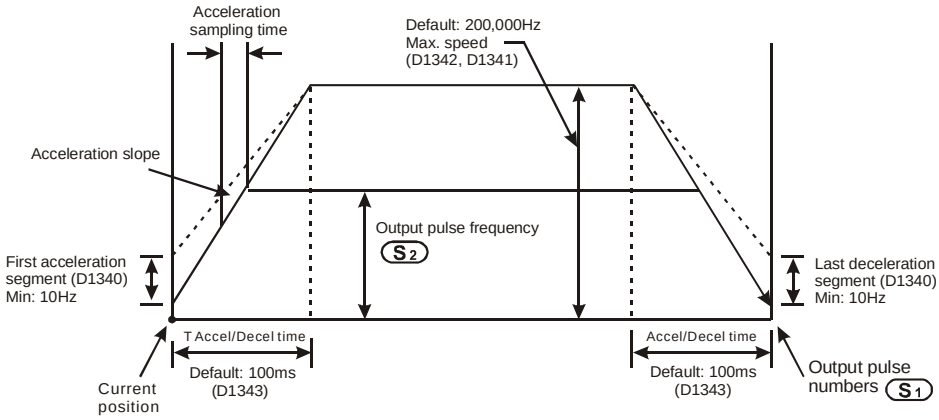


Remarks:

1. Explanations on EH/EH2/SV series MPU:
   a) Absolute position control: Designating the traveling distance starting from the zero point (0); also known as a absolute driving method.

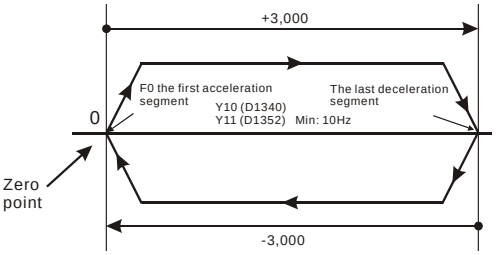


b) Settings of absolute positioning and the acceleration/deceleration speed:

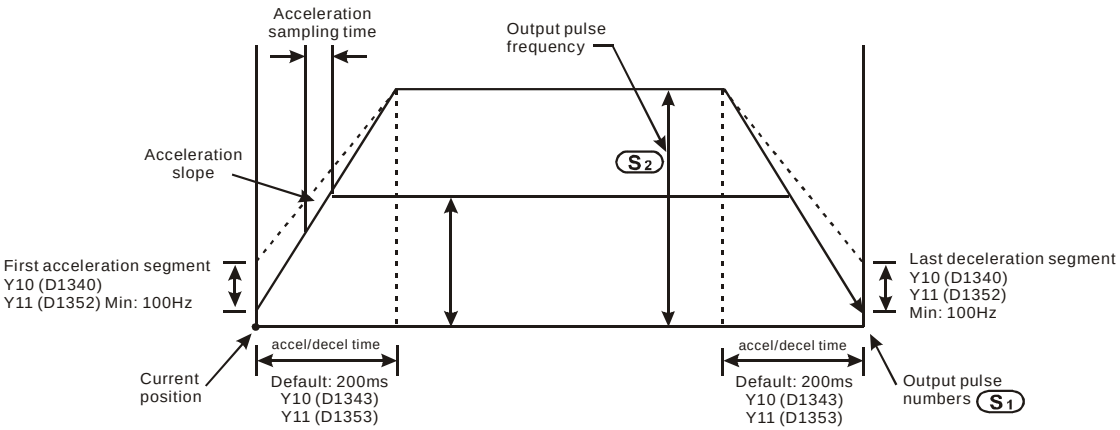


2. Explanations on SC series MPU:

a) Absolute position control: Designating the traveling distance starting from the zero point (0); also known as a absolute driving method.



b) Settings of absolute positioning and the acceleration/deceleration speed: D1343 (D1353) is for settings of the time of the first acceleration segment and last deceleration segment of Y10 (Y11). D1340 (D1352) is for settings of start/end frequency of Y10 (Y11).



c) Many DRVA instructions can be compiled synchronously in the program, but only one instruction can be activated whenever the PLC executes the program. For example, if Y10 output has already been activated by an instruction, other instructions that are also used to activate Y10 output will not be excecuted. Therefore, the principle of the instruction activation sequence is “first activated, first executed”.

d) When Y10 is activated by DDRVA instruction, the output function of Y10 will be disabled until DDRVAis OFF. The same rule applies to Y11.

e) Once the instruction is activated, all other parameters cannot be modified until the instruction is disabled.

f) When the instruction is disabled but the output has not yet completed:

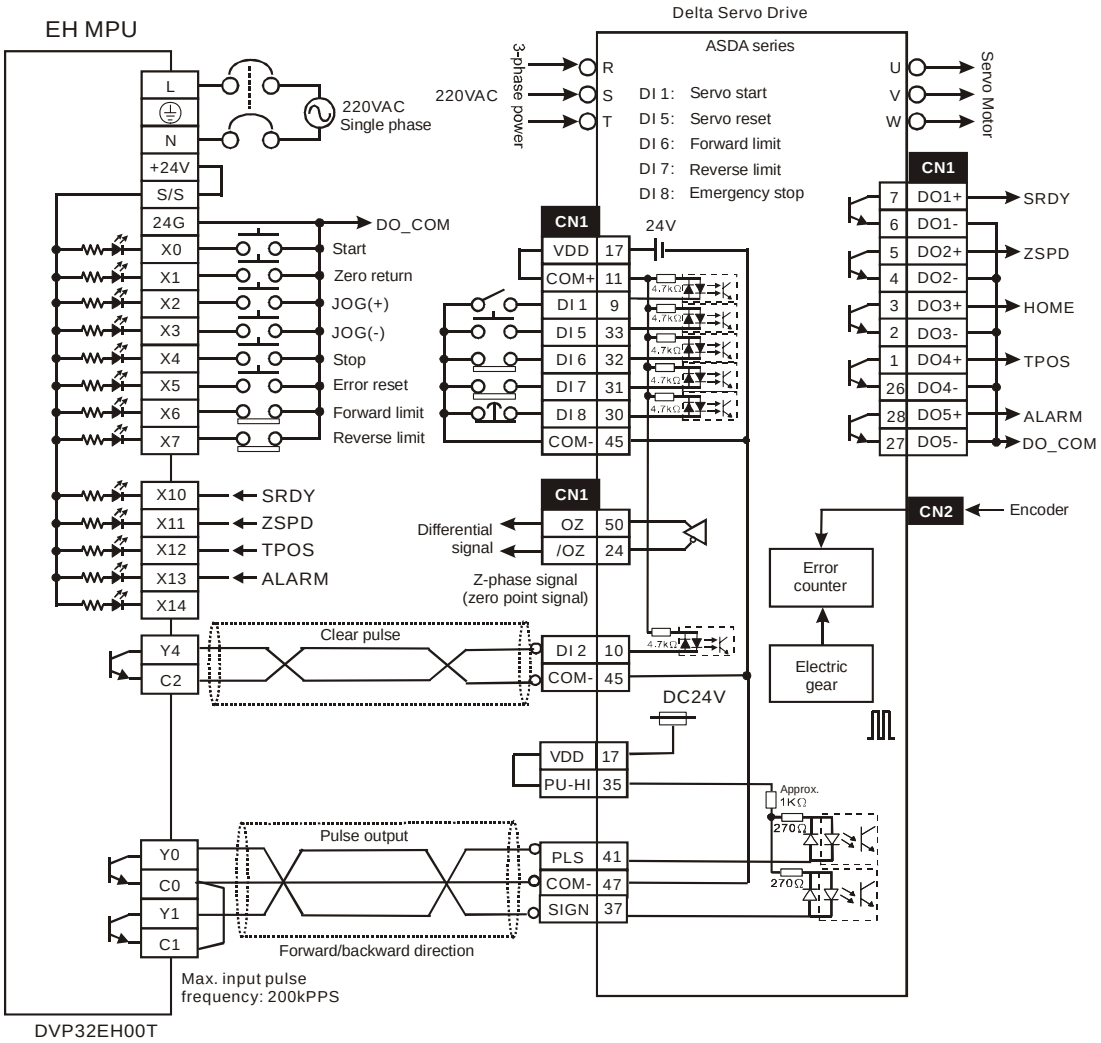
M1334 = On indicates that Y10 will stop output immediately.

M1334 = Off indicates that Y10 will decelerate according to the deceleration time till it reaches end frequency and stop the pulse output.

M1335 corresponds to Y11 output and applies the same rule.

3. See remarks of DDRVI instruction for more details on the flags.

4. Wiring of DVP-EH series and Delta ASDA servo drive:



Note:

(a) The parameter setting of Delta ASDA servo drive:

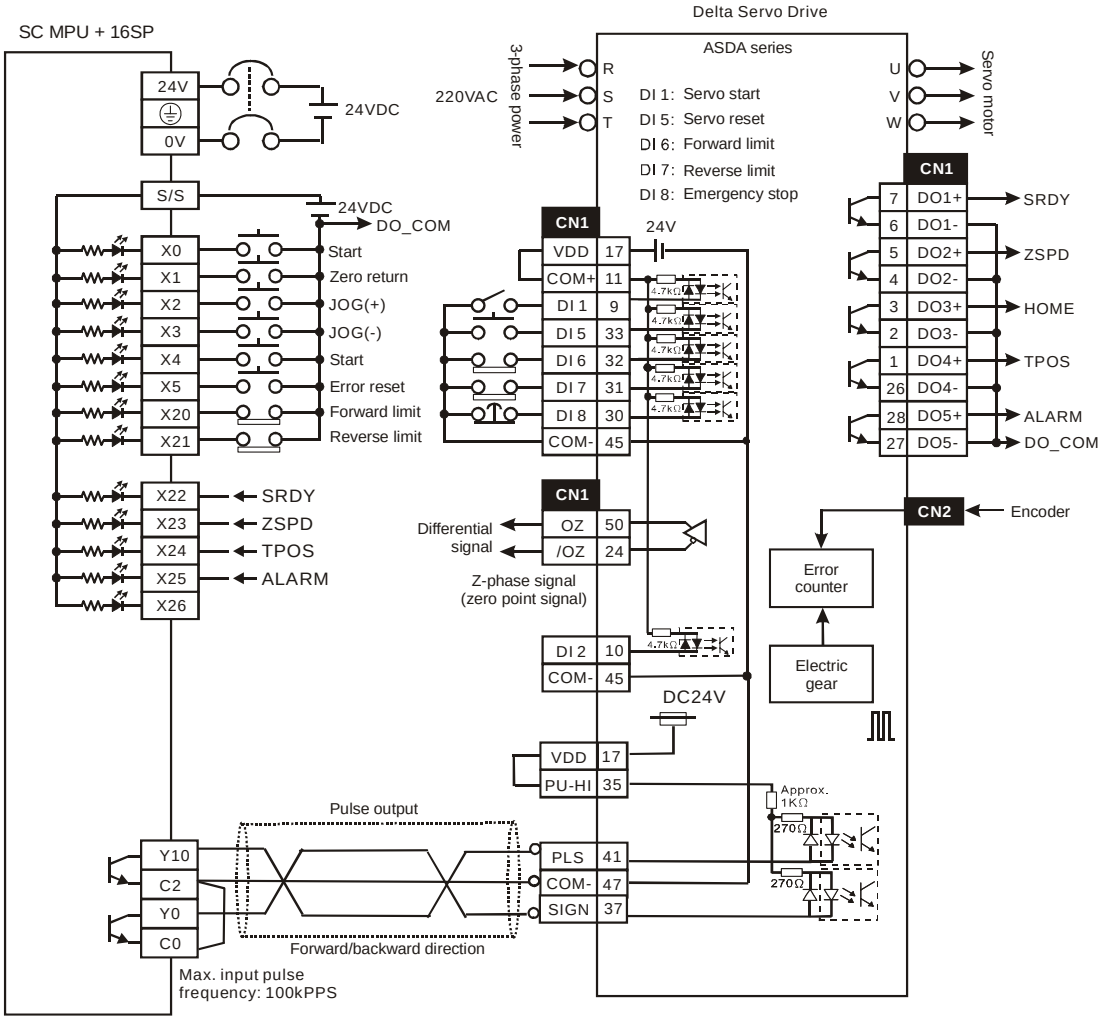
P1-01: position mode

P1-00: pulse input type as Pulse+DIR.

(b) The forward/reverse limit switch should be connected to SERVO AMP.

(c) The “clear pulse” signal will clear the current number of pulses left inside the servo.

5. Wiring of DVP-SC series and Delta ASDA servo drive:



DVP12SC11T+DVP16SP11T

Note:

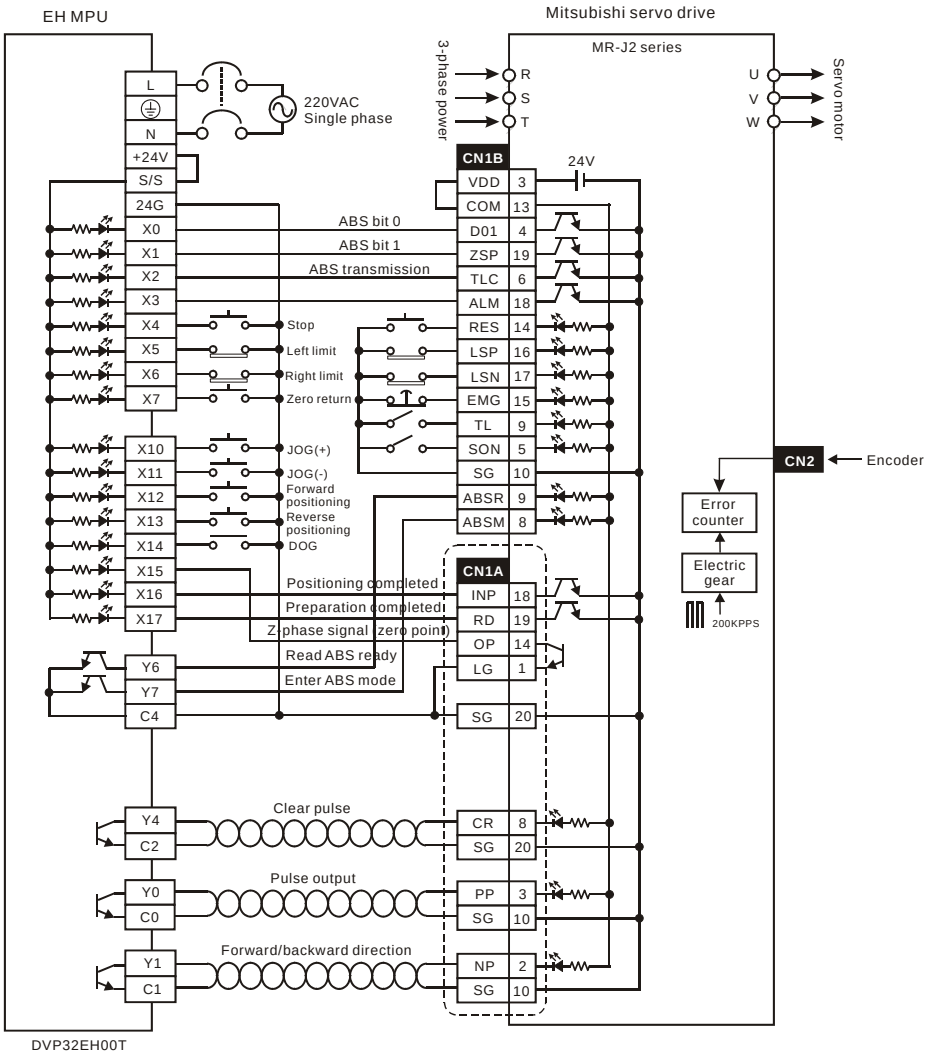
(a) The parameter setting of Delta ASDA servo drive:

P1-01: position mode

P1-00: pulse input type as Pulse+DIR.

(b) The forward/reverse limit switch should be connected to SERVO AMP.

6. Wiring of DVP-EH series PLC and a Mitsubishi MR-J2-□A Servo drive:



Note:

(a) When detecting an absolute position by using DABSR instruction, the parameter setting of a Mitsubishi MR-J2-□A servo drive that connects to Delta EH series PLC:

P0: position mode.

P1: using absolute value.

P21: pulse input type as Pulse+DIR.

(b) The forward/reverse limit switch should be connected to SERVO AMP.

(c) When using OP (Z-phase signal) in servo and given that the Z-phase signal is a high-frequency one when the motor is running at high speed, the valid detection can only be possible when the signal is within the range detectable by PLC. When using OP (Z phase signal) of the servo, if Z phase signal is a high frequency signal during high-speed motor operation, the high frequency signal shall be within the available range that can be detected by PLC.

7. Cautions when designing a position control program:

a) There is no limitation on the times of using the position control instructions, API 156 ZRN, API 157 PLSV, API 158 DRVI, and API 159 DRVA. However, the user still have to note that:

i. Do not drive the position control instructions which use the same output CH0(Y0, Y1) or CH1(Y2, Y3) simultaneously. Otherwise, they will be treated as repeated outputs and cannot function normally.

ii. It is recommended that you use the step ladder instruction (STL) to design the position control program (see the example below).

b) How to use the position control instructions (API 156 ABSR, API 157 PLSV, API 158 DRVI, and API 159 DRVA) and pulse output instructions (API 57 PLSY, API 58 PWM and API 59 PLSR) at the same time.

The position control instruction and pulse output instruction share the 32 bits of the present value register (D1337 high word; D1336 low word) of CH0 (Y0, Y1) or the present value register of CH1 (Y2, Y3), which will make the operation complicated. Therefore, it is recommanded that you replace the pulse output instruction with position control instruction.

c) Explanations on the (Y0, Y1) pulses from CH0 and (Y2, Y3) pulses from CH1.

Voltage range: DC5V ~ DC24V

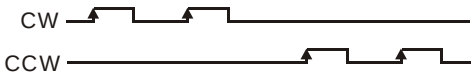
Current range: 10mA ~ 100mA

Output pulse frequency: Y0, Y2 at 200KHz; Y1, Y3 at 10KHz.

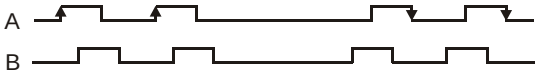
8. Settings of pulse output signals in the operation of position control for EH/EH2/SV series MPU:

a) Pulse + DIR (recommended)

b) CW/CCW (limited frequency at 10KHz)



c) A/B-phase output (limited frequency at 10KHz)



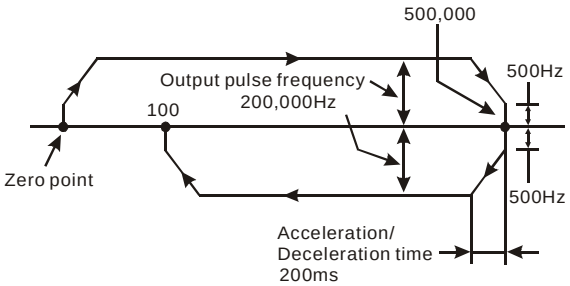
9. Follow the above output settings of PLC for the pulse input parameters of SERVO AMP or stepping motor.

10. For EH/EH2/SV series MPU, when Y0 output adopts many high-speed pulse output instructions (PLSY, PWM, PLSR) and position control instructions (ZRN, PLSV, DRVI, DRVA) in a program and these instructions are executed synchronously in the same scan period, PLC will execute the instruction with the fewest step numbers.

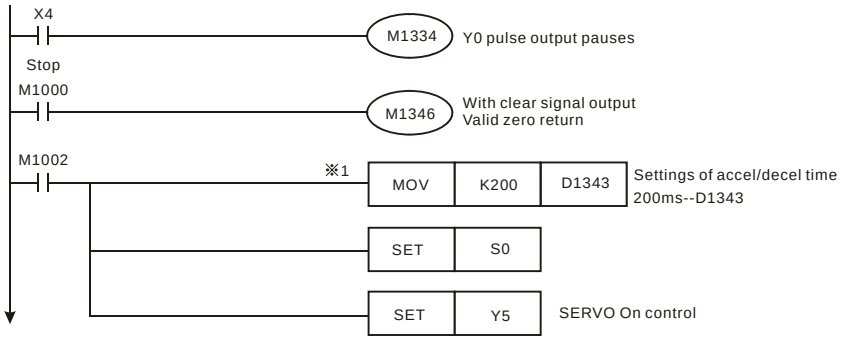
Programming example for forward/reverse operation:

For the wiring, see the wiring drawing of DVP-EH series and Mitsubishi MR-J2-□A servo drive

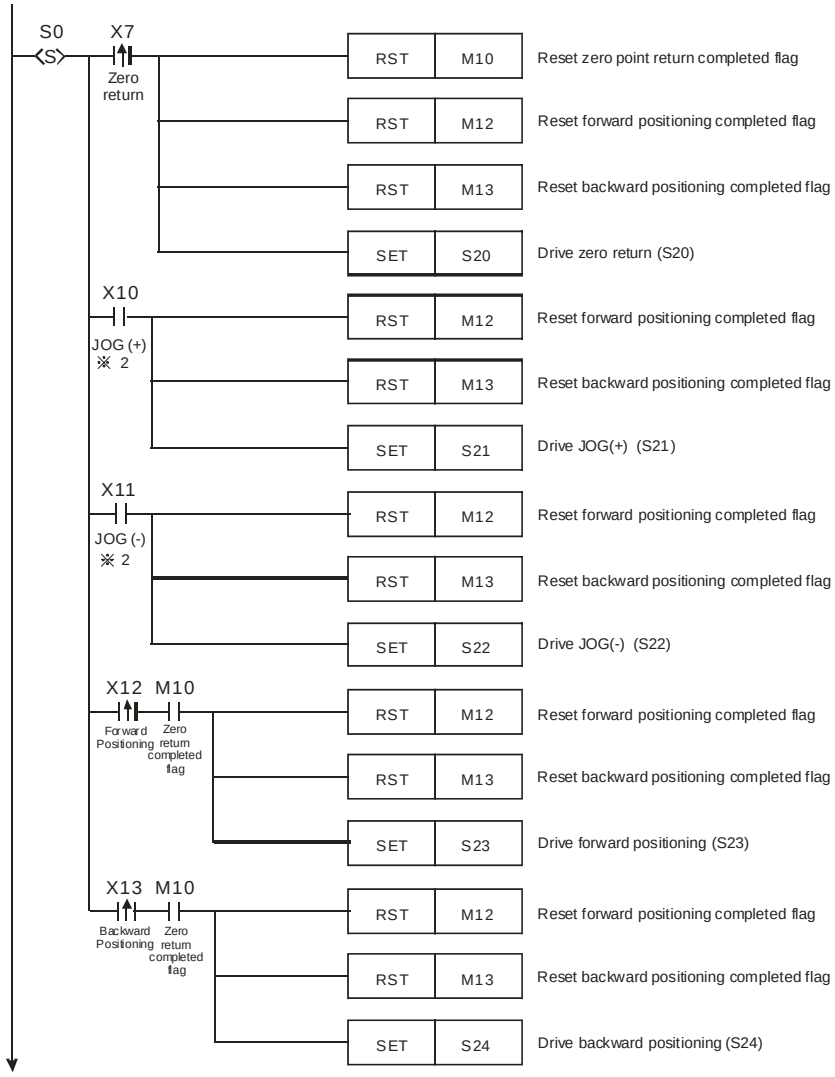
One operation mode performs positioning by absolute position:



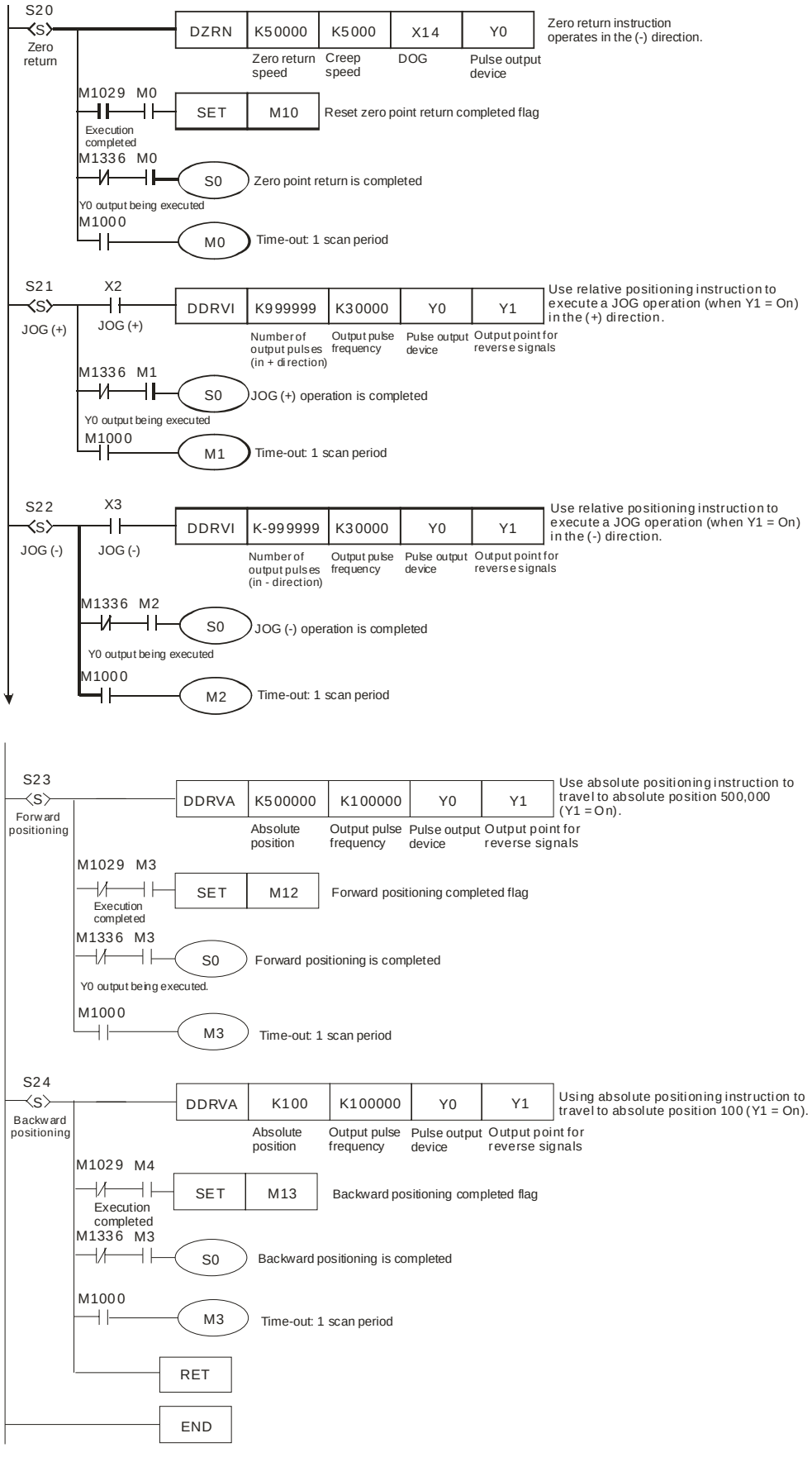
11. Programming example of using step ladder instruction (STL):



※1. If the accel./decel. time (D1343) of CH1 can be default setting, (100ms) this program step can be ignored.



※2. The max. traveling distance of a JOG operation equals to the max. number of output pulses (-2,147,483,648 ~ +2,147,483,647) of API 158 DDRVI instruction. Please re-execute JOG of the traveling distance exceeds the range.



API	Mnemonic		Operands	Function	Controllers
160	TCMP	P	(S1) (S2) (S3) (S) (D)	Time Compare	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	TCMP, TCMPP: 11 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	
S_2					*	*	*	*	*	*	*	*	*	*	*	
S_3					*	*	*	*	*	*	*	*	*	*	*	
S											*	*	*			
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

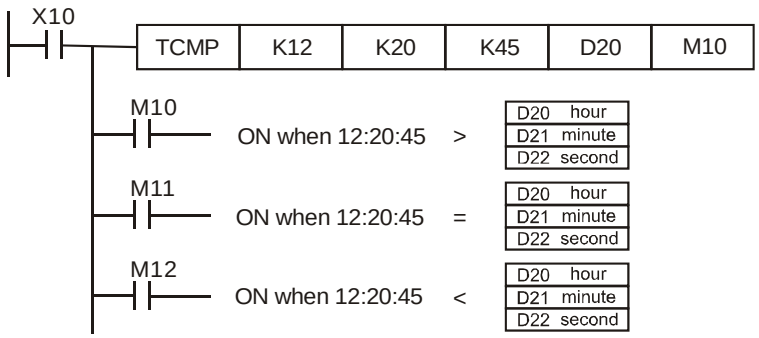
S_1: "Hour" for comparison **S_2**: "Minute" for comparison **S_3**: "Second" for comparison **S**: Current time of RTC **D**: Comparison result

Explanations:

1. Range of **S_1**: K0 ~ K23; range of **S_2** and **S_3**: K0 ~ K59
2. **S** will occupy 3 consecutive devices; **D** will occupy 3 consecutive points.
3. See the specifications of each model for their range of use.
4. **S_1**, **S_2** and **S_3** are compared with the present values of "hour", "minute" and "second" starting from **S**. The comparison result is stored in **D**.
5. **S** is the "hour" of the current time (K0 ~ K23) in RTC; **S** + 1 is the "minute" (K0 ~ K59) and **S** + 2 is the "second" (K0 ~ K59).
6. **S** is read by TRD instruction and the comparison is started by TCMP instruction. If **S** exceeds the range, the program will regard this as an operation error and the instruction will not be executed, M1067 and M1068 = On and D1067 will record the error code 0E1A (hex).

Program Example:

1. When X10= On, the instruction will compare the current time in RTC (D20 ~ D22) with the set value 12:20:45 and display the result in M10 ~ M12. When X10 goes from On to Off, the instruction will not be executed, but the On/Off stauts prior to M10 ~ M12 will remain.
2. Connect M10 ~ M12 in series or in parallel to obtain the result of ≧, ≦, and ≠.



API	Mnemonic		Operands	Function	Controllers
161	TZCP	P	(S1) (S2) (S) (D)	Time Zone Compare	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1											*	*	*			TZCP, TZCPP: 9 steps
S_2											*	*	*			
S											*	*	*			
D		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

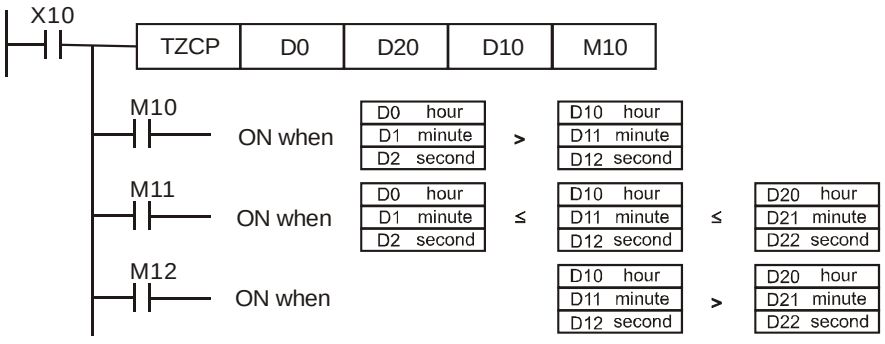
S_1: Lower bound of the time for comparison **S_2**: Upper bound of the time for comparison **S**: Current time of RTC **D**: Comparison result

Explanations:

1. **S_1**, **S_2**, and **S** will occupy 3 consecutive devices.
2. The content in **S_1** must be less than the content in **S_2**.
3. **D** will occupy 3 consecutive points.
4. See the specifications of each model for their range of use.
5. **S** is compared with **S_1** and **S_2**. The comparsion result is stored in **D**.
6. **S_1** , **S_1** +1, **S_1** +2: The “hour”, “minute” and “second” of the lower bound of the time for comparison.
7. **S_2**, **S_2** +1, **S_2** +2: The “hour”, “minute” and “second”ond” of the upper bound of the time for comparison.
8. **S** , **S** +1, **S** +2: The “hour”, “minute” and “second” of the current time of RTC.
9. D0 designated by **S** is read by TRD instruction and the comparison is started by TZCP instruction. If **S_1**, **S_2**, and **S** exceed their ranges, the program will regard this as an operation error and the instruction will not be executed, M1067 and M1068 = On and D1067 will record the error code 0E1A (hex).
10. When **S** < **S_1** and **S** < **S_2**, **D** will be On. When **S** > **S_1** and **S** > **S_2**, **D** + 2 will be On. In other occasions, **D** + 1 will be On.

Program Example:

When X10= On, TZCP instruction will be executed and one of M10 ~ M12 will be On. When X10 = Off, TZCP instruction will not be executed and the status of M10 ~ M12 prior to X10 = Off will remain unchanged.



API	Mnemonic		Operands	Function	Controllers
162	TADD	P	(S1) (S2) (D)	Time Addition	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1											*	*	*			TADD, TADDP: 7 steps
S_2											*	*	*			
D											*	*	*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

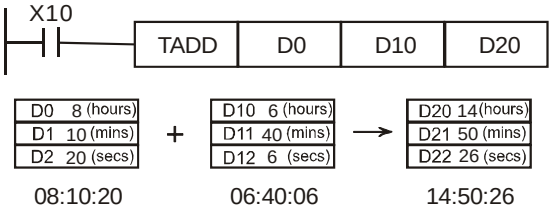
S_1: Time summand **S_2**: Time addend **D**: Time sum

Explanations:

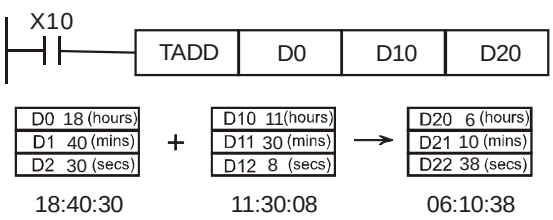
1. **S_1**, **S_2**, and **D** will occupy 3 consecutive devices.
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag); M1022 (carry flag)
4. **S_1** + **S_2** = **D**. The hour, minute, and second of the RTC designated in **S_1** plus the hour, minute, and second designated in **S_2**. The result is stored in the hour, minute, and second of the register designated in **D**.
5. If **S_1** and **S_2** exceed their ranges, the program will regard this as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 record the error code 0E1A (hex).
6. If the sum is larger than 24 hours, the carry flag M1022 will be On and the value in **D** will be the result of "sum minuses 24 hours".
7. If the sum equals 0 (00:00:00), the zero flag M1020 will be On.

Program Example:

1. When X10= On, TADD instruction will be executed and the hour, minute and second in RTC designated in D0 ~ D2 will plus the hour, minute and second in RTC designated in D10 ~ D12. The sum is stored in the hour, minute and second of the register designated in D20 ~ D22.



2. If the sum is larger than 24 hours, M1022 will be On.



API	Mnemonic	Operands	Function	Controllers
163	TSUB P	S_1 S_2 D	Time Subtraction	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1											*	*	*			TSUB, TSUBP: 7 steps
S_2											*	*	*			
D											*	*	*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

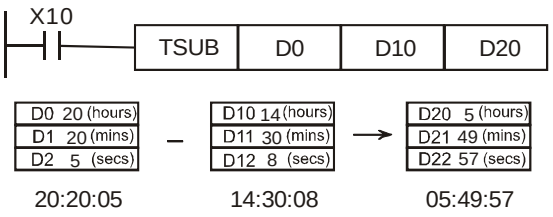
S_1: Time minuend **S_2**: Time subtrahend **D**: Time remainder

Explanations:

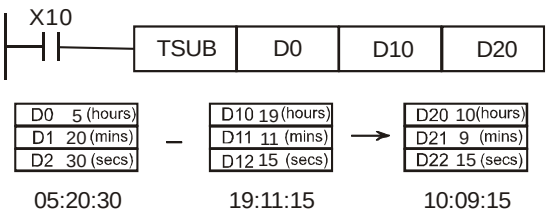
1. **S_1**, **S_2**, and **D** will occupy 3 consecutive devices.
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag); M1021 (borrow flag)
4. **S_1** − **S_2** = **D**. The hour, minute, and second of the RTC designated in **S_1** minus the hour, minute, and second designated in **S_2**. The result is stored in the hour, minute, and second of the register designated in **D**.
5. If **S_1** and **S_2** exceed their ranges, the program will regard this as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 record the error code 0E1A (hex).
6. If the remainder is a negative value, the borrow flag M1021 will be On. The value in **D** will be the result of "the negative value pluses 24 hours".
7. If the remainder equals 0 (00:00:00), the zero flag M1020 will be On.

Program Example:

1. When X10= On, TADD instruction will be executed and the hour, minute and second in RTC designated in D0 ~ D2 will minus the hour, minute and second in RTC designated in D10 ~ D12. The remainder is stored in the hour, minute and second of the register designated in D20 ~ D22.



2. If the subtraction result is a negative value, M1021 will be On.



API	Mnemonic	Operands	Function	Controllers
166	TRD P	D	Time Read	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
D											*	*	*			TRD, TRDP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

D: The device for storing the current time read in RTC

Explanations:

1. **D** will occupy 7 consecutive devices.
2. See the specifications of each model for their range of use.
3. Flags: M1016, M1017, M1076. See remarks for more details.
4. The built-in RTC in EH/EH2/SV/SA/SX/SC series MPU offers 7 data (year, week, month, day, hour, minute, second) stored in D1319 ~ D1313. TRD instruction is for program designers to read the current data in RTC and store the data to the 7 registers designated.
5. D1319 only reads the 2-digit year in A.D. If you wish D1319 to read the 4-digit year, see remarks for more information.

Program Example:

1. When X0 = On, the instruction will read the current time in RTC to the designated registers D0 ~ D6.
2. The content of D1318: 1 = Monday; 2 = Tuesday … 7 = Sunday.

Special D	Item	Content		General D	Item
D1319	Year (A.D.)	00~99	→	D0	Year (A.D.)
D1318	Day (Mon ~ Sun)	1~7	→	D1	Day (Mon ~ Sun)
D1317	Month	1~12	→	D2	Month
D1316	Day	1~31	→	D3	Date
D1315	Hour	0~23	→	D4	Hour
D1314	Minute	0~59	→	D5	Minute
D1313	Second	0~59	→	D6	Second

Remarks:

1. Flags and special registers for the built-in RTC in EH/EH2/SV/SA/SX/SC series MPU.

Device	Name	Function
M1016	Displaying year in A.D. in RTC	When Off, D1319 will display 2-digit year in A.D. When On, D1319 will display “2-digit year in A.D + 2,000”.
M1017	±30 seconds correction	Correction takes place when M1017 goes from Off to On (reset to 0 when in 0 ~ 29 second; minute pluses 1 and second resets to 0 in 30 ~ 59 second)
M1076	Malfunction of RTC	On when the set value exceeds the range. (only available when the power is being switched on).
D1313	Second	0 ~ 59
D1314	Minue	0 ~ 59
D1315	Hour	0 ~ 23
D1316	Day	1 ~ 31
D1317	Month	1 ~ 12
D1318	Week	1 ~ 7
D1319	Year	0 ~ 99 (2-digit year in A.D.)

2. How to correct RTC:

   There are 2 ways to correct the built-in RTC.

   a) By a specific instruction. (See API 167 TWR instruction)

   b) By peripheral devices, WPLSoft, the ladder diagram editing software.

3. How to display 4-digit year in A.D.:

   a) Normally, the year is only displayed in 2 digits (e.g. 2003 displayed as 03). If you wish the year to be displayed in 4 digits, please key in the following program at the start of the program.

   b) The original 2-digit year will be switched to a 4-digit year, i.e. the 2-digit year will pluses 2,000.

   c) If you wish to write in new time in the 4-digit year display mode, you can only write in a 2-digit year (0 ~ 99, indicating year 2000 ~ 2099). For example, 00=year 2000, 50=year 2050 and 99=year 2099.

API	Mnemonic	Operands	Function	Controllers
167	TWR P	S	Time Write	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S											*	*	*			TWR, TWRP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Device for storing the new time to be written into RTC

Explanations:

1. **S** will occupy 7 consecutive devices.
2. See the specifications of each model for their range of use.
3. Flags: M1016, M1017, M1076. See remarks of API 166 TRD for more details.
4. To make adjustment on the RTC built in EH/EH2/SV/SA/SX/SC series MPU, use this instruction to write the correct time into the RTC.
5. When this instruction is executed, the new set time will be written in the RTC built in PLC immediately. Therefore, please be noted that the new set time has to match the current time then when the instruction is executed.
6. If **S** exceeds its range, the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code 0E1A (hex).

Program Example 1:

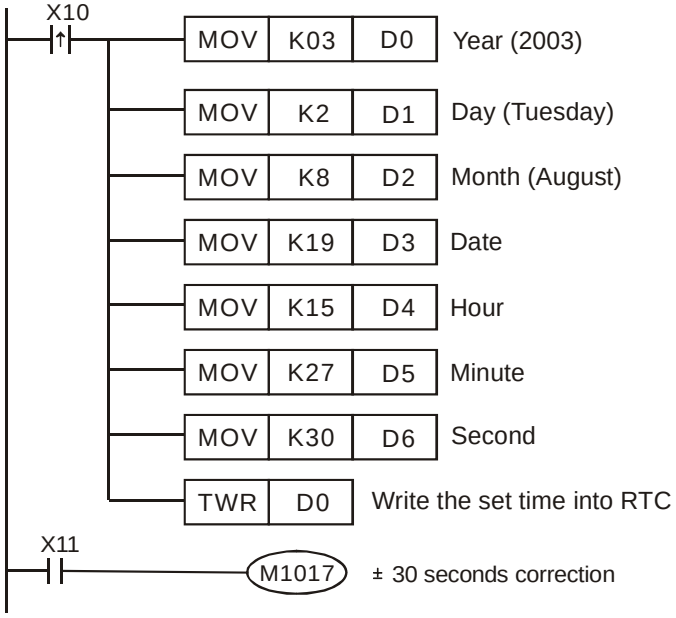
When X0= On, write the correct current time into the RTC.

	General D	Item	Content		Special D	Item	
New set time	D20	Year (A.D.)	00~99	→	D1319	Year (A.D.)	Real Time Clock
	D21	Day (Mon ~ Sun)	1~7	→	D1318	Day (Mon ~ Sun)	
	D22	Month	1~12	→	D1317	Month	
	D23	Date	1~31	→	D1316	Date	
	D24	Hour	0~23	→	D1315	Hour	
	D25	Minute	0~59	→	D1314	Minute	
	D26	Second	0~59	→	D1313	Second	

Program Example 2:

1. Set the current time in the RTC as 15:27:30, Tuesday, August 19, 2003.
2. D0 ~ D6 indicate the new set time in the RTC.
3. X10 = On for changing the current time in the RTC and make the changed value the new set value.

4. Whenever X11 = On, RTC will perform a ±30 second correction. The correction is performed according to the rules: When the second hand of RTC locates at 1 ~ 29, the second will be automatically reset to “0” and the minute hand will remain at its location. When the second hand locates at 30 ~ 59, the second will be automatically reset to “0” and the minute hand will increase by 1 minute.



API	Mnemonic	Operands	Function	Controllers
169	D HOUR	(S) (D1) (D2)	Hour Meter	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	HOUR: 7 steps DHOUR: 13 steps
D_1													*			
D_2		*	*	*												

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

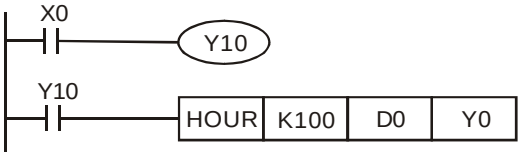
S: Period of time when $\mathbf{D_2}$ is On (in hour) $\mathbf{D_1}$: Current value being measured (in hour) $\mathbf{D_2}$: Output device

Explanations:

1. If **S** is used in device F, only 16-bit instruction is applicable.
2. $\mathbf{D_1}$ will occupy 2 consecutive points. $\mathbf{D_1}$ + 1 uses 16-bit register in 16-bit or 32-bit instruction.
3. See the specifications of each model for their range of use.
4. HOUR instruction can be used 4 times of SA/SX/SC.
5. Range of **S**: K1 ~ K32,767 (unit: hour); range of $\mathbf{D_1}$: K0 ~ K32,767 (unit: hour). $\mathbf{D_1}$ + 1 refers to the current time that is less than an hour (range: K0 ~K3,599; unit: second).
6. This instruction times the time and when the time reaches the set time (in hour), $\mathbf{D_2}$ will be On. This function allows the user to time the operation of the machine or conduct maintenance works.
7. After $\mathbf{D_2}$ is On, the timer will resume the timing.
8. In the 16-bit instruction, when the current time measured reaches the maximum 32,767 hours/3,599 seconds, the timing will stop. To restart the timing, $\mathbf{D_1}$ and $\mathbf{D_1}$ + 1 have to be reset to “0”.
9. In the 32-bit instruction, when the current time measured reaches the maximum 2,147,483,647 hours/3,599 seconds, the timing will stop. To restart the timing, $\mathbf{D_1}$ ~ $\mathbf{D_1}$ + 2 have to be reset to “0”.
10. There is no limitations on the times of using this instruction in the program for EH series MPU; however, only 4 instructions can be executed at the same time.

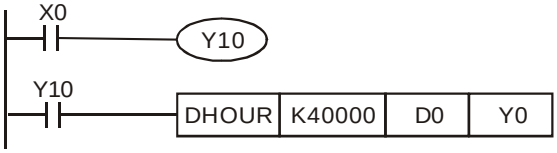
Program Example 1:

In 16-bit instruction, when X0 = On, Y10 will be On and the timing will start. When the timing reaches 100 hours, Y0 will be On and D0 will record the current time measured (in hour) and D1 will record the current time that is less than an hour (0 ~ 3,599; unit: second).



Program Example 2:

In 32-bit instruction, when X0 = On, Y10 will be On and the timing will start. When the timing reaches 40,000 hours, Y0 will be On. D1 and D0 will record the current time measured (in hour) and D2 will record the current time that is less than an hour (0 ~ 3,599; unit: second).



API	Mnemonic			Operands	Function	Controllers
170	D	GRY	P	(S) (D)	BIN → Gray Code	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	GRY, GRYP: 5 steps DGRY, DGRYP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

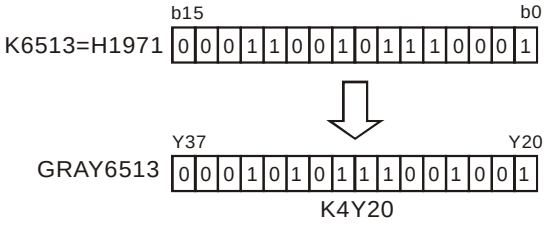
S: Source device for BIN value **D**: Device for storing Gray code

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. This instruction converts the BIN value in the device designated in **S** into Gray code and stores the value in **D**.
4. See the ranges of **S** as indicated below. If **S** exceeds the ranges, the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code 0E1A (hex).
   In 16-bit instruction: 0 ~ 32,767
   In 32-bit instruction: 0 ~ 2,147,483,647

Program Example:

When X0 = On, the instruction will convert constant K6,513 into Gray code and store the result in K4Y20.



API	Mnemonic			Operands	Function	Controllers
171	D	GBIN	P	(S) (D)	Gray Code → BIN	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*	*	*	*	*	*	*	*	*	*	GBIN, GBINP: 5 steps DGBIN, DGBINP: 9 steps
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

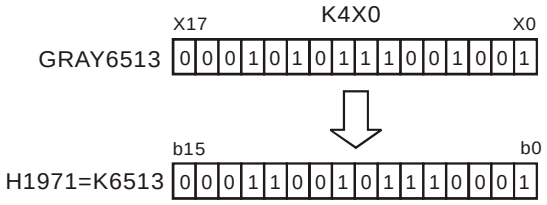
S: Source device for Gray code **D**: Device for storing BIN value

Explanations:

1. If **S** and **D** are used in device F, only 16-bit instruction is applicable.
2. See the specifications of each model for their range of use.
3. This instruction converts the Gray code in the device designated in **S** into BIN value and stores the value in **D**.
4. This instruction converts the content (in Gray code) in the absolute position encoder connected at the PLC input terminal into BIN value and store the result in the designated register.
5. See the ranges of **S** as indicated below. If **S** exceeds the ranges, the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code 0E1A (hex).
   In 16-bit instruction: 0 ~ 32,767
   In 32-bit instruction: 0 ~ 2,147,483,647

Program Example:

When X20 = On, the Gray code in the absolute position encoder connected at X0 ~ X17 will be converted into BIN value and stored in D10.



API	Mnemonic	Operands	Function	Controllers
172	D ADDR P	(S1) (S2) (D)	Floating Point Addition	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1													*			DADDR, DADDRP: 13 steps
S_2													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

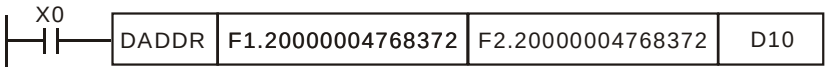
S_1: Floating point summand **S_2**: Floating point addend **D**: Sum

Explanations:

1. **S_1** and **S_2** can be floating point values (FX.XX).
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)
4. In DADDR instruction, floating point values (e.g. F1.2) can be entered directly into **S_1** and **S_2** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.
5. When **S_1** and **S_2** stores the floating point values in register D, their functions are the same as API 120 EADD.
6. **S_1** and **S_2** can designate the same register. In this case, if the "continuous execution" type instruction is in use and during the On period of the drive contact, the register will be added once in every scan by a "pulse execution" type instruction (DADDRP).
7. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.
8. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.
9. If the operation result is "0", the zero flag M1020 will be On.

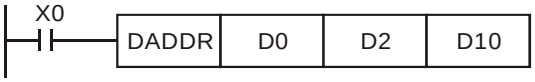
Program Example 1:

When X0 = On, the floating point F1.20000004768372 will plus F2.20000004768372 and the result F3.40000009536743 will be stored in the data registers (D10, D11).



Program Example 2:

When X0 = On, the floating point value (D1, D0) + floating point value (D3, D2) and the result will be stored in the registers designated in (D11, D10).



Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series.

DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic	Operands	Function	Controllers
173	D SUBR P	(S1) (S2) (D)	Floating Point Subtraction	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DSUBR, DSUBRP: 13 steps
S_1													*			
S_2													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

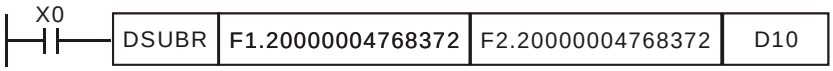
S_1: Floating point minuend **S_2**: Floating point subtrahend **D**: Remainder

Explanations:

1. **S_1** and **S_2** can be floating point values (FX.XX).
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)
4. In DSUBR instruction, floating point values (e.g. F1.2) can be entered directly into **S_1** and **S_2** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.
5. When **S_1** and **S_2** stores the floating point values in register D, their functions are the same as API 121 ESUB.
6. **S_1** and **S_2** can designate the same register. In this case, if the "continuous execution" type instruction is in use and during the On period of the drive contact, the register will be subtracted once in every scan by a "pulse execution" type instruction (DSUBRP).
7. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.
8. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.
9. If the operation result is "0", the zero flag M1020 will be On.

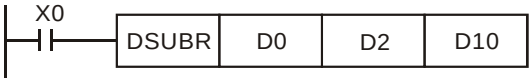
Program Example 1:

When X0 = On, the floating point F1.20000004768372 will minus F2.20000004768372 and the result F-1 will be stored in the data registers (D10, D11).



Program Example 2:

When X0 = On, the floating point value (D1, D0) – floating point value (D3, D2) and the result will be stored in the registers designated in (D11, D10).



Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series.

DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic			Operands	Function	Controllers
174	D	MULR	P	(S1) (S2) (D)	Floating Point Multiplication	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DMULR, DMULRP: 13 steps
S_1													*			
S_2													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

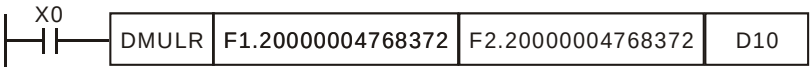
S_1: Floating point multiplicand **S_2**: Floating point multiplicator **D**: Product

Explanations:

1. **S_1** and **S_2** can be floating point values (FX.XX).
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)
4. In DMULR instruction, floating point values (e.g. F1.2) can be entered directly into **S_1** and **S_2** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.
5. When **S_1** and **S_2** stores the floating point values in register D, their functions are the same as API 122 EMUL.
6. **S_1** and **S_2** can designate the same register. In this case, if the "continuous execution" type instruction is in use and during the On period of the drive contact, the register will be multiplied once in every scan by a "pulse execution" type instruction (DMULRP).
10. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.
11. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.
12. If the operation result is "0", the zero flag M1020 will be On.

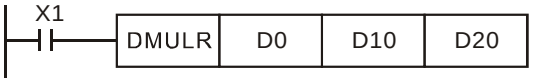
Program Example 1:

When X0 = On, the floating point F1.20000004768372 will multiply F2.20000004768372 and the result F2.64000010490417 will be stored in the data registers (D10, D11).



Program Example 2:

When X1 = On, the floating point value (D1, D0) × floating point value (D11, D10) and the result will be stored in the registers designated in (D21, D20).



Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series. DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic			Operands	Function	Controllers
175	D	DIVR	P	S_1 S_2 D	Floating Point Division	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DDIVR, DDIVRP: 13 steps
S_1													*			
S_2													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

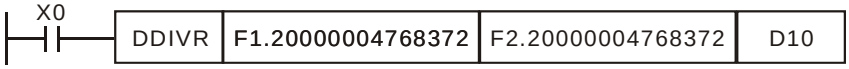
S_1: Floating point dividend **S_2**: Floating point divisor **D**: Quotient

Explanations:

1. **S_1** and **S_2** can be floating point values.
2. See the specifications of each model for their range of use.
3. Flags: M1020 (zero flag), M1021 (borrow flag), M1022 (carry flag)
4. In DDIVR instruction, floating point values (e.g. F1.2) can be entered directly into **S_1** and **S_2** or stored in register D for operation. When the instruction is being executed, operand **D** will store the operation result.
5. When **S_1** and **S_2** stores the floating point values in register D, their functions are the same as API 123 EDIV.
6. If **S_2** is "0", the program will regard it as an operation error and the instruction will not be executed. M1067 and M1068 will be On and D1067 will record the error code H'0E19.
7. If the absolute value of the operation result is larger than the maximum floating point displayable, the carry flag M1022 will be On.
8. If the absolute value of the operation result is smaller than the minimum floating point displayable, the borrow flag M1021 will be On.
9. If the operation result is "0", the zero flag M1020 will be On.

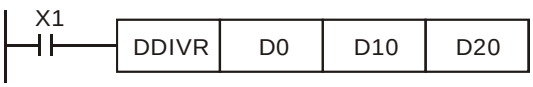
Program Example 1:

When X0 = On, the floating point F1.20000004768372 will be divided by F2.20000004768372 and the result F0.545454561710358 will be stored in the data registers (D10, D11).



Program Example 2:

When X1 = On, the floating point value (D1, D0) ÷ floating point value (D11, D10) and the quotient will be stored in the registers designated in (D21, D20).



Remarks:

The functions of this instruction are in V6.6 of ES/EX/SS series, V1.6 of SA/SX series and V1.4 of SC series.

DADDR instruction supports EH2/SV series, but not EH series.

API	Mnemonic		Operands	Function	Controllers
180	MAND	P	(S1) (S2) (D) (n)	Matrix 'AND' Operation	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1							*	*	*	*	*	*	*			MAND, MANDP: 9 steps
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

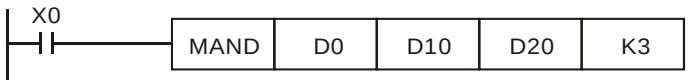
S_1: Matrix source device 1 **S_2**: Matrix source device 2 **D**: Operation result **n**: Array length

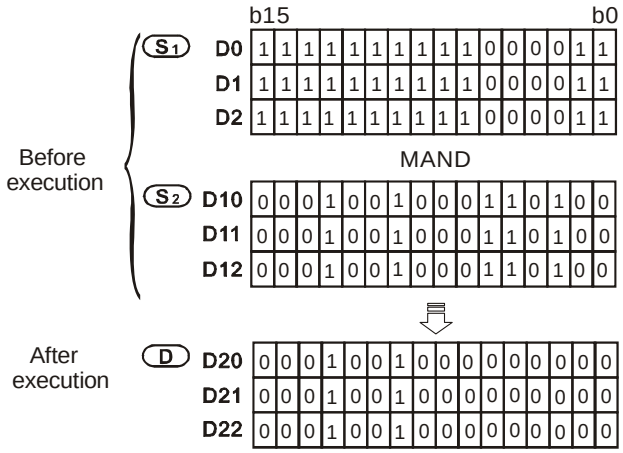
Explanations:

1. Range of **n**: K1 ~ K256
2. **S_1**, and **S_2** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. The two matrix sources **S_1** and **S_2** perform matrix 'AND' operation according to the array length **n**. The result is stored in **D**.
6. Operation rule of matix 'AND' : The result will be 1 if both two bits are 1; otherwise the result will be 0.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'AND' operation. The result will be stored in the 3 arrays of 16-bitd registers D20 ~ D22.





Remarks:

1. Explanations on the matrix instruction:
   a) A matix consists of more than 1 consecutive 16-bit registers. The number of registers in the matrix is the

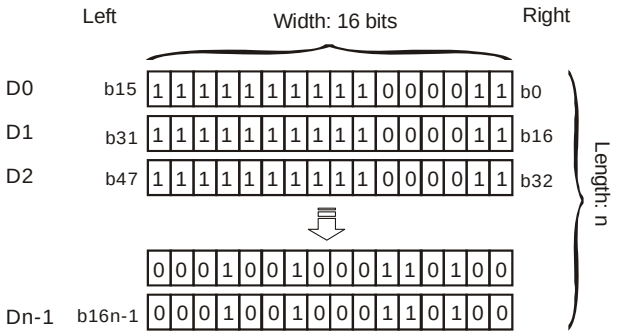
length of the array (n). A matrix contains 16 × n bits (points) and there is only 1 bit (point) offered for an operand at a time.

b) The matrix instruction gathers a series of 16 × n bits (b_0 ~ b_{16n-1}) and designates a single point for operation. The point will not be seen as a value.

c) The matrix instruction processes the moving, copying, comparing and searching of one-to-many or many-to-many matrix status, which is a very handy and important application instruction.

d) The matrix operation will need a 16-bit register to designate a point among the 16n points in the matrix for the operation. The register is the Pointer (Pr) of the matrix, designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix.

e) There are left displacement, right displacement and rotation in a matrix operation. The bit number decreases from left to right (see the figure below).



f) The matrix width (C) is fixed at 16 bits.

g) Pr: matrix pointer. E.g. if Pr is 15, the designated point will be b15.

h) Array length (R) is n: n = 1 ~ 256.

Example: The matrix is composed of D0, n = 3; D0 = HAAAA, D1 = H5555, D2 = HAAFF

	C_{15}	C_{14}	C_{13}	C_{12}	C_{11}	C_{10}	C_9	C_8	C_7	C_6	C_5	C_4	C_3	C_2	C_1	C_0	
R_0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	**D0**
R_1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	**D1**
R_2	1	0	1	0	1	0	1	0	1	1	1	1	1	1	1	1	**D2**

Example: The matrix is composed of K2X0, n = 3; K2X0 = H37, K2X10 = H68, K2X20 = H45

	C_{15}	C_{14}	C_{13}	C_{12}	C_{11}	C_{10}	C_9	C_8	C_7	C_6	C_5	C_4	C_3	C_2	C_1	C_0	
R_0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	X_0~X_7
R_1	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	X_{10}~X_{17}
R_2	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	X_{20}~X_{27}

Fill “0” into the blank in R0(C_{15}-C_8), R1(C_{15}-C_8), and R2(C_{15}-C_8).

API	Mnemonic		Operands	Function	Controllers
181	MOR	P	(S1) (S2) (D) (n)	Matrix 'OR' Operation	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1							*	*	*	*	*	*	*			MOR, MORP: 9 steps
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

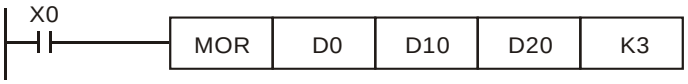
S_1: Matrix source device 1 **S_2**: Matrix source device 2. **D**: Operation result **n**: Array length

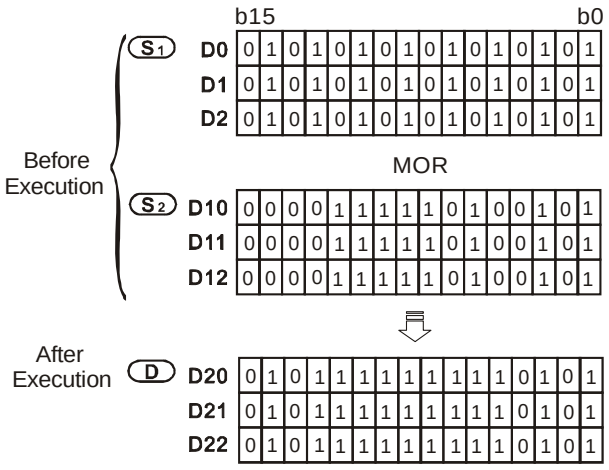
Explanations:

1. Range of **n**: K1 ~ K256
2. **S_1**, and **S_2** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. The two matrix sources **S_1** and **S_2** perform matrix 'OR' operation according to the array length **n**. The result is stored in **D**.
6. Operation rule of matrix 'OR': The result will be 1 if either of the two bits is 1. The result is 0 only when both two bits are 0.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'OR' operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.





API	Mnemonic		Operands	Function	Controllers
182	MXOR	P	S_1 S_2 D n	Matrix 'XOR' Operation	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1							*	*	*	*	*	*	*			MXOR, MXORP: 9 steps
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

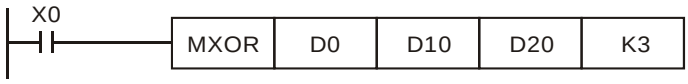
S_1: Matrix source device 1 **S_2**: Matrix source device 2 **D**: Operation result **n**: Array length

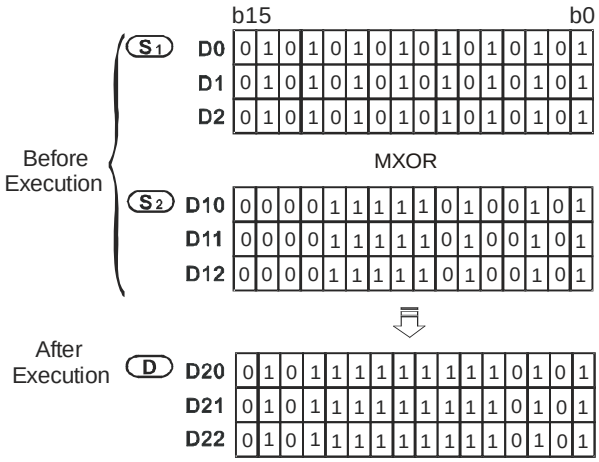
Explanations:

1. Range of **n**: K1 ~ K256
2. **S_1**, and **S_2** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. The two matrix sources **S_1** and **S_2** perform matrix 'XOR' operation according to the array length **n**. The result is stored in **D**.
6. Operation rule of matrix 'XOR': The result will be 1 if the two bits are different. The result will be 0 if the two bits are the same.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'XOR' operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.





API	Mnemonic		Operands	Function	Controllers
183	MXNR	P	S_1 S_2 D n	Matrix 'XNR' Operation	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MXNR, MXNRP: 9 steps
S_1							*	*	*	*	*	*	*			
S_2							*	*	*	*	*	*	*			
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

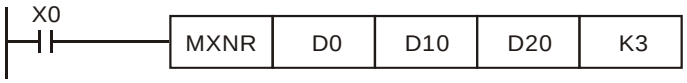
S_1: Matrix source device 1 **S_2**: Matrix source device 2 **D**: Operation result **n**: Array length

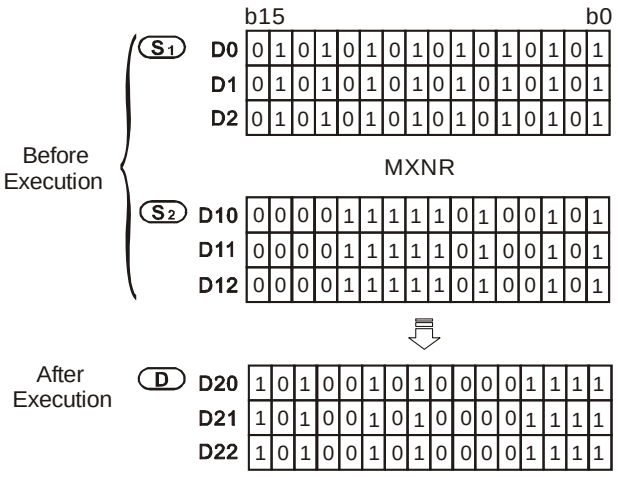
Explanations:

1. Range of **n**: K1 ~ K256
2. **S_1**, and **S_2** designate KnX, KnY, KnM and KnS; **D** designates KnYm KnM and KnS
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. The two matrix sources **S_1** and **S_2** perform matrix 'XNR' operation according to the array length **n**. The result is stored in **D**.
6. Operation rule of matrix 'XNR': The result will be 1 if the two bits are the same. The result will be 0 if the two bits are different.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 and the 3 arrays of 16-bit registers D10 ~ D12 will perform a matrix 'XNR' operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.





API	Mnemonic		Operands	Function	Controllers
184	MINV	P	(S) (D) (n)	Matrix Inverse Operation	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			MINV, MINVP: 7 steps
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

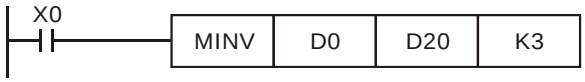
S: Matrix source device **D**: Operation result **n**: Array length

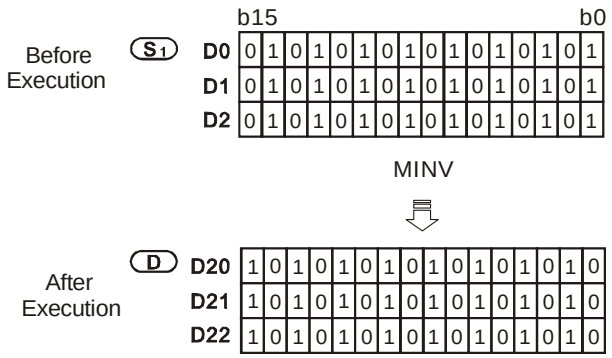
Explanations:

1. Range of **n**: K1 ~ K256
2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n ≦ 4.
4. See the specifications of each model for their range of use.
5. **S** performs an inverse matrix operation according to the array length **n**. The result is stored in **D**.

Program Example:

When X0 = On, the 3 arrays of 16-bit registers D0 ~ D2 perform a matrix inverse operation. The result will be stored in the 3 arrays of 16-bit registers D20 ~ D22.





API	Mnemonic		Operands	Function	Controllers
185	MCMP	P	S_1 S_2 n D	Matrix Compare	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1							*	*	*	*	*	*	*			MCMP, MCMPP: 9 steps
S_2							*	*	*	*	*	*	*			
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Matrix source device 1 **S_2**: Matrix source device 2 **n**: Array length

D: Pointer (Pr), for storing the value of target location

Explanations:

1. Range of **n**: K1 ~ K256
2. **S_1**, and **S_2** designate KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. Flags: M1088 ~ M1092. See remarks for more details.
6. This instruction compares every bit in **S_1** with every bit in **S_2** starting from location **D** and finds out the location of different bits. The location will be stored in **D**.
7. The matrix comparison flag (M1088) decides to compare between equivalent values (M1088 = 1) or different values (M1088 = 0). When the comparison is completed, it will stop immediately and the matrix bit search flag will turn “On” (M1091 = 1). When the comparison progresses to the last bit, the matrix search end flag (M1089) will turn “On” and the No. where the comparison is completed is stored in **D**. The comparison will start from the 0^{th} bit in the next scan period and the matrix search start flag will turn “On” (M1090 = 1). When **D** exceeds the range, the pointer error flag will turn “On” (M1092 = 1).
8. The matrix operation will need a 16-bit register to designate a point among the 16n points in the matrix for the operation. The register is the Pointer (Pr) of the matrix, designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b0 ~ b16n-1 in the matrix. Please avoid changing the Pr value during the operation in case the comparing and searching will not ne correct. If the Pr value exceeds its range, M1092 will be On and the instruction will not be executed.
9. When M1089 and M1091 take place at the same time, both flags will be “1” at the same time.

Program Example:

1. When X0 goes from Off to On, the matrix search start falg M1090 = 0. The searching will start from the bit marked with “*” (current Pr value +1) for bits of different status (M1088 = 0).

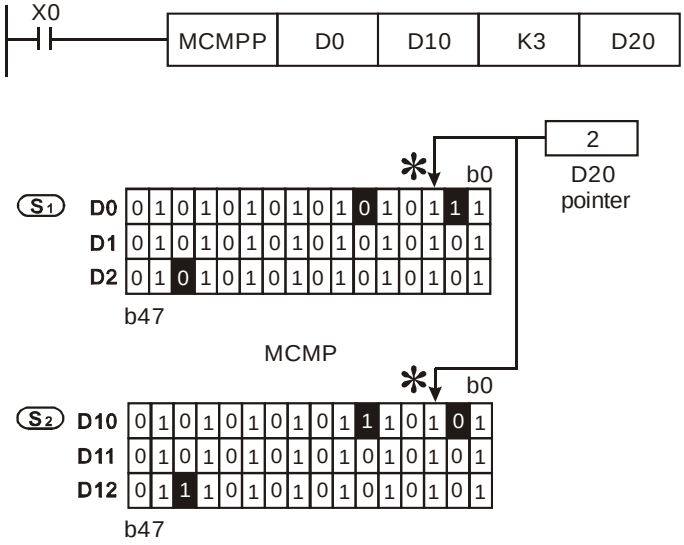
2. Set the Pr value D20 = 2. When X0 goes from Off to On for 4 times, we can obtain the 4 execution results ❶, ❷, ❸, ❹.

   ❶ D20 = 5, M1091 = 1, M1089 = 0.

   ❷ D20 = 45, M1091 = 1, M1089 = 0.

❸ D20 = 47, M1091 = 0, M1089 = 1.

❹ D20 = 1, M1091 = 1, M1089 = 0.



Remarks:

Flags explanations:

Flags	Function
M1088	Matrix comparison flag. Comparing between equivalent values (M1088 = 1) or different values (M1088 = 0).
M1089	Matrix search end flag. When the comparison reaches the last bit, M1089 = 1.
M1090	Matrix search start flag. Comparing from bit 0 (M1090 = 1).
M1091	Matrix bit search flag. When the comparison is completed, the comparison will stop immediately (M1091=1).
M1092	Matrix pointer error flag. When the pointer Pr exceeds its range, M1092 = 1.

API	Mnemonic		Operands	Function	Controllers
186	MBRD	P	(S) (n) (D)	Read Matrix Bit	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			MBRD, MBRDP: 7 steps
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Matrix source device **n**: Array length **D**: Pointer (Pr), for storing the value of target location

Explanations:

1. Range of **n**: K1 ~ K256
2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. Flags: M1089 ~ M1095. See remarks for more details.
6. When this instruction is executed, it first determines if the matrix pointer clear flag (M1094) is “On”. If it is “On”, pointer D is cleared as 0. The instruction then reads the On/Off status from the 0^{th} bit of **S** to the matrix rotation/displacement/output carry flag (M1095). Whenever finishing reading 1 bit, the instruction determines whether the matrix pointer increasing flag (M1093) is “On”. If it is “On”, the value of pointer D will plus 1. When the reading is processed to the last bit, the matrix search end flag (M1089) will turn “On” and pointer D record the No. of read bits.
7. The Pointer (Pr) of the matrix is designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b_0 ~ b_{16n-1} in the matrix. If the Pr value exceeds its range, M1092 will be On and the instruction will not be executed.

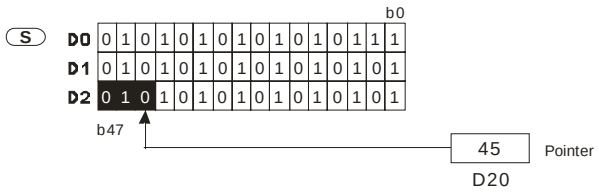
Program Example:

1. When X0 goes from Off to On, M1094 will be set to “0” and M1093 to “1”. Therefore, the Pr will plus 1 after every reading.
2. Set the Pr value D20 = 45. When X0 goes from Off to On for 3 times, we can obtain the 3 execution results ❶, ❷, ❸.

   ❶ D20 = 46, M1095 = 0, M1089 = 0.

   ❷ D20 = 47, M1095 = 1, M1089 = 0.

   ❸ D20 = 47, M1095 = 1, M1089 = 1.



Remarks:

Flag explanations:

Flags	Function
M1088	Matrix search end flag. When the comparison reaches the last bit, M1089 = 1.
M1092	Matrix pointer error flag. When the pointer Pr exceeds its range, M1092 = 1.
M1093	Matrix pointer increasing flag. Adding 1 to the current value of the Pr.
M1094	Matrix pointer clear flag. Clearing the current value of the Pr to 0.
M1095	Matrix rotation/displacement/output carry flag.

API	Mnemonic		Operands	Function	Controllers
187	MBWR	P	(S) (n) (D)	Write Matrix Bit	ES/EX/SS | SA/SX/SC | EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			MBWR, MBWRP: 7 steps
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S: Matrix source device **n**: Array length **D**: Pointer (Pr), for storing the value of target location

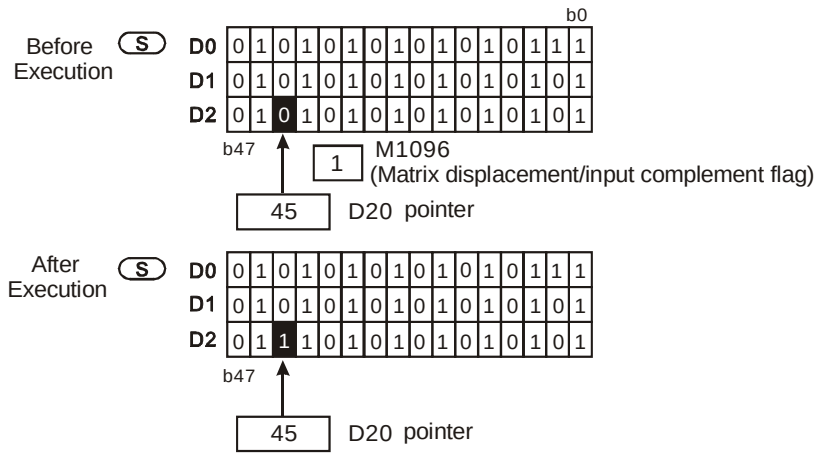
Explanations:

1. Range of **n**: K1 ~ K256
2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. Flags: M1089 ~ M1096. See remarks for more details.
6. When this instruction is executed, if first determines if the matrix pointer clear flag (M1094) is “On”, If it is “On”, pointer D is cleared as 0. The instruction then writes the value in the matrix displacement/input complement flag (M1096) into the location starting from the 0^{th} bit of **S**. Whenever finishing writing 1 bit, the instruction determines whether the matrix pointer increasing flag (M1093) is “On”. If it is “On”, the value of pointer D will plus 1. When the writing is processed to the last bit, the matrix search end flag (M1089) will turn “On” and pointer D records the No. of written bits. If D exceeds its range, M1092 will be On.
7. The Pointer (Pr) of the matrix is designated by the user in the instruction. The vaild range of Pr is 0 ~ 16n -1, corresponding to b_0 ~ b_{16n-1} in the matrix. If the Pr value exceeds its range, M1092 will be On and the instruction will not be executed.

Program Example:

1. When X0 goes from Off to On, M1094 will be set to “0” and M1093 to “1”. Therefore, the Pr will plus 1 after every writing.

2. Set the Pr value D20 = 45 and M1096 = 1. When X0 goes from Off to On for 1 time, we can obtain the execution results: D20 = 45, M1096 = 1, M1089 = 0.



Remarks:

Flag explanations:

Flags	Function
M1088	Matrix search end flag. When the comparison reaches the last bit, M1089 = 1.
M1092	Matrix pointer error flag. When the pointer Pr exceeds its range, M1092 = 1.
M1093	Matrix pointer increasing flag. Adding 1 to the current value of the Pr.
M1094	Matrix pointer clear flag. Clearing the current value of the Pr to 0.
M1096	Matrix displacement/input complement flag.

API	Mnemonic		Operands	Function	Controllers		
					ES/EX/SS	SA/SX/SC	EH/SV
188	MBS	P	(S) (D) (n)	Matrix Bit Displacement			

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			MBS, MBSP: 7 steps
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

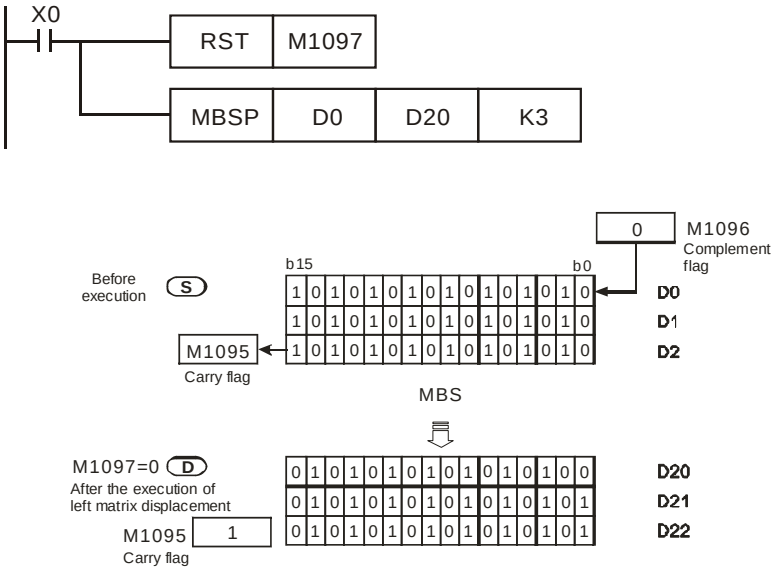
S: Matrix source device **D**: Operation result **n**: Array length

Explanations:

1. Range of **n**: K1 ~ K256
2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. Flags: M1095 ~ M1097. See remarks for more details.
6. This instruction performs left-right displacement on the matrix bits in **S** according to array length **n**. M1097 determines the left (M1097 = 0) or right (M1097 = 1) displacement of matrix bits. The empty bits derived from every displacement of 1 bit (when left displacement: b_0; when right displacement: b_{16n-1}) is filled by the status of the complement flag (M1096). The spare bits (when left displacement: b_{16n-1}; when right displacement: b_0) are sent to the carry flag (M1095). The result is stored in **D**.
7. The pulse execution instruction MBSP is generally adopted.

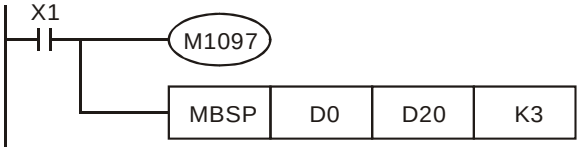
Program Example 1:

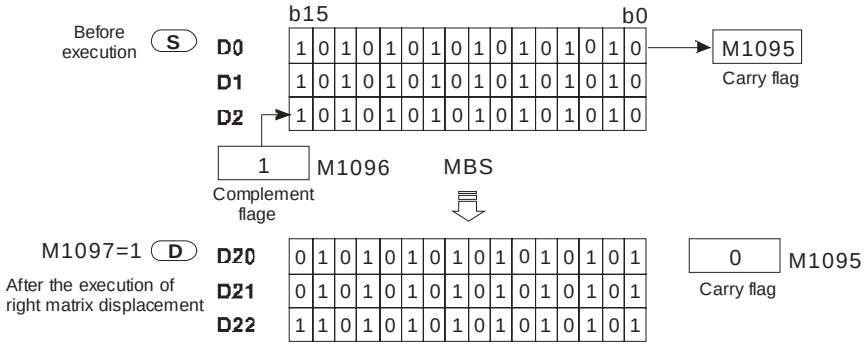
When X0 = On, M1097 = Off, indicating a left matrix displacement is performed. Set M1096 = 0 and the 16-bit registers D0 ~ D2 will perform a left matrix displacement and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The carry flag M1095 will be “1”.



Program Example 2:

When X1 = On, M1097 = On, indicating a right matrix displacement is performed. Set M1096 = 1 and the 16-bit registers D0 ~ D2 will perform a right matrix displacement and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The carry flag M1095 will be “0”.





Explanations:

Flag explanations:

Flags	Function
M1095	Matrix rotation/displacement/output carry flag.
M1096	Matrix displacement/input complement flag.
M1097	Matrix rotation/displacement direction flag.

API	Mnemonic		Operands	Function	Controllers		
189	MBR	P	(S) (D) (n)	Matrix Bit Rotation	ES/EX/SS	SA/SX/SC	EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			MBR, MBRP: 7 steps
D								*	*	*	*	*	*			
n					*	*							*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

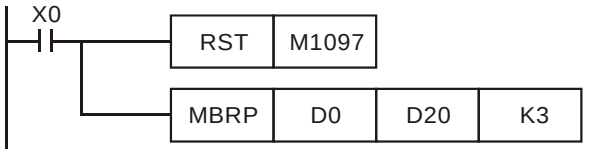
S: Matrix source device **D**: Operation result **n**: Array length

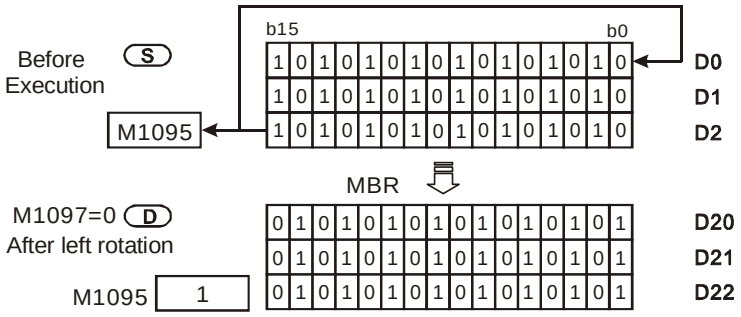
Explanations:

1. Range of **n**: K1 ~ K256
2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. Flags: M1095, M1097. See remarks for more details.
6. This instruction performs left-right rotation on the matrix bits in **S** according to array length **n**. M1097 determines the left (M1097 = 0) or right (M1097 = 1) rotation of matrix bits. The empty bits derived from every rotation of 1 bit (when left rotation: b_0; when right rotation: b_{16n-1}) is filled by rotation bits (when left rotation: b_{16n-1}; when right rotation: b_0). The result is stored in **D**. Rotation bits not only fill the empty bits but also send the status of bits to the carry flag M1095.
7. The pulse execution instruction MBRP is generally adopted.

Program Example 1:

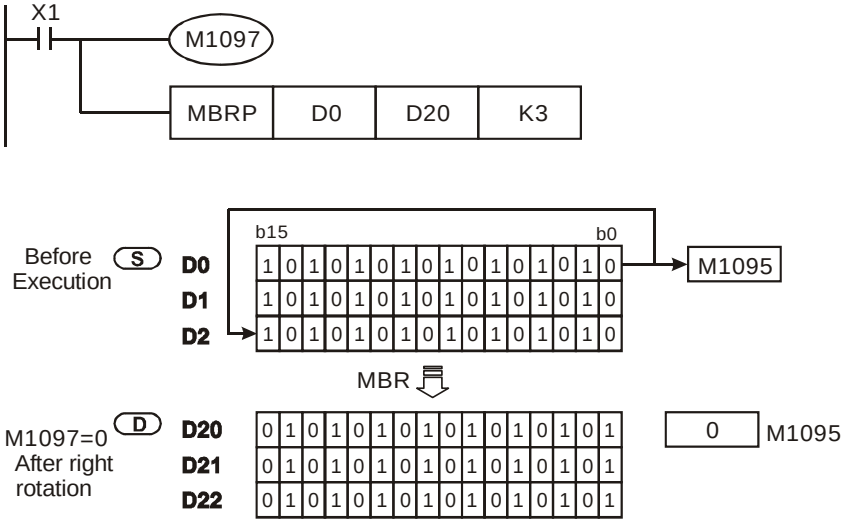
When X0 = On, M1097 = Off, indicating a left matrix rotation is performed. The 16-bit registers D0 ~ D2 will perform a left matrix rotation and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The carry flag M1095 will be “1”.





Program Example 2:

When X1 = On, M1097 = On, indicating a right matrix rotation is performed. The 16-bit registers D0 ~ D2 will perform a right matrix rotation and the result will be stored in the matrix of the 16-bit registers D20 ~ D22. The carry flag M1095 will be “0”.



Remarks:

Flag explanations:

Flags	Function
M1095	Matrix rotation/displacement/output carry flag.
M1097	Matrix rotation/displacement direction flag.

API	Mnemonic		Operands	Function	Controllers
190	MBC	P	(S) (n) (D)	Matrix Bit Status Counting	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S							*	*	*	*	*	*	*			MBC, MBCP: 7 steps
n					*	*							*			
D								*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

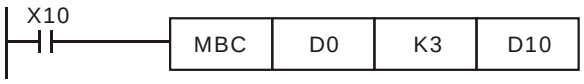
S: Matrix source device **n**: Array length **D**: Counting result

Explanations:

1. Range of **n**: K1 ~ K256
2. **S** designates KnX, KnY, KnM and KnS; **D** designates KnY, KnM and KnS.
3. SA/SX/SC can designate n = 4. EH/EH2/SV can designate n $\leqq$ 4.
4. See the specifications of each model for their range of use.
5. Flags: M1098, M1099. See remarks for more details.
6. This instruction counts the number of bits which are “1” or “0” in **S** by array length **n**. The result is stored in **D**.
7. The instruction counts the number of bits which are “1” when M1098 = 1 and counts the number of bits which are “0” when M1098 = 0. When the operation result is “0”, M1099 = 1.

Program Example:

When X10 = On, in the matrix of D0 ~ D2, when M1098 = 1, the instruction counts the total number of bits which are “1” and store the number in D10. When M1098 = 0, the instruction counts the total number of bits which are “0” and store the number in D10.



D0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1
D1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1
D2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1

D10 12 M1098=0

D10 36 M1098=1

Remarks:

Flag explanations:

Flags	Function
M1098	Counting the number of bits which are “1” or “0”
M1099	On when the counting result is “0”.

API	Mnemonic	Operands	Function	Controllers
191	D PPMR	(S1) (S2) (S) (D)	2-Axis Relative Point to Point Motion	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DPPMR: 17 steps
S_2					*	*							*			
S					*	*							*			
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Number of output pulses of X axis **S_2**: Number of output pulses of Y axis **S**: Max. point to point output frequency **D**: Pulse output device

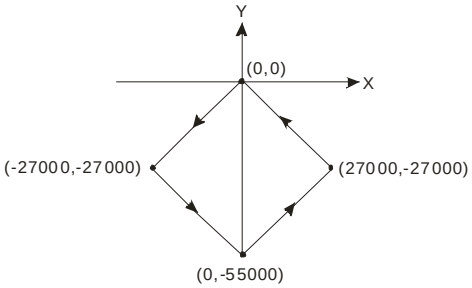
Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks for more details.
2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports “pulse + direction” mode.
3. **S_1** and **S_2** are the designated (relative designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.
4. **D** can designate Y0 and Y4.
   When Y0 is designated:
   Y0 refers to 1st group X-axis pulse output device.
   Y1 refers to 1st group X-axis direction signal.
   Y2 refers to 1st group Y-axis pulse output device.
   Y3 refers to 1st group Y-axis direction signal.
   Y4 refers to 2nd group X-axis pulse output device.
   Y5 refers to 2nd group X-axis direction signal.
   Y6 refers to 2nd group Y-axis pulse output device.
   Y7 refers to 2nd group Y-axis direction signal.
   When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.
5. D1340 (D1379) refers to the settings of the start/end frequencies of the 1st/2nd 2-axis motion. D1343 (D1381) refers to the time of the first acceleration segment and last deceleration segment of the 1st/2nd 2-axis motion. The time shall be longer than 10ms. If the time is shorter than 10ms or longer than 10,000ms, the output will be operated at 10ms. Default setting = 100ms.
6. If the maximum output frequency setting is less than 10Hz, the output will be operated at 10Hz. If the setting is more than 200KHz, the output will be operated at 200KHz.

7. When the 2-axis synchronous motion instruction is enabled, the start frequency and acceleration/deceleration time in Y axis will be same as the settings in X axis.
8. The number of output pulses for the 2-axis motion shall not be less than 59; otherwise the line drawn will not be straight enough.
9. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.

Program Example:

1. Draw a rhombus as the figure below.

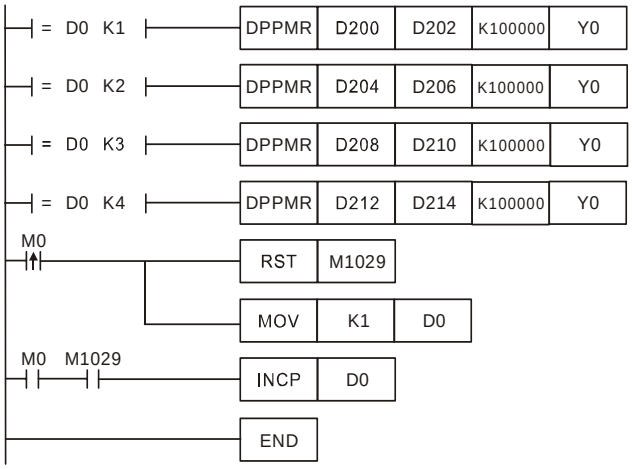


2. Steps:

a) Set the four coordinates (0,0), (-27000, -27000), (0, -55000), (27000, -27000) (as the figure above). Calculate the relative coordinates of the four points and obtain (-27000, -27000), (27000, -28000), (27000, 27000), and (-27000, 27000). Place them in the 32-bit (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Write program codes as follows.

c) PLC RUN. Set M0 as On and start the 2-axis line drawing.



3. Motion explanation:

When PLC RUN and M0 = On, PLC will start the first point-to-point motion by 100KHz. D0 will plus 1 whenever a point-to-point motion is completed and the second point-to-point motion will start to execute automatically. The same motion will keep executing until the fourth point-to-point motion is completed.

Remarks:

1. Flag explanations:
   - M1029： On when the 1st group 2-axis pulse output is completed.
   - M1036： On when the 2nd group 2-axis pulse output is completed.
   - M1334： On when the 1st group 2-axis pulse output is forbidden.
   - M1336： 1st group 2-axis pulse output indication flag
   - M1520： On when the 2nd group 2-axis pulse output is forbidden.
   - M1522： 2nd group 2-axis pulse output indication flag
2. Special register explanations:
   - D1336, D1337 **:** Pulse present value register for Y0 output of the 1st group X-axis motion. The present value increases or decreases following the rotation direction. (D1337 high word; D1336 low word)
   - D1338, D1339 **:** Pulse present value register for Y2 output of the 1st group Y-axis motion. The present value increases or decreases following the rotation direction. (D1339 high word; D1338 low word)
   - D1340 **:** Frequency settings of the first acceleration and last deceleration segment for the Y0 output of the 1st group X-axis motion and Y2 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.
   - D1343 **:** Time settings of the first acceleration and last deceleration segment for the Y0 output of the 1st group X-axis motion and Y2 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.
   - D1375, D1376 **:** Pulse present value register for Y4 output of the 2nd group X-axis motion. The present value increases or decreases following the rotation direction. (D1337 high word; D1336 low word)
   - D1377, D1378 **:** Pulse present value register for Y6 output of the 2nd group Y-axis motion. The present value increases or decreases following the rotation direction. (D1339 high word; D1338 low word)
   - D1379 **:** Frequency settings of the first acceleration and last deceleration segment for the Y4 output of the 2nd group X-axis motion and Y6 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.
   - D1381 **:** Time settings of the first acceleration and last deceleration segment for the Y4 output of the 2nd group X-axis motion and Y6 of the Y-axis motion for API 191 DPPMR and API 192 DPPMA.

API	Mnemonic	Operands	Function	Controllers
192	D PPMA	(S1) (S2) (S) (D)	2-Axis Absolute Point to Point Motion	ES/EX/SS | SA/SX/SC | EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DPPMA: 17 steps
S_2					*	*							*			
S					*	*							*			
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Number of output pulses of X axis **S_2**: Number of output pulses of Y axis **S**: Max. point to point output frequency **D**: Pulse output device

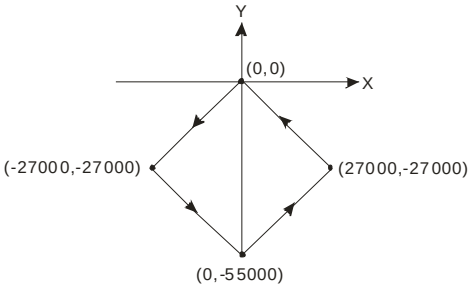
Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks of API 191 DPPMR for more details.
2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports “pulse + direction” mode.
3. **S_1** and **S_2** are the designated (absolute designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.
4. **D** can designate Y0 and Y4.
   When Y0 is designated:
   Y0 refers to 1st group X-axis pulse output device.
   Y1 refers to 1st group X-axis direction signal.
   Y2 refers to 1st group Y-axis pulse output device.
   Y3 refers to 1st group Y-axis direction signal.
   Y4 refers to 2nd group X-axis pulse output device.
   Y5 refers to 2nd group X-axis direction signal.
   Y6 refers to 2nd group Y-axis pulse output device.
   Y7 refers to 2nd group Y-axis direction signal.
   When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.
5. D1340 (D1379) refers to the settings of the start/end frequencies of the 1st/2nd 2-axis motion. D1343 (D1381) refers to the time of the first acceleration segment and last deceleration segment of the 1st/2nd 2-axis motion. The time shall be longer than 10ms. If the time is shorter than 10ms or longer than 10,000ms, the output will be operated at 10ms. Default setting = 100ms.
6. If the maximum output frequency setting is less than 10Hz, the output will be operated at 10Hz. If the setting is more than 200KHz, the output will be operated at 200KHz.

7. When the 2-axis synchronous motion instruction is enabled, the start frequency and acceleration/deceleration time in Y axis will be same as the settings in X axis.
8. The number of output pulses for the 2-axis motion shall not be the values within 1 ~ 59; otherwise the line drawn will not be straight enough.
9. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.

Program Example:

1. Draw a rhombus as the figure below.

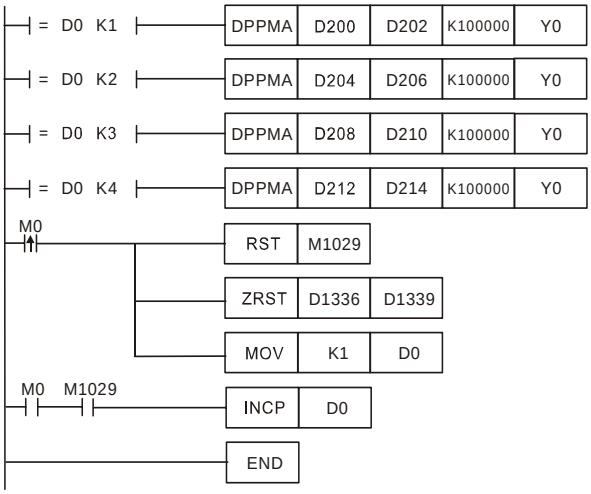


2. Steps:

a) Set the four coordinates (0,0), (-27000, -27000), (0, -55000), (27000, -27000) (as the figure above). Place them in the 32-bit (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Write program codes as follows.

c) PLC RUN. Set M0 as On and start the 2-axis line drawing.



3. Motion explanation:

When PLC RUN and M0 = On, PLC will start the first point-to-point motion by 100KHz. D0 will plus 1 whenever a point-to-point motion is completed and the second point-to-point motion will start to execute automatically. The same motion will keep executing until the fourth point-to-point motion is completed.

API	Mnemonic	Operands	Function	Controllers
193	D CIMR	S_1 S_2 S D	2-Axis Relative Position Arc Interpolation	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DCIMR: 17 steps
S_1					*	*							*			
S_2					*	*							*			
S													*			
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Number of output pulses of X axis **S_2**: Number of output pulses of Y axis **S**: Parameter setting **D**: Pulse output device

Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks of API 191 DPPMR for more details.
2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports “pulse + direction” mode.
3. **S_1** and **S_2** are the designated (relative designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.
4. The lower 16 bits of **S** (settings of direction and resolution): K0 refers to clockwise 10-segment (average resolution) output; K2 refers to clockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 1 and 2). K1 refers to counterclockwise 10-segment (average resolution) output; K3 refers to counterclockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 3 and 4).
5. The higher 16 bits of **S** (settings of motion time): K1 refers to 0.1 second. The setting range for average resolution is K1 ~ K100 (0.1 sec. ~ 10 secs.), for higher resolution is K2 ~ K200 (0.2 sec. ~ 20 secs.) This instruction is restricted by the maximum pulse output frequency; therefore when the set time goes faster than the actual output time, the set time will be automatically modified.

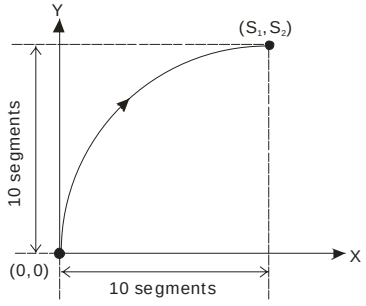


Figure 1

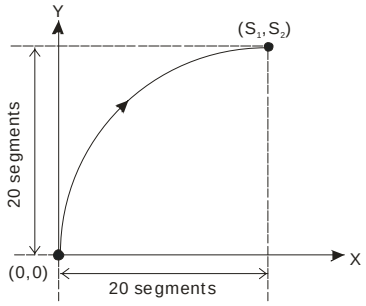


Figure 2

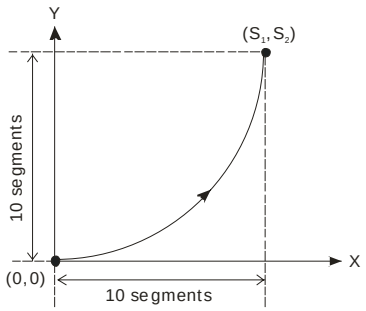


Figure 3

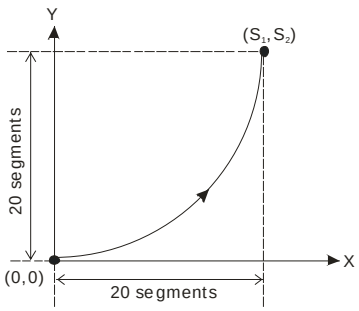


Figure 4

6. **D** can designate Y0 and Y4.
   When Y0 is designated:
   Y0 refers to 1st group X-axis pulse output device.
   Y1 refers to 1st group X-axis direction signal.
   Y2 refers to 1st group Y-axis pulse output device.
   Y3 refers to 1st group Y-axis direction signal.
   When Y4 is designated:
   Y4 refers to 2nd group X-axis pulse output device.
   Y5 refers to 2nd group X-axis direction signal.
   Y6 refers to 2nd group Y-axis pulse output device.
   Y7 refers to 2nd group Y-axis direction signal.
   When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.
7. Draw four 90° arcs.
8. When the direction signal is On, the direction is positive. When the direction signal is Off, the direction is negative. When **S** is set as K0, K2, the arcs will be clockwise (see figure 5). When **S** is set as K1, K3, the arcs will be counterclockwise (see figure 6).

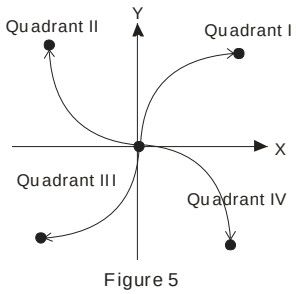


Figure 5

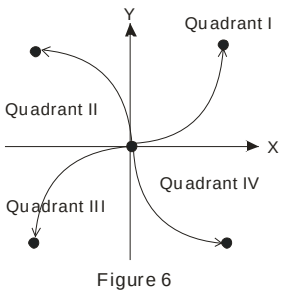


Figure 6

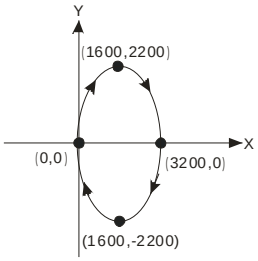
9. When the 2-axis motion is being executed in 10 segments (of average resolution), the operation time of the instruction when the instruction is first enabled is approximately 5ms. The number of output pulses cannot be less than 100 and more than 1,000,000; otherwise, the instruction cannot be enabled.
10. When the 2-axis motion is being executed in 20 segments (of high resolution), the operation time of the instruction when the instruction is first enabled is approximately 10ms. The number of output pulses cannot be less than 1,000 and more than 10,000,000; otherwise, the instruction cannot be enabled.

11. If you wish the number of pulses in 10-segment or 20-segment motion to be off the range, you may adjust the gear ratio of the servo for obtaining your desired number.
12. Every time when the instruction is executed, only one 90° arc can be drawn. It is not necessary that the arc has to be a precise arc, i.e. the numbers of output pulses in X and Y axes can be different.
13. There are no settings of start frequency and acceleration/deceleration time.
14. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.
15. The settings of direction and resolution in the lower 16 bits of **S** can only be K0 ~ K3.
16. The settings of motion time in the high 16 bits of **S** can be slower than the the fastest suggested time but shall not be faster than the fastest suggested time.
17. The fastest suggested time for the arc interpolation:

Segments	Max. target position (pulse)	Fastest suggested set time (unit:100ms)
Average resolution	100 ~ 10,000	1
	10,001 ~ 19,999	2
	:	:
	Less than 1,000,000	Less than 100
Higher resolution	1,000 ~ 20,000	2
	20,000 ~ 29,999	3
	:	:
	Less than 10,000,000	Less than 200

Program Example 1:

1. Draw an ellipse as the figure below.



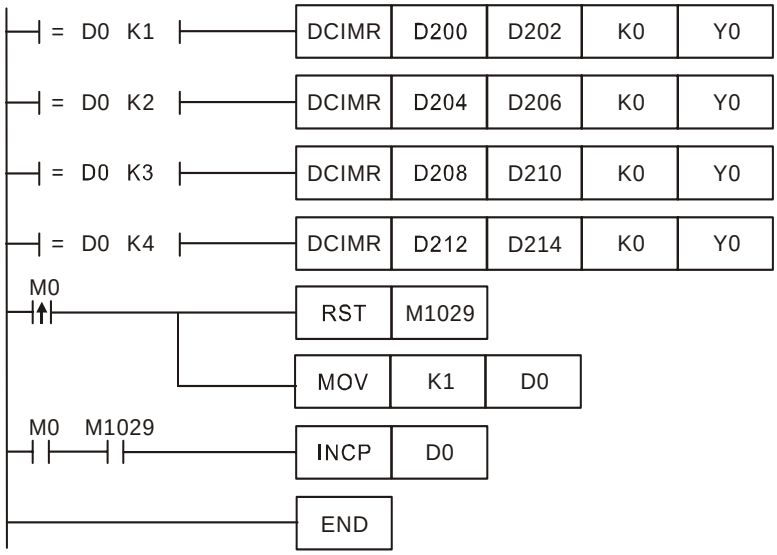
2. Steps:

a) Set the four coordinates (0,0), (1600, 2200), (3200, 0), (1600, -2200) (as the figure above). Calculate the relative coordinates of the four points and obtain (1600, 2200), (1600, -2200), (-1600, -2200), and (-1600, 2200). Place them in the 32-bit (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Select “draw clockwise arc” and “average resolution” (**S** = K0).

c) Write program codes as follows.

d) PLC RUN. Set M0 as On and start the drawing of the ellipse.

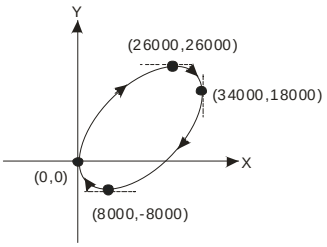


3. Motion explanation:

   When PLC RUN and M0 = On, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The same motion will keep executing until the fourth segment of arc is completed.

Program Example 2:

1. Draw a tilted ellipse as the figure below.



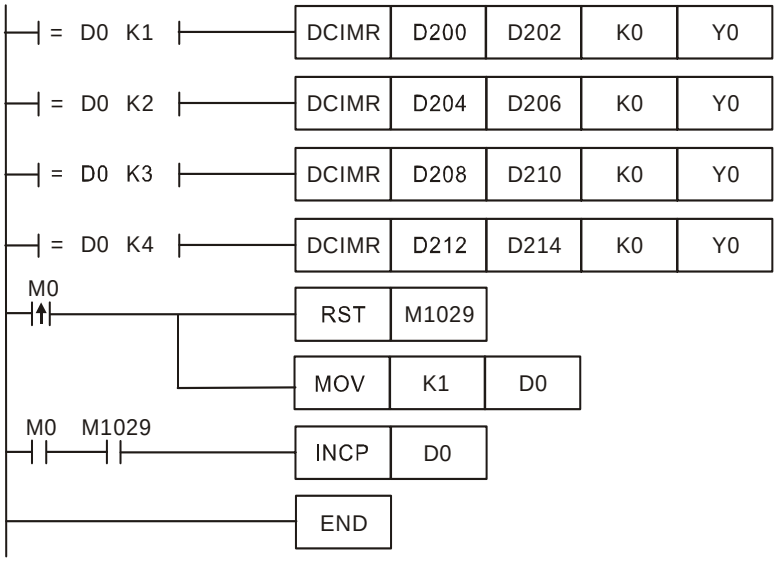
2. Steps:

a) Find the max. and min. coordinates on X and Y axes (0,0), (26000,26000), (34000,18000), (8000,-8000) (as the figure above). Calculate the relative coordinates of the four points and obtain (26000,26000)、(8000,-8000)、(-26000,-26000), (-8000,8000). Place them respectively in the 32-bit (D200,D202), (D204,D206), (D208,D210) and (D212,D214).

b) Select “draw clockwise arc” and “average resolution” (**S** = K0).

c) Select DCIMR instruction for drawing arc and write program codes as follows.

d) PLC RUN. Set M0 as On and start the drawing of the ellipse.



3. Motion explanation:

   When PLC RUN and M0 = On, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The same motion will keep executing until the fourth segment of arc is completed.

API	Mnemonic	Operands	Function	Controllers
194	D CIMA	S_1 S_2 S D	2-Axis Absolute Position Arc Interpolation	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DCIMA: 17 steps
S_2					*	*							*			
S													*			
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Number of output pulses of X axis **S_2**: Number of output pulses of Y axis **S**: Parameter setting **D**: Pulse output device

Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks of API 191 DPPMR for more details.
2. This instruction only supports EH2/SV series MPU, not EH series. In terms of pulse output methods, this instructin only supports “pulse + direction” mode.
3. **S_1** and **S_2** are the designated (absolute designation) number of output pulses in X axis (Y0 or Y4) and Y axis (Y2 or Y6). The range of the number is -2,147,483,648 ~ +2,147,483,647. When **S_1** and **S_2** are larger than pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word), and CH3 (D1378 high word, D1377 low word), the output direction will be positive and direction signals Y1, Y3, Y5, Y7 will be On. When **S_1** and **S_2** are less than pulse present value registers, the output direction will be negative and direction signals Y1, Y3, Y5, Y7 will be Off.
4. The lower 16 bits of **S** (settings of direction and resolution): K0 refers to clockwise 10-segment (average resolution) output; K2 refers to clockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 1 and 2). K1 refers to counterclockwise 10-segment (average resolution) output; K3 refers to counterclockwise 20-segment (higher resolution) output and a 90° arc can be drawn (see figure 3 and 4).
5. The higher 16 bits of **S** (settings of motion time): K0 refers to 0.1 second. The setting range for average resolution is K1 ~ K100 (0.1 sec. ~ 10 secs.), for higher resolution is K2 ~ K200 (0.2 sec. ~ 20 secs.) This instruction is restricted by the maximum pulse output frequency; therefore when the set time goes faster than the actual output time, the set time will be automatically modified.

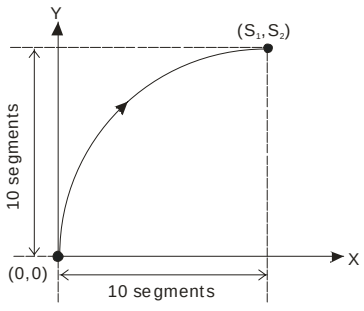


Figure 1

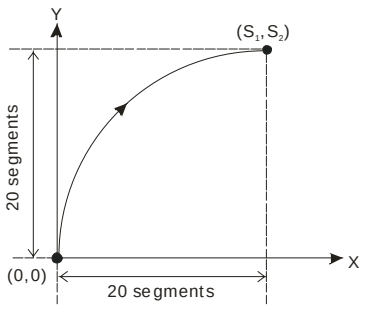


Figure 2

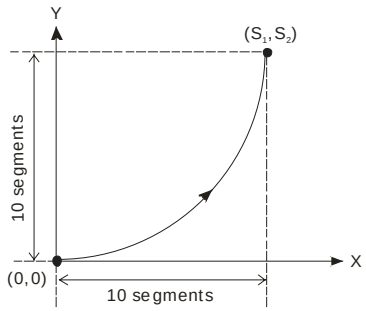


Figure 3

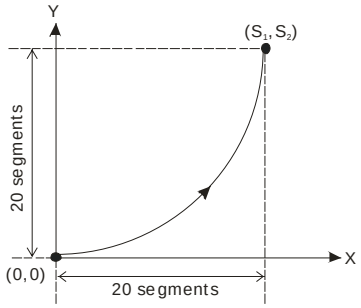


Figure 4

6. **D** can designate Y0 and Y4.
   When Y0 is designated:
   Y0 refers to 1st group X-axis pulse output device.
   Y1 refers to 1st group X-axis direction signal.
   Y2 refers to 1st group Y-axis pulse output device.
   Y3 refers to 1st group Y-axis direction signal.
   When Y4 is designated:
   Y4 refers to 2nd group X-axis pulse output device.
   Y5 refers to 2nd group X-axis direction signal.
   Y6 refers to 2nd group Y-axis pulse output device.
   Y7 refers to 2nd group Y-axis direction signal.
   When direction signal outputs, Off will not occur immediately after the pulse output is over. Direction signal will turn Off when the drive contact is Off.
7. Draw four 90° arcs.
8. When the direction signal is On, the direction is positive. When the direction signal is Off, the direction is negative. When **S** is set as K0, K2, the arcs will be clockwise (see figure 5). When **S** is set as K1, K3, the arcs will be counterclockwise (see figure 6).

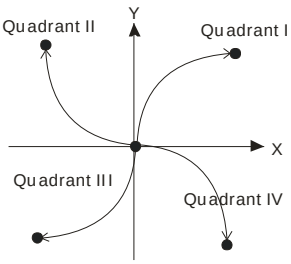


Figure 5

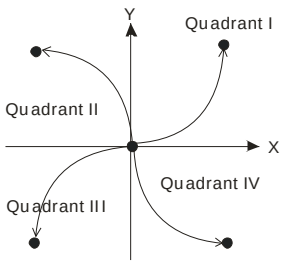


Figure 6

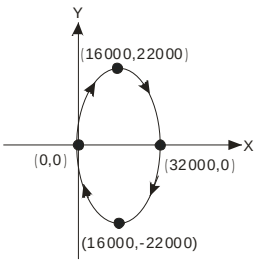
9. When the 2-axis motion is being executed in 10 segments (of average resolution), the operation time of the instruction when the instruction is first enabled is approximately 5ms. The number of output pulses cannot be less than 100 and more than 1,000,000; otherwise, the instruction cannot be enabled.
10. When the 2-axis motion is being executed in 20 segments (of high resolution), the operation time of the instruction when the instruction is first enabled is approximately 10ms. The number of output pulses cannot be less than 1,000 and more than 10,000,000; otherwise, the instruction cannot be enabled.

11. If you wish the number of pulses in 10-segment or 20-segment motion to be off the range, you may adjust the gear ratio of the servo for obtaining your desired number.
12. Every time when the instruction is executed, only one 90° arc can be drawn. It is not necessary that the arc has to be a precise arc, i.e. the numbers of output pulses in X and Y axes can be different.
13. There are no settings of start frequency and acceleration/deceleration time.
14. There is no limitation on the number of times using the instruction. However, assume CH1 or CH2 output is in use, the 1st group X/Y axis will not be able to output. If CH3 or CH4 output is in use, the 2nd group X/Y axis will not be able to output.
15. The settings of direction and resolution in the lower 16 bits of **S** can only be K0 ~ K3.
16. The settings of motion time in the high 16 bits of **S** can be slower than the the fastest suggested time but shall not be faster than the fastest suggested time.
17. The fastest suggested time for the arc interpolation:

Segments	Max. target position (pulse)	Fastest suggested set time (unit:100ms)
Average resolution	100 ~ 10,000	1
	10,001 ~ 19,999	2
	:	:
	Less than 1,000,000	Less than 100
Higher resolution	1,000 ~ 20,000	2
	20,000 ~ 29,999	3
	:	:
	Less than 10,000,000	Less than 200

Program Example 1:

1. Draw an ellipse as the figure below.



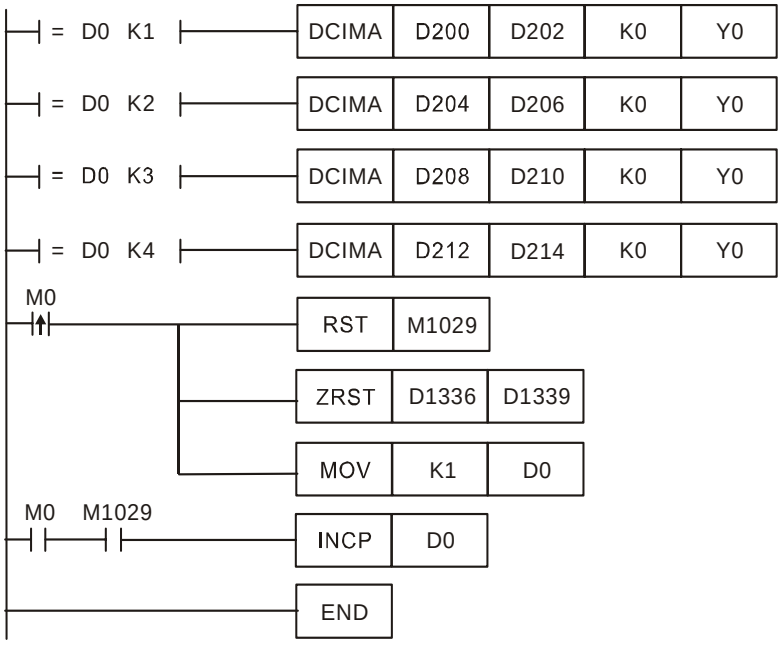
2. Steps:

a) Set the four coordinates (0,0), (16000, 22000), (32000, 0), (16000, -22000) (as the figure above). Place them in the 32-bit (D200, D202), (D204, D206), (D208, D210), (D212, D214).

b) Select “draw clockwise arc” and “average resolution” (**S** = K0).

c) Select DCIMA instruction for drawing arc and write program codes as follows.

d) PLC RUN. Set M0 as On and start the drawing of the ellipse.

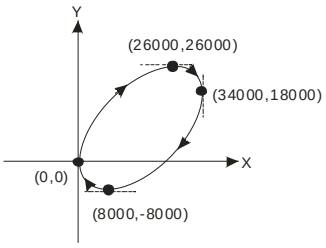


3. Motion explanation: ：

   When PLC RUN and M0 = On, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The same motion will keep executing until the fourth segment of arc is completed.

Program Example 2:

1. Draw a tilted ellipse as the figure below.



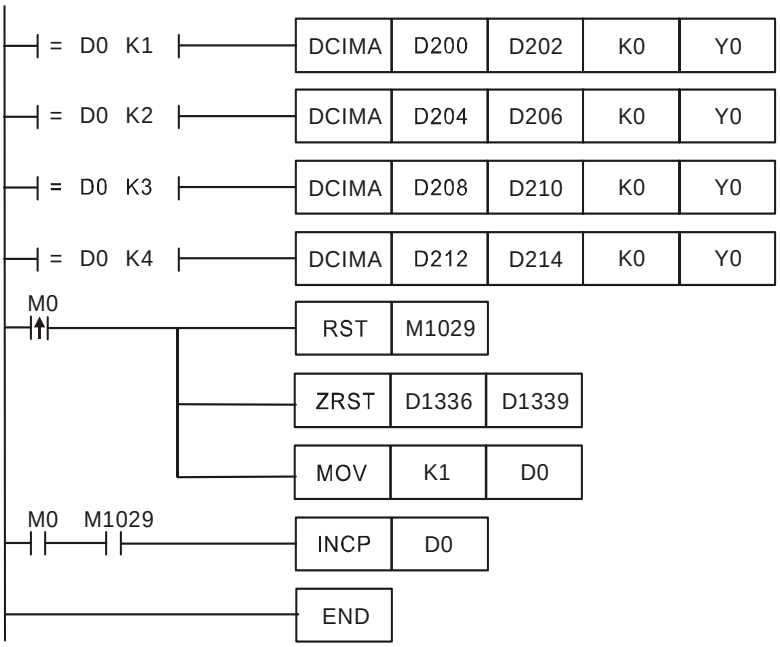
2. Steps:

a) Find the max. and min. coordinates on X and Y axes (0,0), (26000,26000), (34000,18000), (8000,-8000) (as the figure above). Place them respectively in the 32-bit (D200,D202), (D204,D206), (D208,D210) and (D212,D214).

b) Select “draw clockwise arc” and “average resolution” (**S** = K0).

c) Select DCIMA instruction for drawing arc and write program codes as follows.

d) PLC RUN. Set M0 as On and start the drawing of the ellipse.



3. Motion explanation:

   When PLC RUN and M0 = On, PLC will start the drawing of the first segment of the arc. D0 will plus 1 whenever a segment of arc is completed and the second segment of the arc will start to execute automatically. The same motion will keep executing until the fourth segment of arc is completed.

API	Mnemonic	Operands	Function	Controllers
195	D PTPO	S_1 S_2 D	Single-Axis Pulse Output by Table	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1													*			DPTPO: 13 steps
S_2													*			
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Source start device **S_2**: Number of segments **D**: Pulse output device

Explanations:

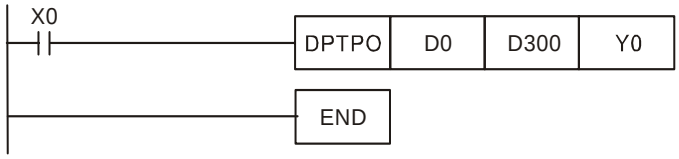
1. Flags: M1029, M1030, M1334, M1335. See remarks for more details.
2. This instruction only supports EH2/SV series MPU, not EH series.
3. According to the value of **S_2** + 0, every segment consecutively occupy four register D. (**S_1** + 0) refers to output frequency. (**S_1** + 2) refers to the number of output pulses.
4. When the output frequency of **S_1** is less than 1, PLC will automatically modify it as 1. When the value is larger than 200,000KHz, PLC will automatically modify it as 200,000KHz.
5. **S_2** + 0: number of segments (range: 1 ~ 60). **S_2** + 1: number of segments being executed. Whenever the program scans to this instruction, the instruction will automatically update the segment No. that is currently being executed.
6. **D** can only designate output devices Y0, Y2, Y4 and Y6 and can only perform pulse output control. For the pin for direction control, the user has to compile other programs to control.
7. This instruction does not offer acceleration and deceleration functions. Therefore, when the instruction is disabled, the output pulses will stop immediately.
8. In every program scan, each channel can only be executed by one instruction. However, there is no limitation on the number of times using this instruction.
9. When the instruction is being executed, the user is not allowed to update the frequency or number of the segments. Changes made will not be able to make changes in the actual output.

Program Example:

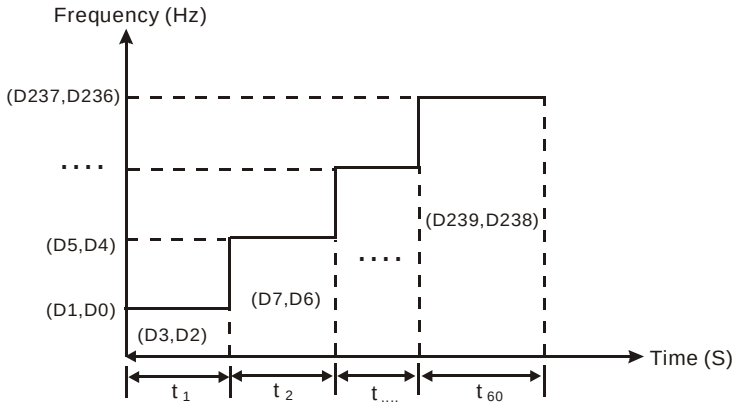
1. When X0 = On, the output will be operated according to the set frequency and number of pulses in every segment.
2. Format of the table:

S_2 = D300, number of segments (D300 = K60)	**S_1** = D0, frequency (**S_1** + 0)	**S_1** = D0, number of output pulses (**S_1** + 2)
K1 (1st segment)	D1, D0	D3, D2
K2 (2nd segment)	D5, D4	D7, D6
: :	: :	: :
K60 (60th segment)	D237, D236	D239, D238

3. Monitor the segment No. that is currently being executed in register D301.



4. The pulse output curve:



Remarks:

1. Flag explanations:

   M1029： On when CH0 (Y0) pulse output is completed.

   M1030： On when CH1 (Y2) pulse output is completed.

   M1036： On when CH2 (Y4) pulse output is completed.

   M1037： On when CH3 (Y6) pulse output is completed.

   M1334： When On, CH0 (Y0) pulse output will be forbidden.

   M1335： When On, CH1 (Y2) pulse output will be forbidden.

   M1520： When On, CH2 (Y4) pulse output will be forbidden.

   M1521： When On, CH3 (Y6) pulse output will be forbidden.

   M1336： CH0 (Y0) pulse output indication flag

   M1337： CH1 (Y2) pulse output indication flag

   M1522： CH2 (Y4) pulse output indication flag

   M1523： CH3 (Y6) pulse output indication flag

2. Special register explanations:

   D1336, D1337： Pulse present value register of CH0 (Y0) (D1337 high word, D1336 low word)

   D1338, D1339： Pulse present value register of CH1 (Y2) (D1339 high word, D1338 low word)

   D1375, D1376： Pulse present value register of CH2 (Y4) (D1376 high word, D1375 low word)

   D1377, D1378： Pulse present value register of CH3 (Y6) (D1378 high word, D1377 low word)

API	Mnemonic		Operands	Function	Controllers
196	HST	P	S	High Speed Timer	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S					*	*										HST, HSTP: 3 steps

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

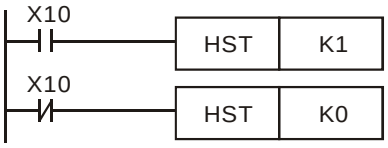
S: Condition to stop the startup of high speed timer

Explanations:

1. Range of **S**: S = K0 (H0), K1 (H1).
2. Flag: M1015
3. When **S** = 1, the high speed timer will be enabled and M1015 = On. The high speed timer starts to time and record the present value in D1015 (min. unit: 100us).
4. Timing range of D1015: K0 ~ K32,767. When the timing reaches K32,767, the next timing will restart from 0.
5. When **S** = 0, the high speed timer will be disabled and M1015 = Off. D1015 will stop the timing immediately.
6. When **S** is neither 1 nor 0, HST instruction will not be executed.

Program Example :

1. When X10 = On, M1015 will be On. The high speed timer will start to time and record the present value in D1015.
2. When X10 = Off, M1015 will be Off. The high speed timer will be shut down.



Remarks:

1. Flag explanations:
   M1015: high speed timer start-up flag
   D1015: high speed timer
2. EH/EH2/SV series MPU do not use this instruction and use special M and special D directly for the timer.
   a) Special M and special D are only applicable when PLC RUN.
   b) When M1015 = On and PLC scans to END instruction, the high speed timer D1015 will be enabled. The minimum timing unit of D1015: 100us.
   c) Timing range of D1015: K0 ~ K32,767. When the timing reaches K32,767, the next timing will restart from K0.
   d) When M1015 = Off, D1015 will stop the timing when encountering END or HST instruction.
3. SA/SX/SC series MPU do not use this instruction and use special M and special D directly for the timer.
   a) Special M and special D are applicable when PLC RUN or STOP.

b) When M1015 = On, the high speed timer D1015 will be enabled. The minimum timing unit of D1015: 100us.

c) Timing range of D1015: K0 ~ K32,767. When the timing reaches K32,767, the next timing will restart from K0.

d) When M1015 = Off, D1015 will stop the timing immediately.

API	Mnemonic	Operands	Function	Controllers
197	D CLLM	S_1 S_2 S_3 D	Close Loop Position Control	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DCLLM: 17 steps
S_1	*											*				
S_2					*	*							*			
S_3					*	*							*			
D		*														

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Feedback source device **S_2**: Target number of feedbacks **S_3**: Target frequency of output

D: Pulse output device

Explanations:

1. Flags: M1029, M1030, M1334, M1335. See remarks for more details.
2. This instruction only supports EH2/SV series MPU, not EH series.
3. The corresponding interruption of **S_1**:

Source device	X0	X1	X2	X3	C241 ~ C254			
Corresponding outout	Y0	Y2	Y4	Y6	Y0	Y2	Y4	Y6
Interruption No.	I00□	I10□	I20□	I30□	I010	I020	I030	I040

□ = 1: rising-edige trigger; □ = 0: falling-edge trigger

   a) When **S_1** designates X as the input points and the pulse output reaches the set target number of feedbacks in **S_2**, the output will continue to operate by the frequency of the last segment until the interruption of X input points occurs.

   b) When **S_1** designates a high speed counter and the pulse output reaches the set target number of feedbacks in **S_2**, the output will continue to operate by the frequency of the last segment until the feedback pulses reaches the target number.

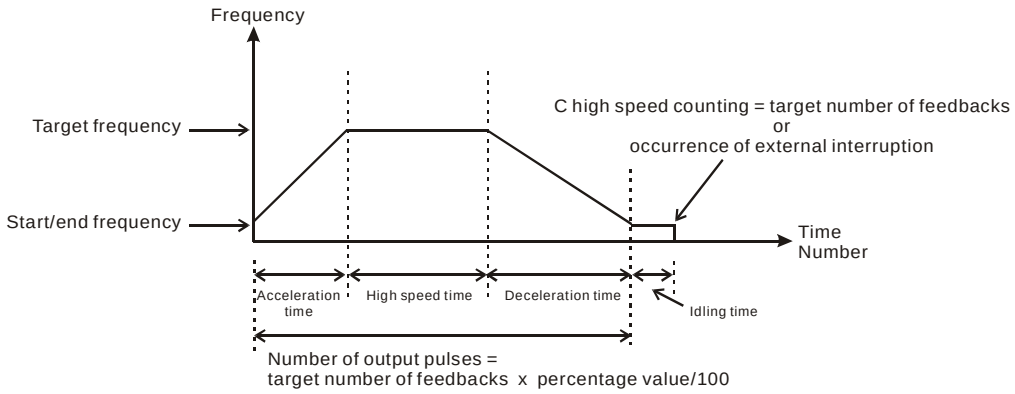
   c) **S_1** can be a high speed counter C or an external interruption X. If **S_1** is C, DCNT instruction should be first executed to enable the high-speed counting function and EI and I0x0 interruption service program to enable the high-speed interruption. If **S_1** is X, EI instruction and I0x0 interruption service program should be executed to enable the external interruption function.

4. The range of **S_2**: -2,147,483,648 ~ +2,147,483,647 (+/- represents the forward/backward direction). When in forward direction, the pulse present value registers CH0 (D1337 high word, D1336 low word), CH1 (D1339 high word, D1338 low word), CH2 (D1376 high word, D1375 low word) and CH3 (D1378 high word, D1377 low word) will increase. When in backward direction, the present value will decrease.
5. If **S_3** is lower than 10Hz, the output will operate at 10Hz; if **S_3** is higher than 200KHz, the output will operate at 200KHz.
6. **D** can only designate Y0, Y2, Y4 and Y6 and the direction signals repectively are Y1, Y3, Y5 and Y7. When there is a direction signal output, the direction signal will not be Off immediately after the pulse output is completed. The direction signal will be Off only when the drive contact is Off.

7. D1340, D1352, D1379 and D1380 are the settings of start/end frequencies of CH0 ~ CH3. The minimun frequency is 10Hz and default is 200Hz.
8. D1343, D1353, D1381 and D1382 are the settings of the time of the first segment and the last deceleration segment of CH0 ~ CH3. The acceleration/deceleration time cannot be shorter than 10ms. The outptu will be operated in 10ms if the time set is shorter than 10ms or longer than 10,000ms. The dafault setting is 100ms.
9. D1198, D1199, D1478 and D1479 are the output/input ratio of the close loop control in CH0 ~ CH3. K1 refers to 1 output pulse out of the 100 target feedback input pulses; K200 refers to 200 output pulses out of the 100 target feedback input pulses. D1198, D1199, D1478 and D1479 are the numerators of the ratio (range: K1 ~ K10,000) and the denominator is fixed as K100 (the user does not have to enter a denominator).
10. M1305, M1306, M1532 and M1533 are the direction signal flags for CH0 ~ CH3. When $\mathbf{S_2}$ is a positive value, the output will be in forward direction and the flag will be Off. When $\mathbf{S_2}$ is a negative value, the output will be in backward direction and the flag will be On.

Close Loop Explanations:

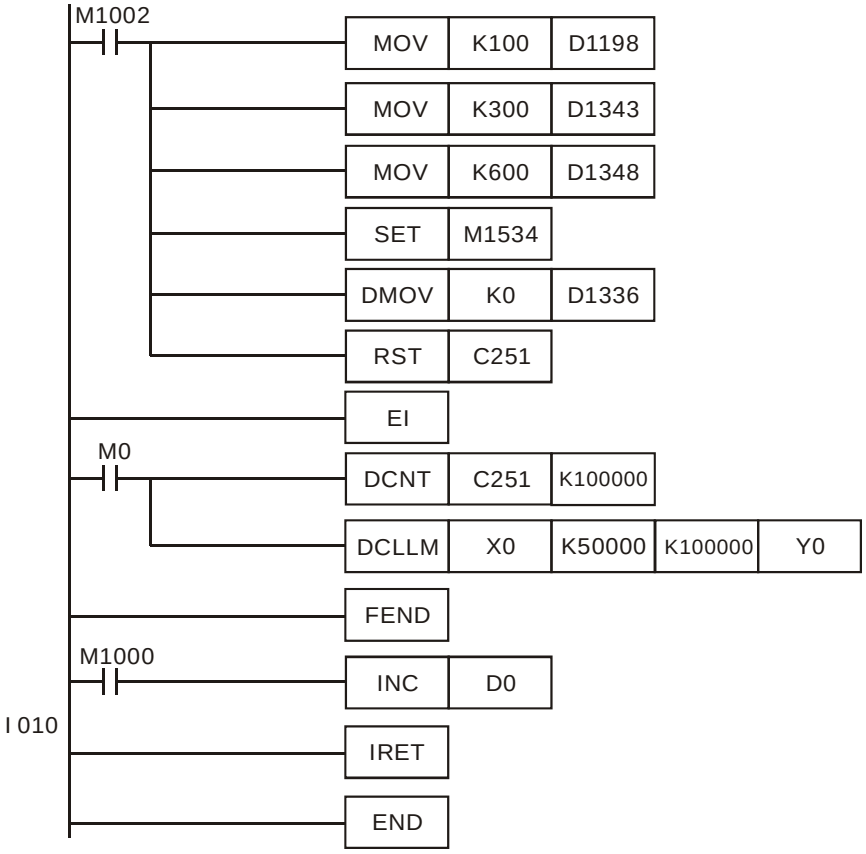
1. Function: Immediately stop the high-speed pulse output according to the number of feedback pulses or external interruption signals.
2. The execution:



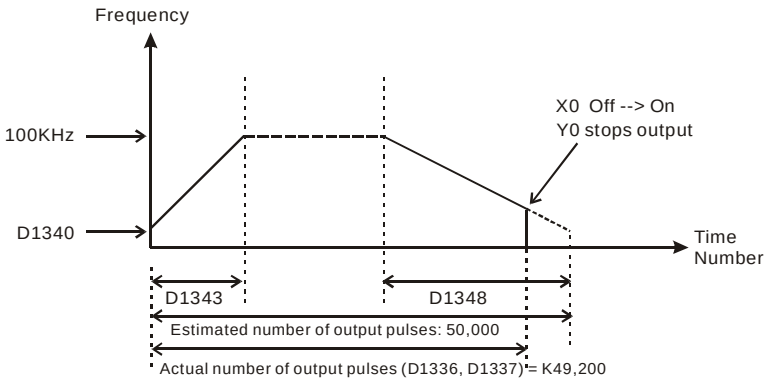
3. How to adjust the time for the completion of the positioning:
   a) The time for the completion of the positioning refers to the time for “acceleration + high speed + deceleration + idling” (see the figure above). For example, you can increase or decrease the entire number of output pulses by making adjustment on the percentage value and further increase or decrease the time required for the positioning.
   b) Among the four segments of time, only the idling time cannot be adjusted directly by the user. However, you can determine if the execution result is good or bad by the length of the idling time. In theory, a bit of idling left is the best result for a positioning.
   c) Owing to the close loop operation, the length of idling time will not be the same in every execution. Therefore, when the content in the special D for displaying the actial number of output pulses is smaller or larger than the calculated number of output pulses (taget number of feedbacks x percentage value/100), you can improve the situation by adjusting the percentage value, acceleration/decelartion time or target frequency.

Program Example:

1. Assume we adopt X0 as the external interruption, together with I001 (rising-edge trigger) interruption program; target number of feedbacks = 50,000; target frequency = 10KHz; Y0, Y1 (CH0) as output pulses; start/end frequency (D1340) = 200Hz; acceleration time (D1343) = 300ms; deceleration time (D1348) = 600ms; percentage value (D1198) = 100; current number of output pulses (D1336, D1337) = 0.
2. Write the program codes as follows:

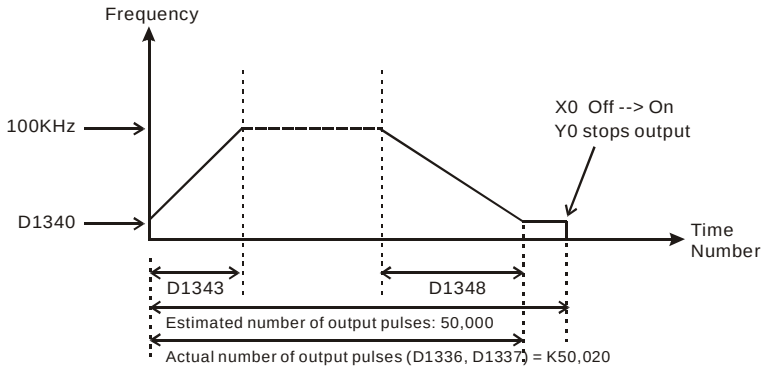


3. Assume the first execution result as:



4. Observe the result of the first execution:
   a) The actual output number 49,200 – estimated output number 50,000 = -800 (a negative value). A negative value indicates that the entire execution finishes earlier and has not completed yet.
   b) Try to shorten the acceleration time (D1343) into 250ms and deceleration time (D1348) into 550ms.

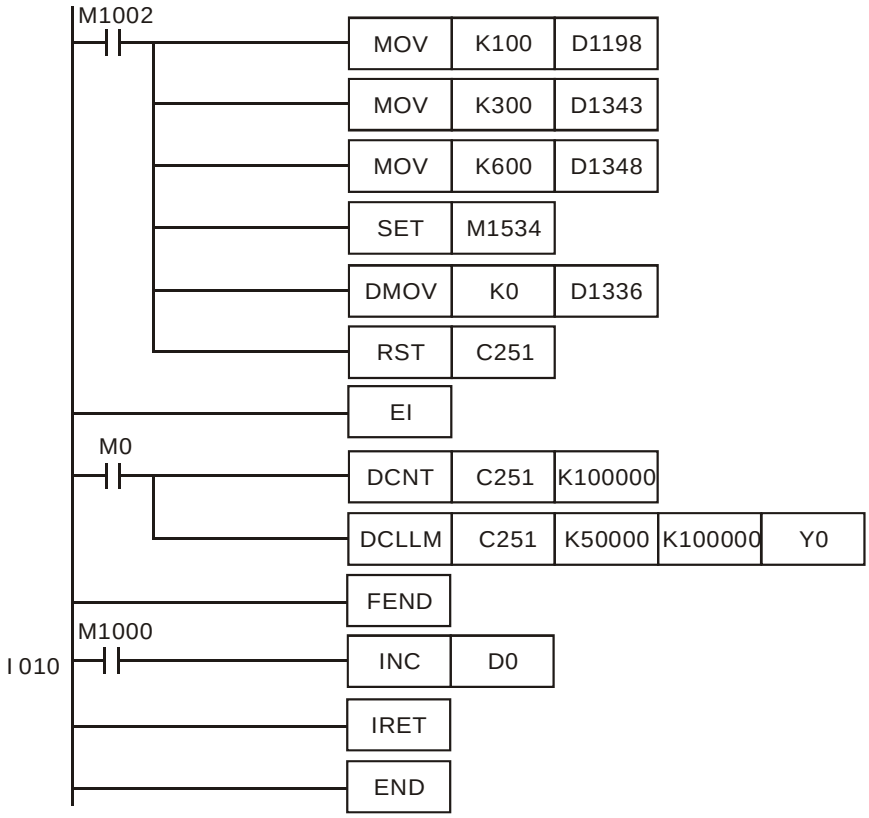
5. Obtain the result of the second execution:



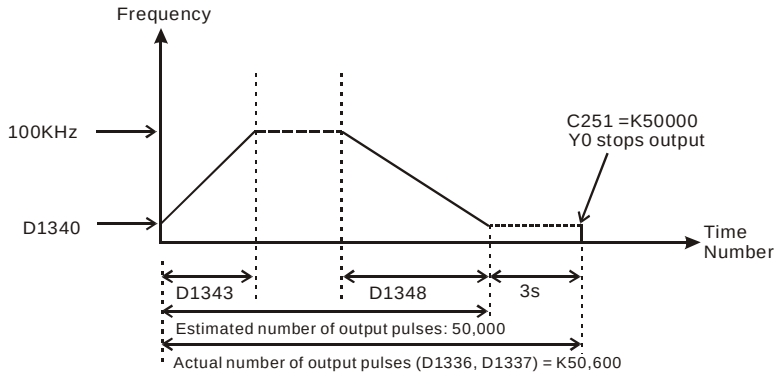
6. Observe the result of the second execution:
   a) The actual output number 50,020 – estimated output number 50,000 = 20
   b) 20 x (1/200Hz) = 100ms (idling time)
   c) 100ms is an appropriate value. Therefore, set the acceleration time as 250ms and deceleration time as 550ms to complete the design.

Program Example 2:

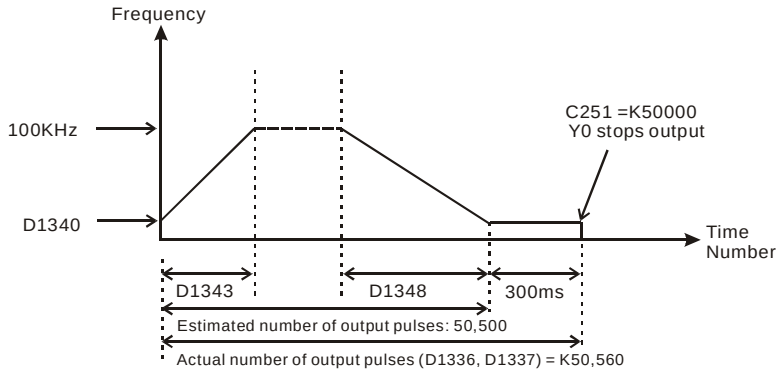
1. Assume the feedback of the encoder is an A/B phase input and we adopt C251 timing (we suggust you clear it to 0 before the execution); target number of feedbacks = 50,000; target output frequency = 100KHz; Y0, Y1 (CH0) as output pulses; start/end frequency (D1340) = 200Hz; acceleration time (D1343) = 300ms; deceleration time (D1348) = 600ms; precentage value (D1198) = 100; current number of output pulses (D1336, D1337) = 0.
2. Write the program codes as follows:



3. Assume the first execution result as:



4. Observe the result of the first execution:
   a) The actual output number 50,600 – estimated output number 50,000 = 600
   b) 600 x (1/200Hz) = 3s (idling time)
   c) 3 seconds are too long. Therefore, increase the percentage value (D1198) to K101.
5. Obatin the result of the second execution:



6. Observe the result of the second execution:
   a) The actual output number 50,560 – estimated output number 50,500 = 60
   b) 60 x (1/200Hz) = 300ms (idling time)
   c) 300ms is an appropriate value. Therefore, set the percentage value (D1198) as K101 to complete the design.

Remarks:

1. Flag explanations:

M1010：	When On, CH0, CH1, CH2 and CH3 will output pulses when encountering END instruction. Off when the output starts.
M1029：	On when CH0 pulse output is completed.
M1030：	On when CH1 pulse output is completed.
M1036：	On when CH2 pulse output is completed.
M1037：	On when CH3 pulse output is completed.
M1334：	When On, CH0 pulse output will be forbidden.
M1335：	When On, CH1 pulse output will be forbidden.

M1520： When On, CH2 pulse output will be forbidden.

M1521： When On, CH3 pulse output will be forbidden.

M1336： CH0 pulse output indication flag

M1337： CH1 pulse output indication flag

M1522： CH2 pulse output indication flag

M1523： CH3 pulse output indication flag

M1305： CH0 direction signal flag

M1306： CH1 direction signal flag

M1532： CH2 direction signal flag

M1533： CH3 direction signal flag

M1534： Deceleration time of CH0 setup flag (must used with D1348)

M1535： Deceleration time of CH1 setup flag (must used with D1349)

M1536： Deceleration time of CH2 setup flag (must used with D1350)

M1537： Deceleration time of CH3 setup flag (must used with D1351)

2. Special register explanations:

D1198： Close loop output/input ratio of CH0 (default: K100)

D1199： Close loop output/input ratio of CH1 (default: K100)

D1478： Close loop output/input ratio of CH2 (default: K100)

D1479： Close loop output/input ratio of CH3 (default: K100)

D1220： Phase setting of CH0 (Y0, Y1): determined by the last 2 digits of D1220; other digits are invalid.

1. K0: Y0 output
2. K1: Y0, Y1 AB-phase output; A ahead of B
3. K2: Y0, Y1 AB-phase output; B ahead of A

D1221： Phase setting of CH1 (Y2, Y3): determined by the last 2 digits of D1221; other digits are invalid.

1. K0: Y2 output
2. K1: Y2, Y3 AB-phase output; A ahead of B
3. K2: Y2, Y3 AB-phase output; B ahead of A

D1229： Phase setting of CH2 (Y4, Y5): determined by the last 2 digits of D1229; other digits are invalid.

1. K0: Y4 output
2. K1: Y4, Y5 AB-phase output; A ahead of B
3. K2: Y4, Y5 AB-phase output; B ahead of A

D1230： Phase setting of CH3 (Y6, Y7): determined by the last 2 digits of D1230; other digits are invalid.

1. K0: Y6 output
2. K1: Y6, Y7 AB-phase output; A ahead of B
3. K2: Y6, Y7 AB-phase output; B ahead of A

D1222： Time difference between the direction signal and pulse output of CH0

D1223： Time difference between the direction signal and pulse output of CH1

D1383： Time difference between the direction signal and pulse output of CH2

D1384： Time difference between the direction signal and pulse output of CH3

D1336： Low word of the current number of output pulses of CH0

D1337： High word of the current number of output pulses of CH0

D1338： Low word of the current number of output pulses of CH1

D1339： High word of the current number of output pulses of CH1

D1375： Low word of the current number of output pulses of CH2

D1376： High word of the current number of output pulses of CH2

D1377： Low word of the current number of output pulses of CH3

D1378： High word of the current number of output pulses of CH3

D1340： Start/end frequency settings of CH0 (default: K200)

D1352： Start/end frequency settings of CH1 (default: K200)

D1379： Start/end frequency settings of CH2 (default: K200)

D1380： Start/end frequency settings of CH3 (default: K200)

D1348： Deceleration time of CH0 pulse output when M1534 = On (default: K100)

D1349： Deceleration time of CH1 pulse output when M1535 = On (default: K100)

D1350： Deceleration time of CH2 pulse output when M1536 = On (default: K100)

D1351： Deceleration time of CH3 pulse output when M1537 = On (default: K100)

D1343： Acceleration/deceleration time of CH0 pulse output (default: K100)

D1353： Acceleration/deceleration time of CH1 pulse output (default: K100)

D1381： Acceleration/deceleration time of CH2 pulse output (default: K100)

D1382： Acceleration/deceleration time of CH3 pulse output (default: K100)

MEMO

API	Mnemonic		Operands	Function	Controllers		
202	SCAL	P	(S1) (S2) (S3) (D)	Proportional Value Calculation	ES/EX/SS	SA/SX/SC	EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			SCAL, SCALP: 9 steps
S_2					*	*							*			
S_3					*	*							*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Source value **S_2**: Slope **S_3**: Offset **D**: Destination device

Explanations:

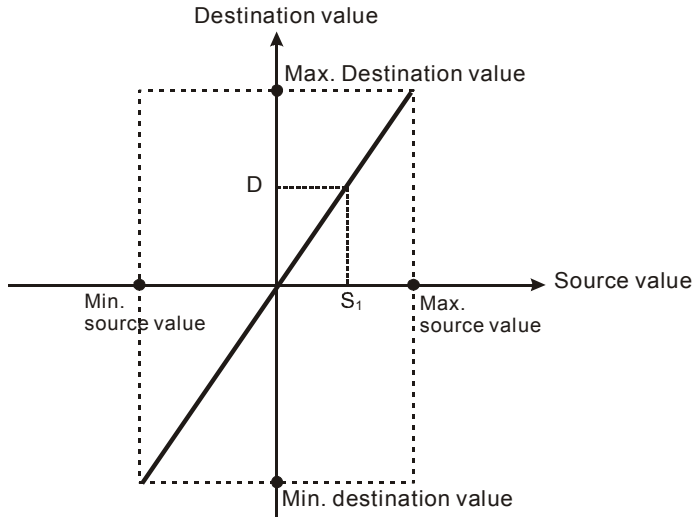
1. Range of S1, S2, S3: -32,768 ~ 32,767
2. Unit of S2: 0.001
3. See the specifications of each model for their range of use.
4. Operation equation in the instruction: **D** = (**S_1** × **S_2**) ÷ 1000 + **S_3**.

   Users have to obtain **S_2** and **S_3** (decimals are rounded up into 16-bit integers) by using the slope and offset equations below.

   Slope equation: **S_2** = [(max. destination value – min. destination value) ÷ (max. source value – min. source value)] × 1,000

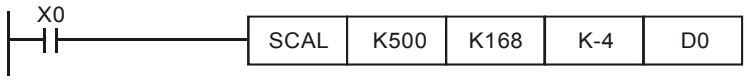
   Offset equation: **S_3** = min. destination value – min. source value × **S_2** ÷ 1,000

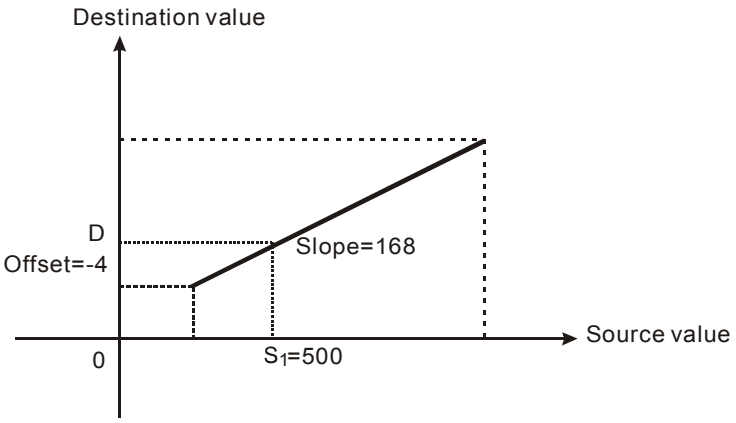
   The output curve is shown as the figure:



Program Example 1:

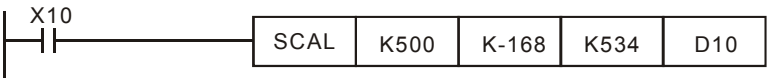
1. Assume **S_1** = 500, **S_2** = 168, **S_3** = -4. When X0 = On, SCAL instruction will be executed and obtain the proportional value at D0.
2. Equation: D0 = (500 × 168) ÷ 1,000 + (-4) = 80

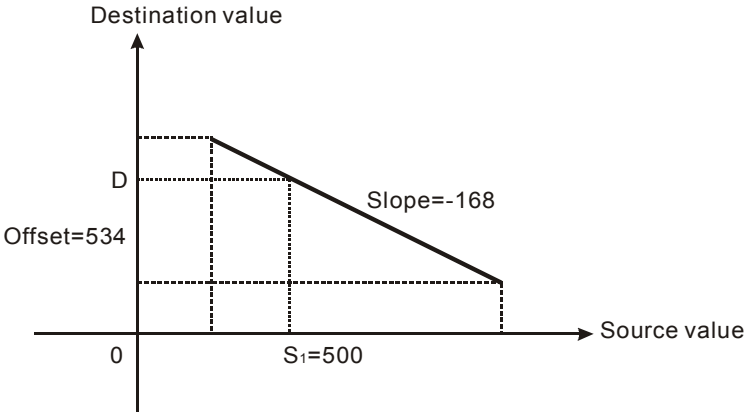




Program Example 2:

1. Assume **S_1** = 500, **S_2** = -168, **S_3** = 534. When X10 = On, SCAL instruction will be executed and obtain the proportional value at D10.
2. Equation: D0 = (500 × -168) ÷ 1,000 + 534 = 450





Remarks:

1. This instruction is applicable for known slope and offset. If slope and offset are unknown, use SCLP instruction for the calculation.
2. **S_2** has to be within the range -32,768 ~ 32,767. If **S_2** falls without the range, use SCLP instruction for the calculation.
3. When using the slope equation, please be aware that the max. source value must > min. source value, but it is not necessary that max. destination value > min. destination value.
4. If the value of **D** > 32,767, **D** = 32,767; if the value of **D** < -32,768, **D** = -32,768.
5. Only ES_V6.2, SA/SX_V1.6, SC_V1.4, EH2/SV_V1.0 and versions above support this instruction. EH series MPU does not support this instruction.

	Mnemonic			Operands	Function	Controllers	
203	D	SCLP	P	(S1) (S2) (D)	Parameter Proportional Value Calculation	ES/EX/SS	SA/SX/SC / EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			SCLP, SCLPP: 7 steps DSCLP, DSCLPP: 13 steps
S_2													*			
D													*			

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Source value **S_2**: Parameter **D**: Destination device

Explanations:

1. See the specifications of each model for the range of operands.
2. Flags: M1162 (decimal integer or binary floating point); M1162 = On -> Binary floating point
3. Settings of **S_2** for 16-bit instruction:

   S_2 occupies 4 consecutive devices in 16-bit instruction.

Device No.	Parameter	Range
S_2	Maximum source value	-32,768 ~ 32,767
S_2 + 1	Minimum source value	-32,768 ~ 32,767
S_2 + 2	Maximum destination value	-32,768 ~ 32,767
S_2 + 3	Minimum destination value	-32,768 ~ 32,767

4. Settings of **S_2** for 32-bit instruction:

   S_2 occupies 8 consecutive devices in 32-bit instruction.

Device No.	Parameter	Range	
		Integer	Floating point
S_2, **S_2** + 1	Maximum source value	-2,147,483,648 ~ 2,147,483,647	Range of 32-bit floating point
S_2 + 2, 3	Minimum source value		
S_2 + 4, 5	Maximum destination value		
S_2 + 6, 7	Minimum destination value		

5. Operation equation in the instruction: **D** = [(**S_1** – min. source value) × (max. destination value – min. destination value)] ÷ (max. source value – min. source value) + min. destination value
6. The operational relation between source value and destination value is as stated below:

   y = kx + b

   y= Destination value (**D**)

   k= Slope = (max. destination value – min. destination value) ÷ (max. source value – min. source value)

   x= Source value (**S_1**)

   b= Offset = Min. destination value – Min. source value × slope

   Bring all the parameters into equation y = kx + b and obtain the equation in the instruction:

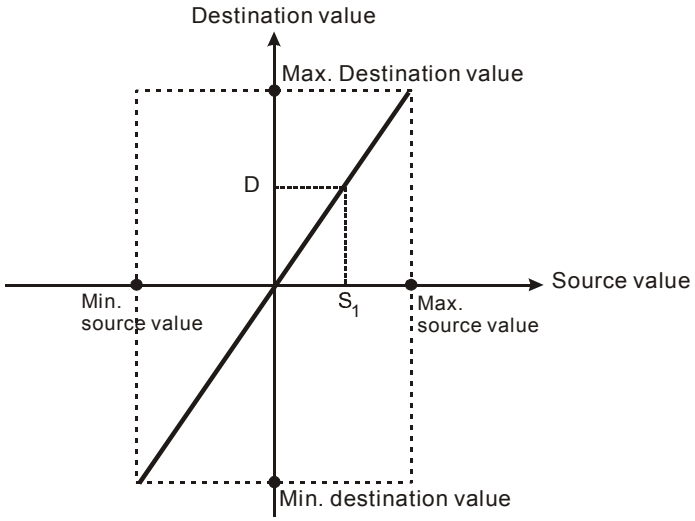
   y = kx + b = D = k **S_1** + b = slope ×**S_1** + offset = slope ×**S_1** + min. destination value – min. source value × slope

= slope × (**S_1** – min. source value) + min. destination value = (**S_1** – min. source value) × (max. destination value – min. destination value) ÷ (max. source value – min. source value) + min. destination value

7. If **S1** > max. source value, **S1** = max. source value

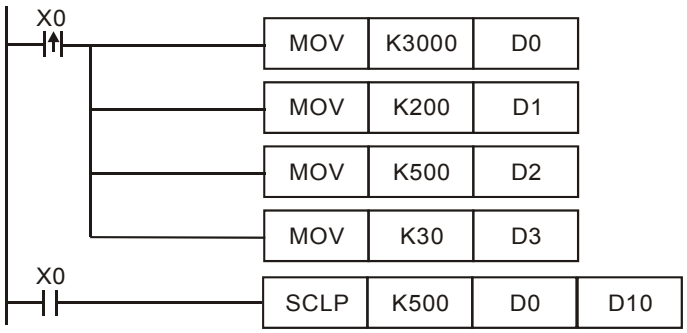
   If **S1** < min. source value, **S1** = min. source value

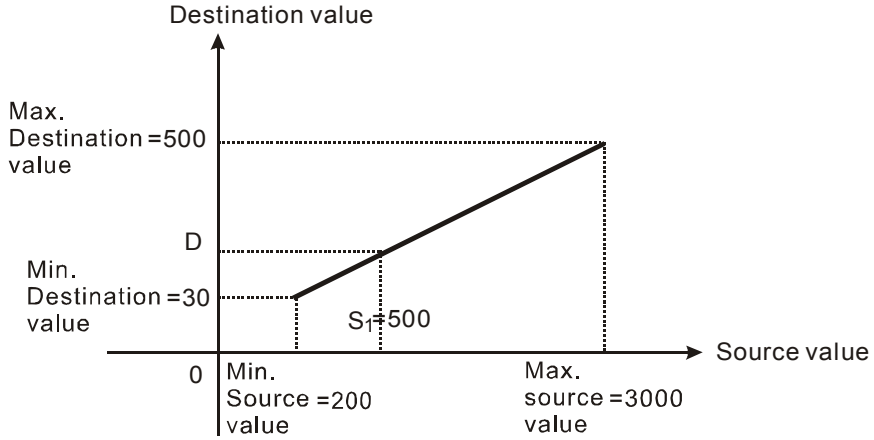
   When all the input values and parameters are set, the output curve is shown as the figure:



Program Example 1:

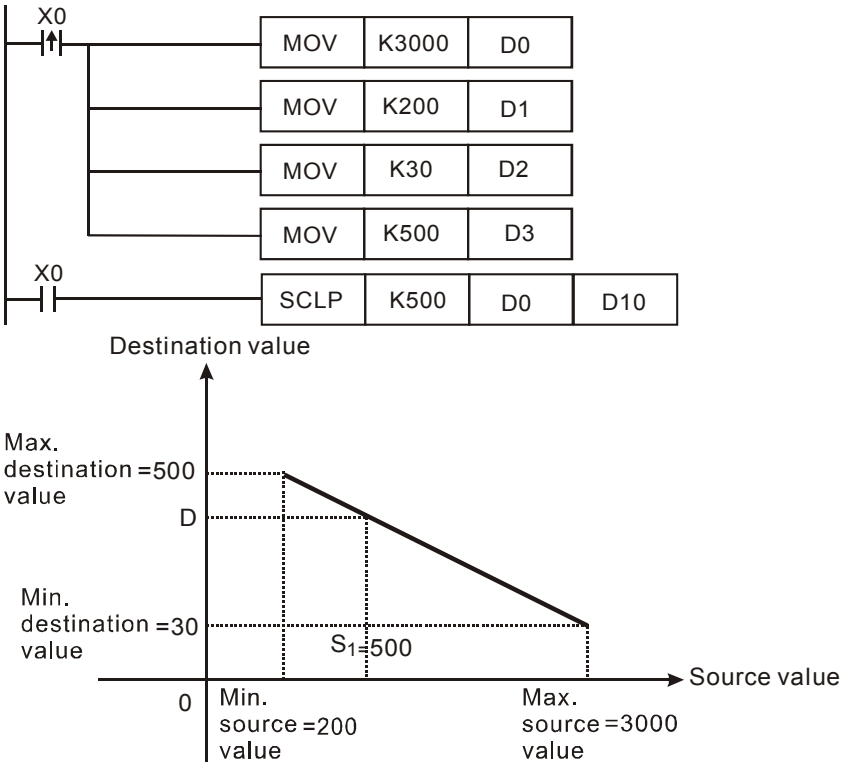
1. Assume **S_1** = 500, max. source value D0 = 3,000, min. source value D1 = 200, max. destination value D2 = 500, and min. destination value D3 = 30. When X0 = On, SCLP instruction will be executed and obtain the proportional value at D10.
2. Equation: D10 = [(500 – 200) × (500 – 30)] ÷ (3,000 – 200) + 30 = 80.35. Round off the result into an integer D10 = 80.





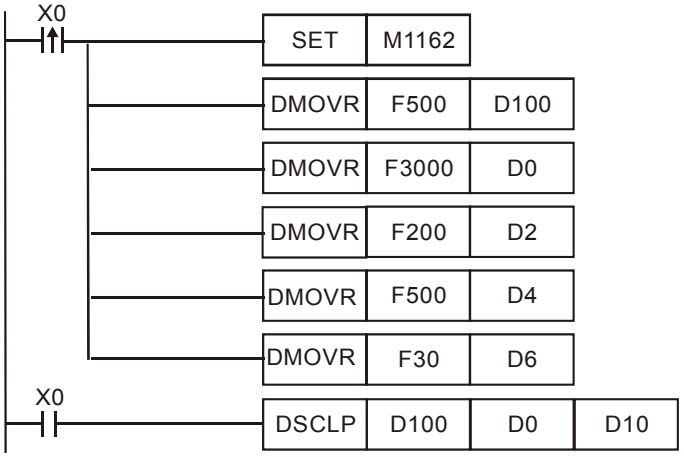
Program Example 2:

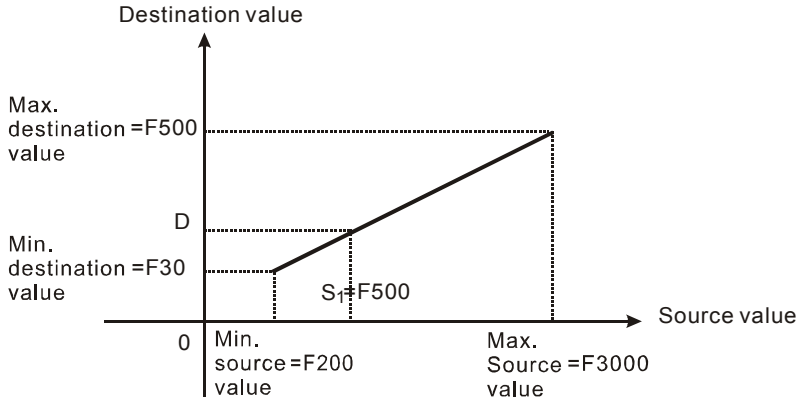
1. Assume S_1 = 500, max. source value D0 = 3,000, min. source value D1 = 200, max. destination value D2 = 30, and min. destination value D3 = 500. When X0 = On, SCLP instruction will be executed and obtain the proportional value at D10.
2. Equation: D10 = [(500 – 200) × (30 – 500)] ÷ (3,000 – 200) + 500 = 449.64. Round off the result into an integer D10 = 450.



Program Example 3:

1. Assume the source of S_1 D100 = F500, max. source value D0 = F3000, min. source value D2 = F200, max. destination value D4 = F500, and min. destination value D6 = F30. When X0 = On, set up M1162, adopt floating point operation and execute DSCLP instruction. The proportional value will be obtained at D10.
2. Equation: D10 = [(F500 – F200) × (F500 – F30)] ÷ (F3000 – F200) + F30 = F80.35. Round off the result into an integer D10 = F80.





Remarks:

1. Range of S_1 for 16-bit instruction: max. source value ≥ S_1 ≥ min. source value; -32,768 ~ 32,767. If the value falls without the bounds, the bound value will be used for calculation.
2. Range of integer S_1 for 32-bit instruction: max. source value ≥ S_1 ≥ min. source value; -2,147,483,648 ~ 2,147,483,647. If the value falls without the bounds, the bound value will be used for calculation.
3. Range of floating point S_1 for 32-bit instruction: max. source value ≥ S_1 ≥ min. source value; following the range of 32-bit floating point. If the value falls without the bounds, the bound value will be used for calculation.
4. Please be aware that the max. source value must > min. source value, but it is not necessary that max. destination value > min. destination value.
5. Only ES_V6.2, SA/SX_V1.6, SC_V1.4, EH2/SV_V1.0 and versions above support this instruction. EH series MPU does not support this instruction.

API	Mnemonic	Operands	Function	Controllers
215~217	D LD#	S_1 S_2	Contact Logical Operation LD#	ES/EX/SS SA/SX/SC EH/SV

OP \ Type	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	LD#: 5 steps DLD#: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Data source device 1 **S_2**: Data source device 2

Explanations:

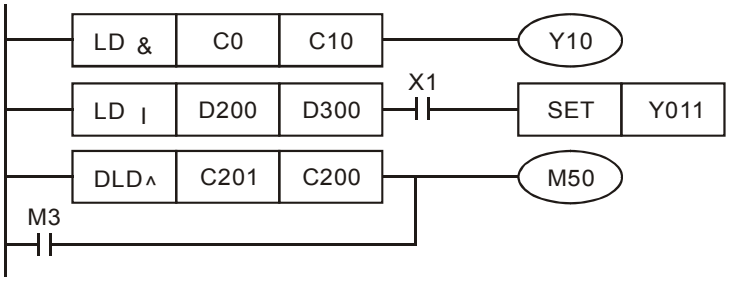
1. See the specifications of each model for the range of operands.
2. This instruction compares the content in **S_1** and **S_2**. If the result is not "0", the continuity of the instruction is enabled. If the result is "0", the continuity of the instruction is disabled.
3. LD# (#**:** &, |, ^) instruction is used for direct connection with BUS.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
215	LD&	**D**LD&	**S_1** & **S_2**≠0	**S_1** & **S_2**=0
216	LD|	**D**LD|	**S_1** | **S_2**≠0	**S_1** | **S_2**=0
217	LD^	**D**LD^	**S_1** ^ **S_2**≠0	**S_1** ^ **S_2**=0

4. **&:** Logical "AND" operation
5. **|:** Logical "OR" operation
6. **^:** Logical "XOR" operation
7. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DLD#). If 16-bit instructions (LD#) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When the result of logical AND operation of C0 and C10 ≠ 0, Y10 = On.
2. When the result of logical OR operation of D200 and D300 ≠ 0 and X1 = On, Y11 = On will be retained.
3. When the result of logical XOR operation of C201 and C200 ≠ 0 or M3 = On, M50 = On.



API	Mnemonic	Operands	Function	Controllers
218~220	D AND#	S_1 S_2	Contact Logical Operation AND#	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	AND#: 5 steps DAND#: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Data source device 1 **S_2**: Data source device 2

Explanations:

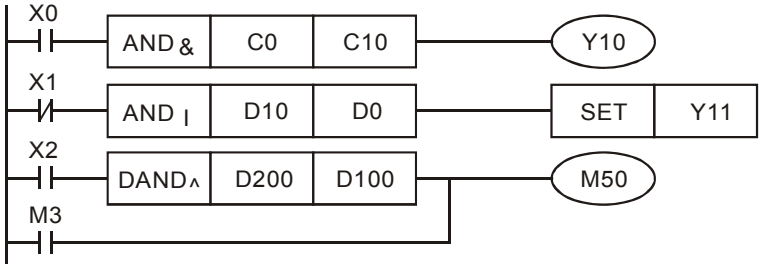
1. See the specifications of each model for the range of operands.
2. This instruction compares the content in **S_1** and **S_2**. If the result is not "0", the continuity of the instruction is enabled. If the result is "0", the continuity of the instruction is disabled.
3. AND# (#**:** &, |, ^) is an operation instruction used on series contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
218	AND&	DAND&	S1 & S2≠0	S1 & S2=0
219	AND|	DAND|	S1 | S2≠0	S1 | S2=0
220	AND^	DAND^	S1 ^ S2≠0	S1 ^ S2=0

4. **&:** Logical "AND" operation
5. **|:** Logical "OR" operation
6. **^:** Logical "XOR" operation
7. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DAND#). If 16-bit instructions (AND#) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When X0 = On and the result of logical AND operation of C0 and C10 ≠ 0, Y10 = On.
2. When X1 = Off and the result of logical OR operation of D10 and D0 ≠ 0 and X1 = On, Y11 = On will be retained.
3. When X2 = On and the result of logical XOR operation of 32-bit register D200 (D201) and 32-bit register D100 (D101) ≠ 0 or M3 = On, M50 = On.



API	Mnemonic	Operands	Function	Controllers
221~223	D OR#	S_1 S_2	Contact Logical operation OR#	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	OR#: 5 steps DOR#: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Data source device 1 **S_2**: Data source device 2

Explanations:

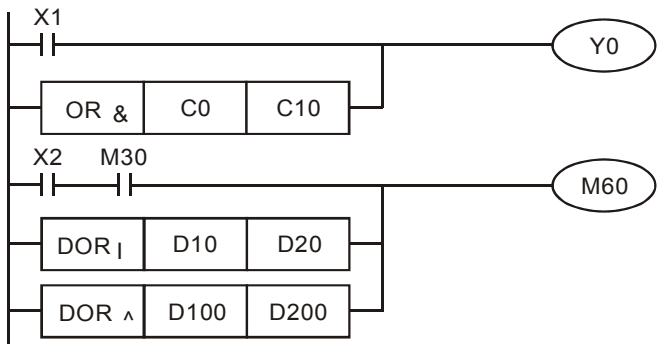
1. See the specifications of each model for the range of operands.
2. This instruction compares the content in **S_1** and **S_2**. If the result is not "0", the continuity of the instruction is enabled. If the result is "0", the continuity of the instruction is disabled.
3. OR# (#**:** &, |, ^) is an operation instruction used on parallel contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
221	OR&	**D**OR&	**S_1** & **S_2**≠0	**S_1** & **S_2**=0
222	OR|	**D**OR|	**S_1** | **S_2**≠0	**S_1** | **S_2**=0
223	OR^	**D**OR^	**S_1** ^ **S_2**≠0	**S_1** ^ **S_2**=0

4. **&:** Logical "AND" operation
5. **|:** Logical "OR" operation
6. **^:** Logical "XOR" operation
7. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DOR#). If 16-bit instructions (OR#) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When X1 = On and the result of logical AND operation of C0 and C10 ≠ 0, Y10 = On.
2. M60 will be On when X2 = On and M30 = On, or the result of logical OR operation of 32-bit register D10 (D11) and 32-bit register D20 (D21) ≠ 0, or the result of logical XOR operation of 32-bit register D200 (D201) and 32-bit counter C235 ≠ 0.



API	Mnemonic	Operands	Function	Controllers
224~230	D LD※	(S1) (S2)	LoaD Compare	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	LD※: 5 steps DLD※: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Data source device 1 **S_2**: Data source device 2

Explanations:

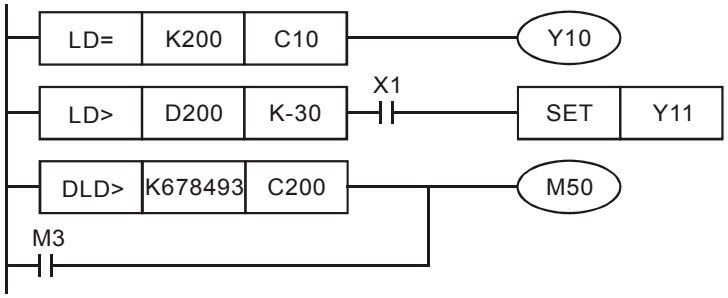
1. See the specifications of each model for the range of operands.
2. This instruction compares the content in **S_1** and **S_2**. Take API224 (LD=) for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
3. LD※ (**※:** =, >, <, <>, ≤, ≥) instruction is used for direct connection with BUS.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
224	LD=	**D**LD=	**S_1**=**S_2**	**S_1**≠**S_2**
225	LD>	**D**LD>	**S_1**>**S_2**	**S_1**≦**S_2**
226	LD<	**D**LD<	**S_1**<**S_2**	**S_1**≧**S_2**
228	LD<>	**D**LD<>	**S_1**≠**S_2**	**S_1**=**S_2**
229	LD<=	**D**LD<=	**S_1**≦**S_2**	**S_1**>**S_2**
230	LD>=	**D**LD>=	**S_1**≧**S_2**	**S_1**<**S_2**

4. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DLD※). If 16-bit instructions (LD※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When the content in C10 = K200, Y10 = On.
2. When the content in D200 > K-30 and X1 = On, Y11= On will be retained.
3. When the content in C200 < K678,493 or M3 = On, M50 = On.



API	Mnemonic	Operands	Function	Controllers
232~238	D AND※	(S1) (S2)	AND Compare	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	AND※: 5 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DAND※: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Data source device 1 **S_2**: Data source device 2

Explanations:

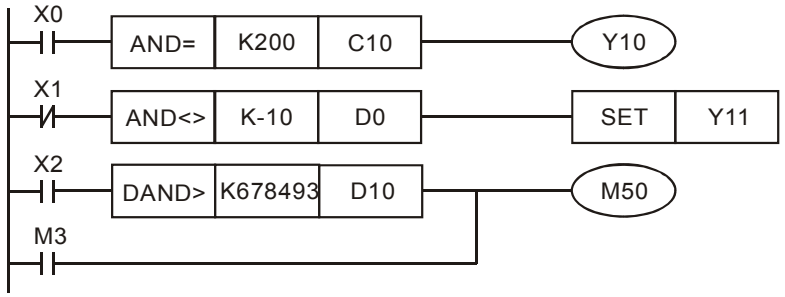
1. See the specifications of each model for the range of operands.
2. This instruction compares the content in **S_1** and **S_2**. Take API232 (AND=) for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
3. AND※ (**※:** =, >, <, <>, ≤, ≥) is a comparison instruction is used on series contacts.

API No.	16 –bit instruction	32 –bit instruction	Continuity condition	No-continuity condition
232	AND＝	**D**AND＝	**S_1**＝**S_2**	**S_1**≠**S_2**
233	AND＞	**D**AND＞	**S_1**＞**S_2**	**S_1**≦**S_2**
234	AND＜	**D**AND＜	**S_1**＜**S_2**	**S_1**≧**S_2**
236	AND＜＞	**D**AND＜＞	**S_1**≠**S_2**	**S_1**＝**S_2**
237	AND＜＝	**D**AND＜＝	**S_1**≦**S_2**	**S_1**＞**S_2**
238	AND＞＝	**D**AND＞＝	**S_1**≧**S_2**	**S_1**＜**S_2**

4. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DAND※). If 16-bit instructions (AND※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When X0 = On and the content in C10 = K200, Y10 = On.
2. When X1 = Off and the content in D0 ≠ K-10, Y11= On will be retained.
3. When X2 = On and the content in 32-bit register D0 (D11) < 678,493 or M3 = On, M50 = On.



API	Mnemonic	Operands	Function	Controllers
240~246	D OR※	(S1) (S2)	OR Compare	ES/EX/SS SA/SX/SC EH/SV

Type / OP	Bit Devices				Word Devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*	*	OR*: 5 steps DOR*: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE								16-bit								32-bit							
ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV	ES	EX	SS	SA	SX	SC	EH	SV

Operands:

S_1: Data source device 1 **S_2**: Data source device 2

Explanations:

1. See the specifications of each model for the range of operands.
2. This instruction compares the content in **S_1** and **S_2**. Take API240 (OR=) for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
3. OR※ (**※:** =, >, <, <>, ≤, ≥) is an comparison instruction used on parallel contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	No-continuity condition
240	OR=	**D**OR=	**S_1**=**S_2**	**S_1**≠**S_2**
241	OR>	**D**OR>	**S_1**>**S_2**	**S_1**≦**S_2**
242	OR<	**D**OR<	**S_1**<**S_2**	**S_1**≧**S_2**
244	OR<>	**D**OR<>	**S_1**≠**S_2**	**S_1**=**S_2**
245	OR<=	**D**OR<=	**S_1**≦**S_2**	**S_1**>**S_2**
246	OR>=	**D**OR>=	**S_1**≧**S_2**	**S_1**<**S_2**

4. When 32-bit counters (C200 ~ C255) are used in this instruction for comparison, make sure to adopt 32-bit instruction (DOR※). If 16-bit instructions (OR※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

Program Example:

1. When X1 = On and the present value of C10 = K200, Y0 = On.
2. M60 will be On when X2 = On, M30 = On and the content in 32-bit register D100 (D101) ≥ K100,000.