automation
СТЭИК
+7 (495) 661-24-61
www.deltronics.ru

DELTA

DVP15MC

DVP50MC

Контроллеры управления движением семейства Delta DVP-MC

Управление приводами по интерфейсу

CANopen

Sync. time: 4 оси за 2 мс

DVP15MC11T (24 оси)
DVP15MC11T-06 (6 осей)

6/24 серво осей
+ 8 виртуальный осей

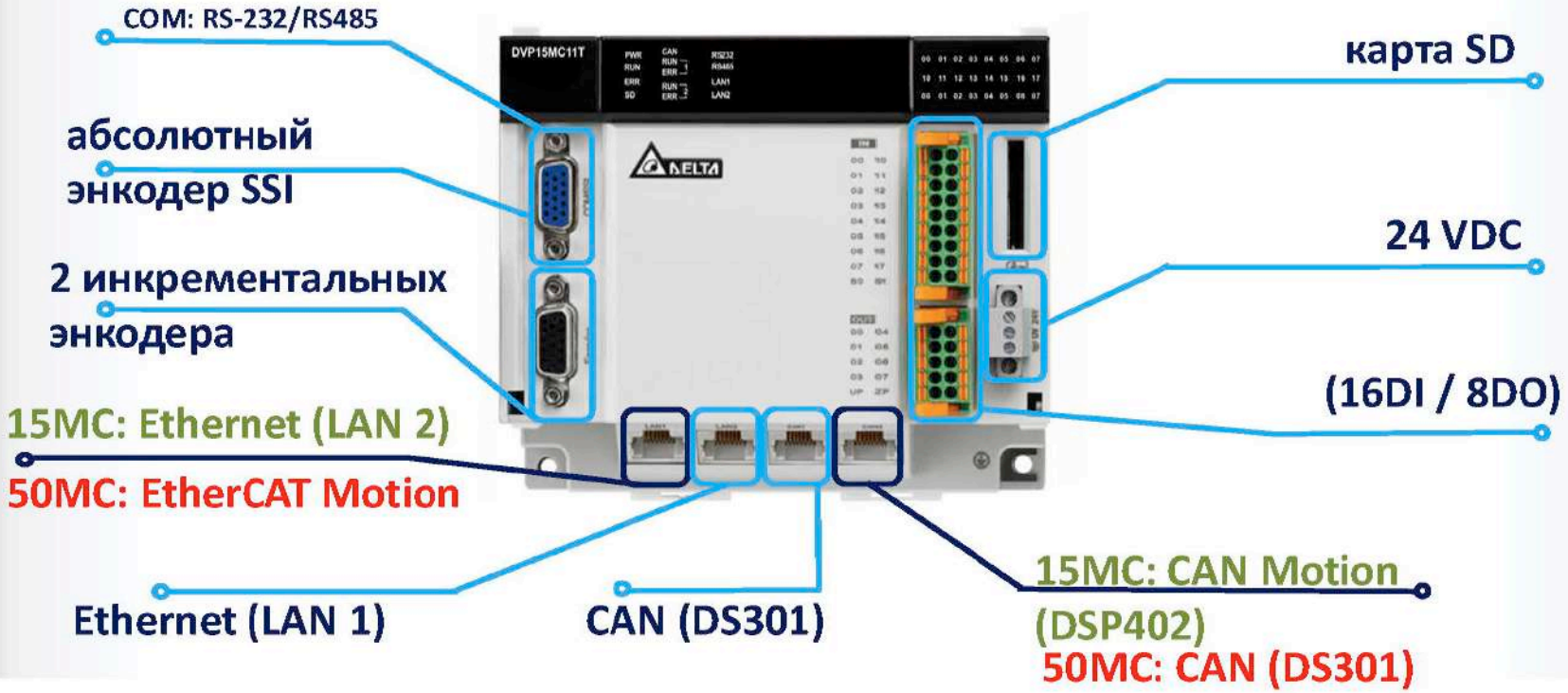
EtherCAT

Sync. time: 24 оси за 1 мс

DVP50MC11T (24 оси)
DVP50MC11T-06 (6 осей)

Интегрированная аппаратная платформа

Богатый набор встроенных интерфейсов



Высокопроизводительный процессор

- Тактовая частота процессора 1 ГГц
- Высокая точность: поддерживается LREAL (формат с плавающей точкой 64 бит)
- Объём программы: 20 Мб; Память данных: 20 Мб

Богатый набор модулей расширения

Расширение бюджетными модулями от серии DVP-S

DVP15MC / DVP50MC

Левосторонние модули

- Макс. 8 модулей
- Коммуникационные
- Аналоговые 16 бит
- Весовые

Правосторонние модули

- Дискретных входов/выходов, макс. 480 точек (14 модулей)
- Аналоговые/температурные, макс. 8 модулей

Управление и связь

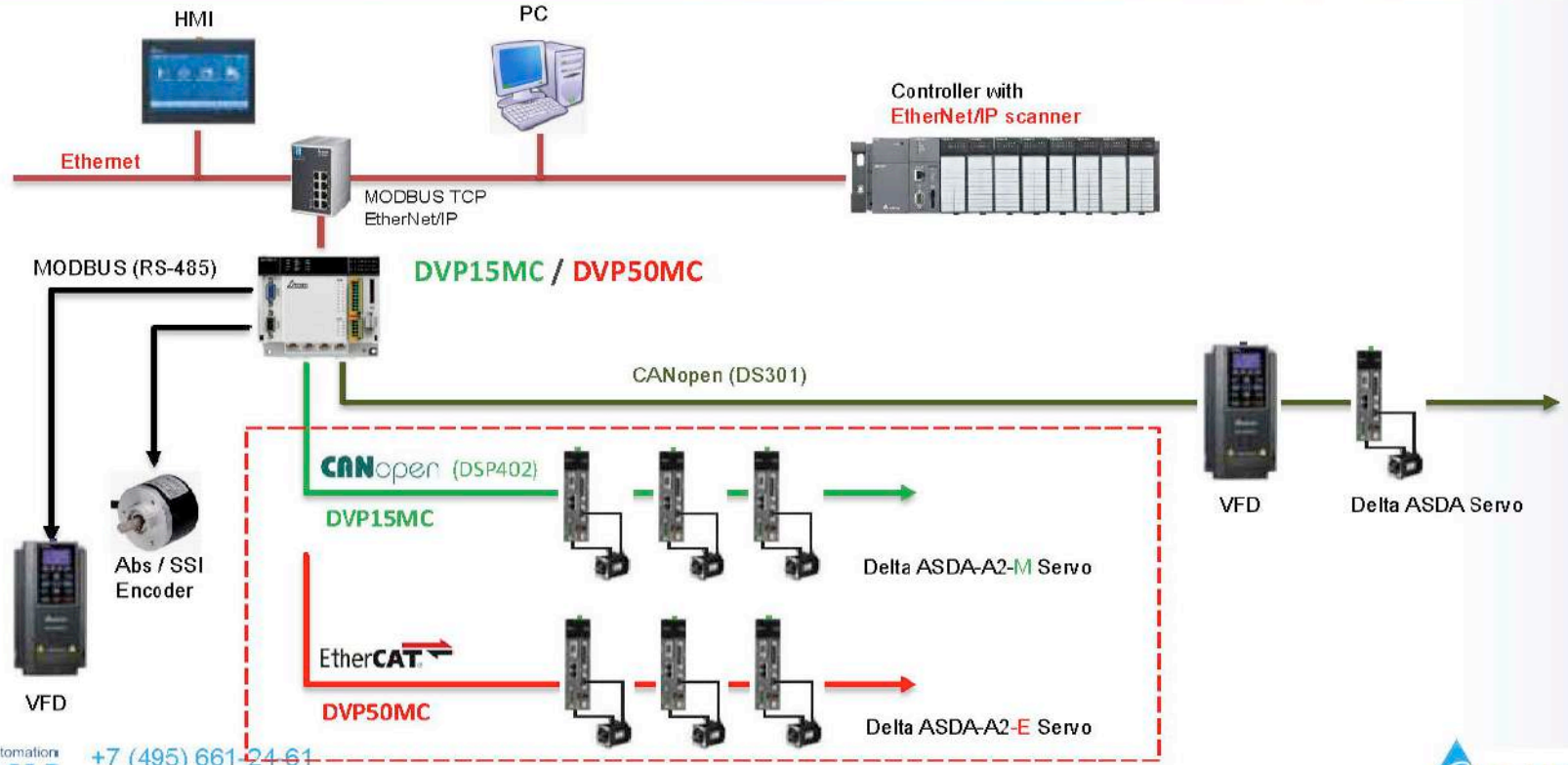
DVP50MC - синхронное управление сервоприводами ASD-A2-E по EtherCAT

DVP15MC - синхронное управление сервоприводами ASD-A2-M по CANopen DS402

DVP50/15MC:

- управление различными ведомыми устройствами по Ethernet (Modbus TCP Server/Client, Ethernet/IP Adapter, Socket TCP/UDP), CANopen DS301, RS485 (Modbus)
- входы для 2 инкрементальных энкодеров и абсолютного энкодера с интерфейсом SSI

Поддержка различных интерфейсов для управления и связи

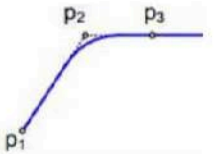


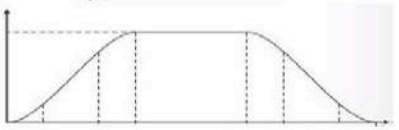
Функции управления движением

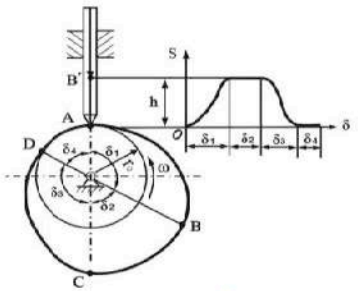
DVP15MC: 4 оси/2 мс
DVP50MC: 24 оси/1 мс

Название модели	Количество физических осей	Количество виртуальных осей	Физические + Виртуальные
DVP15/50MC11T(P)	24	32	32
DVP15/50MC11T(P)-06	6	16	16

- Поддержка библиотек управления движением PLCopen MC
- Библиотека готовых прикладных инструкций Delta Library
- Поддержка буферного режима (безостановочный переход с одного участка траектории на другой)
- Наличие параметра Jerk (максимально плавные разгон и торможение)
- Поддержка энкодеров в качестве мастер оси
- Одноосевое движение: режимы скорости, позиции, момента, возврата в ноль
- Многоосевое движение:
  - Линейная интерполяция до 8 осей
  - Круговая интерполяция до 3 осей
  - Электронный кулачок E-CAM: 64 кривых до 2048 точек каждая
  - Поддержка E-GEAR (зависимое движение одной оси относительно другой)
  - Встроенная процедура барабанной резки
  - Захват текущей позиции по прерыванию



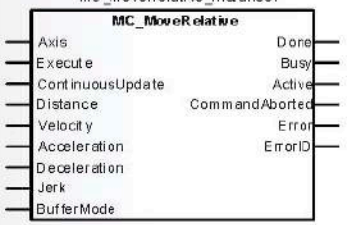




Буферный режим

Переход с одного участка траектории на другой без остановки

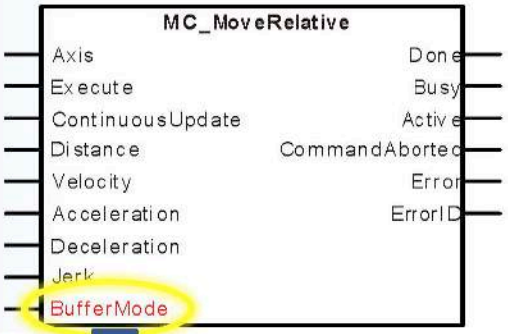
MC_MoveRelative_instance1



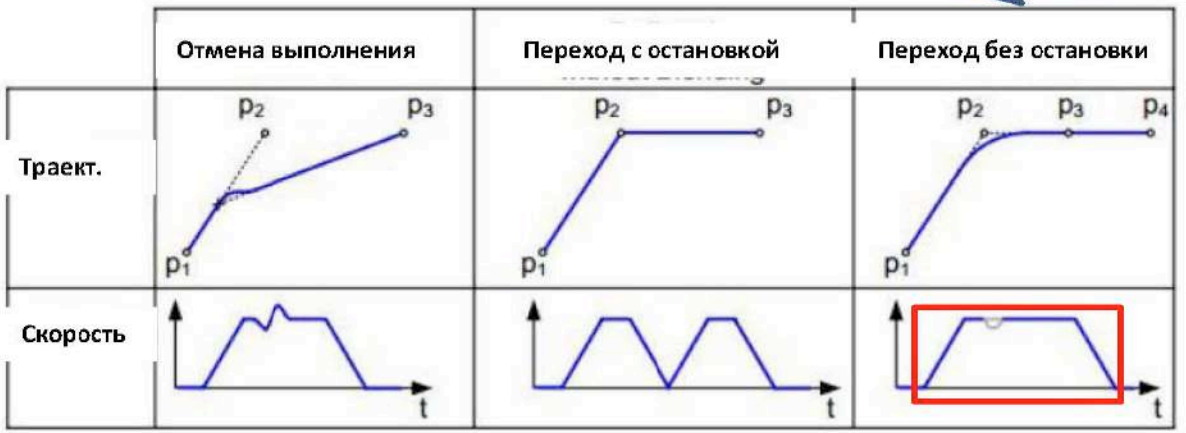
Текущая исполняемая инструкция

Следующая инструкция

MC_MoveRelative_instance2



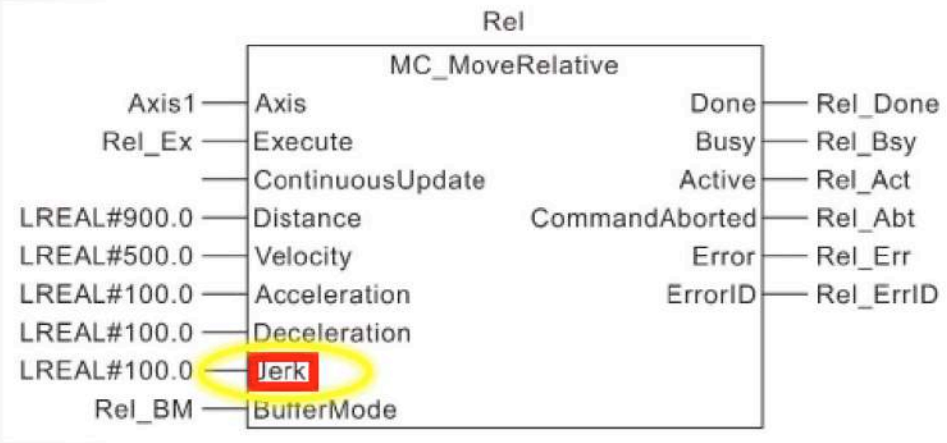
При включении буферного режима переход с одного участка на другой скорость выравнивается по уставкам обеих инструкций без остановки движения

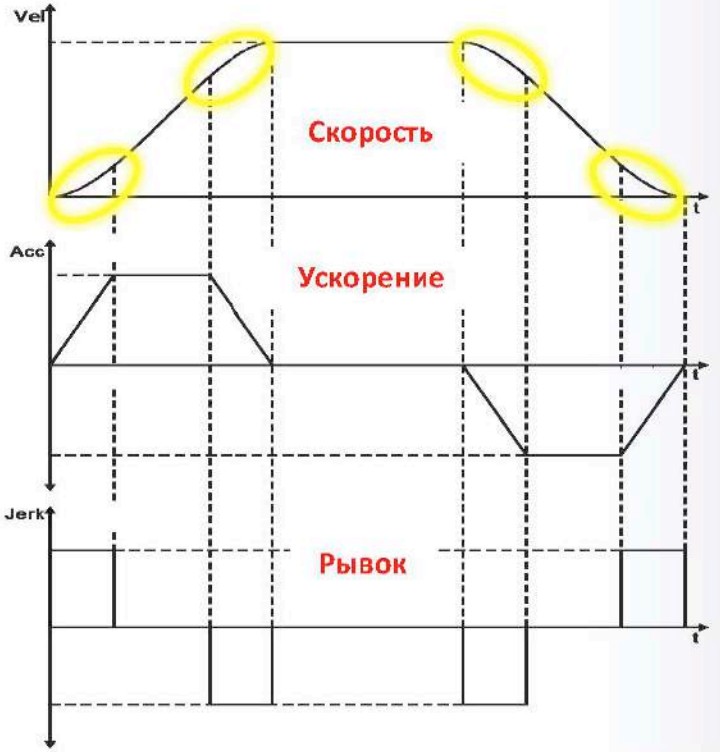


Параметр Jerk

Максимальное сглаживание при управлении скоростью

Jerk – вторая производная от скорости, позволяет максимально плавно управлять ускорением и замедлением при изменениях скорости

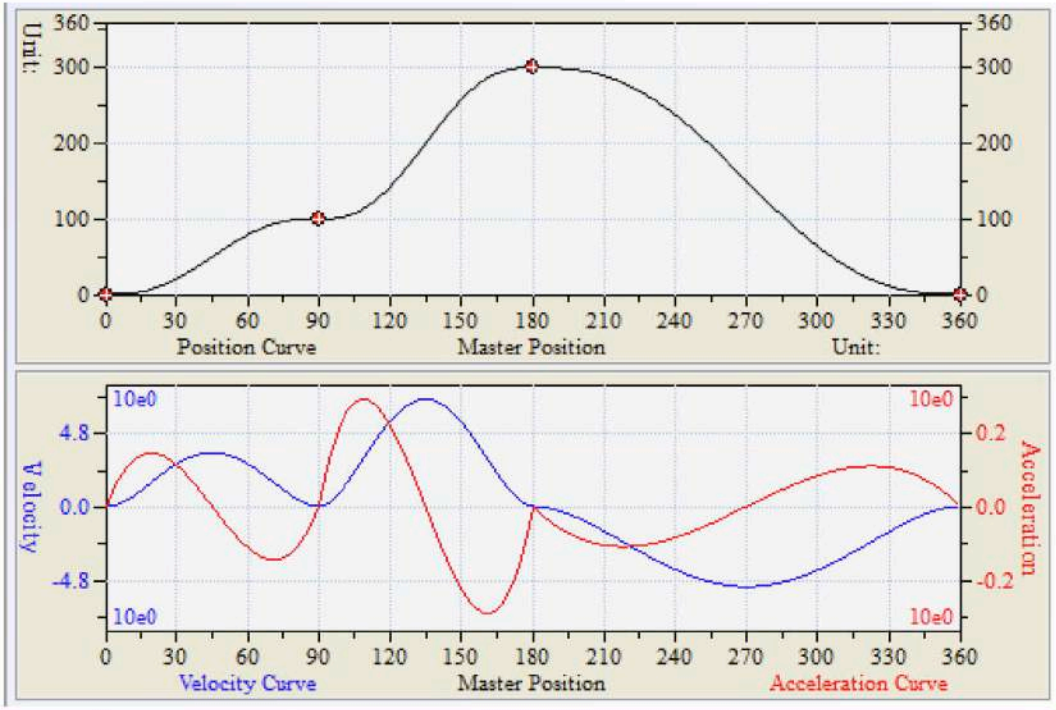




Электронный кулачок E-CAM

Электронный кулачок – соотношение позиций мастер оси (X) и ведомой оси (Y).

Кривая электронного кулачка строится по ключевым точкам. Между точками система осуществляет автоматическое сглаживание для плавности движения. Чем больше ключевых точек, тем точнее и плавнее движение



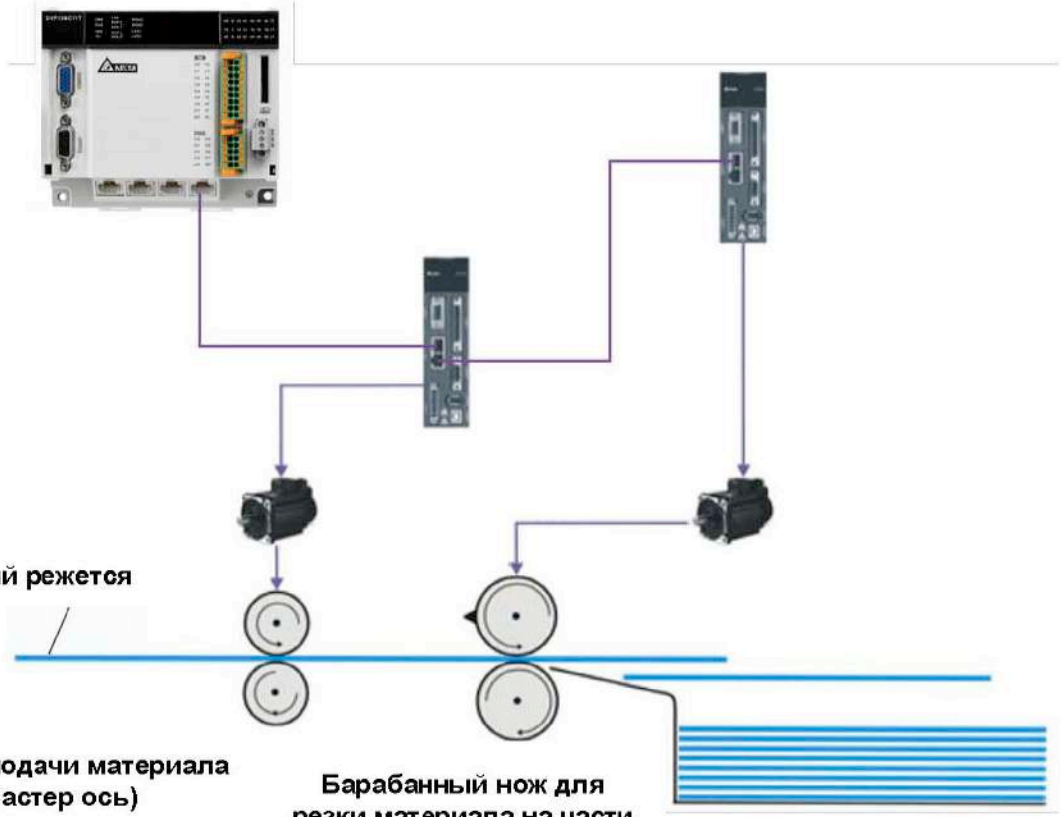
Кривая электронного кулачка

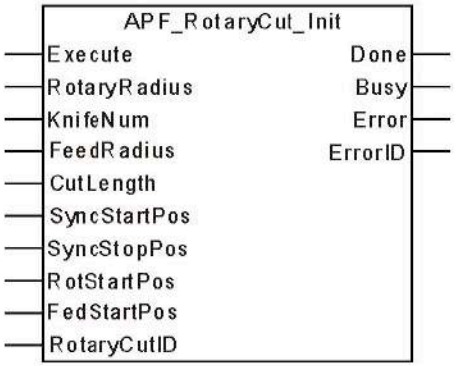
При построении кривой электронного кулачка также рассчитываются графики скорости и ускорения

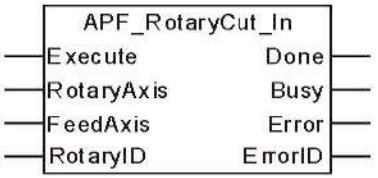
Index	Master	Slave	Velocity	Acceleration	Segment	M
0	0.000	0.000	0.000	0.000		
					Poly5	0
1	90.000	100.000	0.000	0.000		
					Poly5	1
2	180.000	300.000	0.000	-0.000		
					Poly5	0
3	360.000	0.000	0.000	0.000		

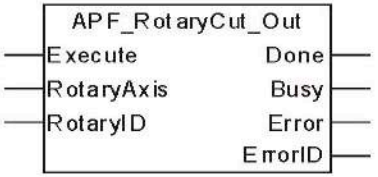
Ключевые точки

Встроенная процедура барабанной резки





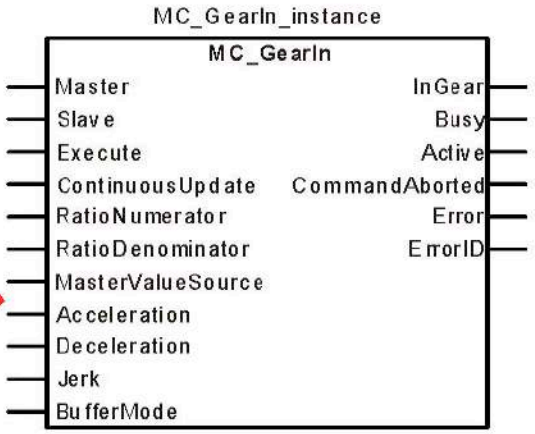




Электронный редуктор E-GEAR

Электронный редуктор – это программная реализация повышения или понижения выходных оборотов относительно входных

Электронный редуктор позволяет реализовать неограниченное число вариантов зависимого движения одной оси относительно другой через простую смену коэффициента редукции в прикладной инструкции, в том числе и изменить направление вращения

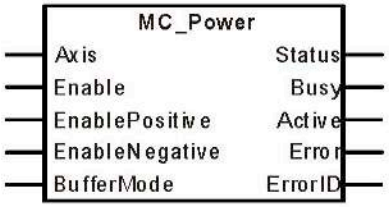


Поддерживают буферный режим

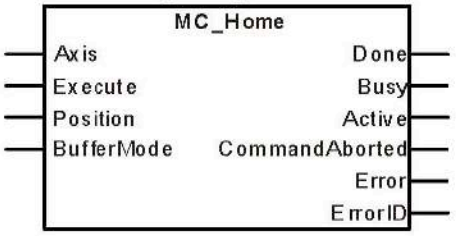
Стандартизованные инструкции управления движением по осям

Позволяют легко и общепринятым способом реализовать перемещение по осям в требуемом режиме: скорости, позиционирования, момента, возврата в ноль, контролировать состояние осей и т.д.

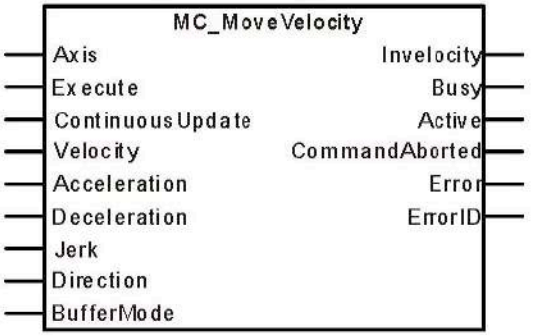
MC_Power_instance



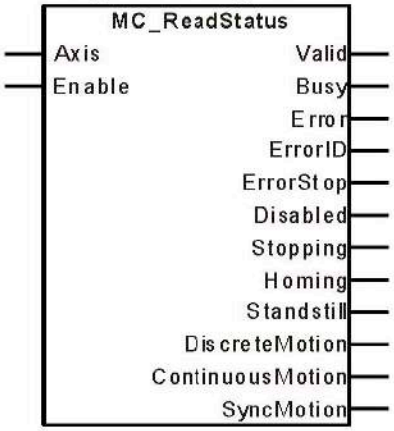
MC_Home_instance



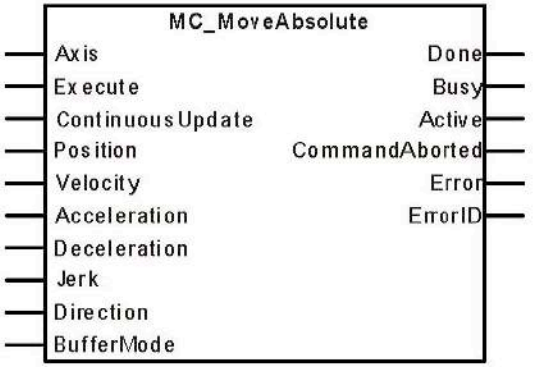
MC_MoveVelocity_instance



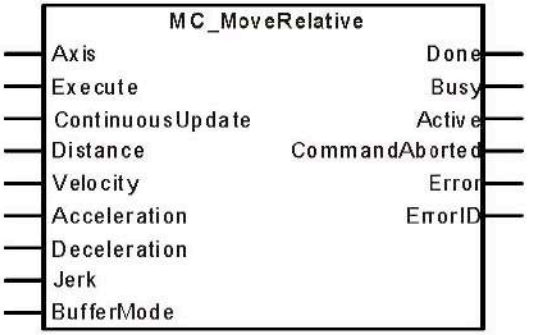
MC_ReadStatus_instance



MC_MoveAbsolute_instance



MC_MoveRelative_instance



DMC_SetTorque_instance

Инструкции линейной и круговой интерполяции

Поддерживают буферный режим

DMC_AddAxisToGroup_instance

DMC_AddAxisToGroup	
AxesGroup	Done
Axis	Busy
Execute	Error
IdentInGroup	ErrorID

Группировка осей

DMC_GroupEnable_instance

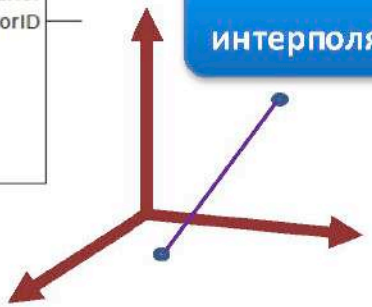
DMC_GroupEnable	
AxesGroup	Status
Enable	Busy
MoveDirectVelocity	CommandAborted
MoveDirectAcceleration	Error
MoveDirectDeceleration	ErrorID
MoveDirectJerk	

Активация группы

DMC_MoveLinearAbsolute_instance

DMC_MoveLinearAbsolute	
AxesGroup	Done
Execute	Busy
Position	Active
Velocity	CommandAborted
Acceleration	Error
Deceleration	ErrorID
Jerk	
CoordSystem	
BufferMode	
TransitionMode	
TransitionParameter	

Линейная интерполяция



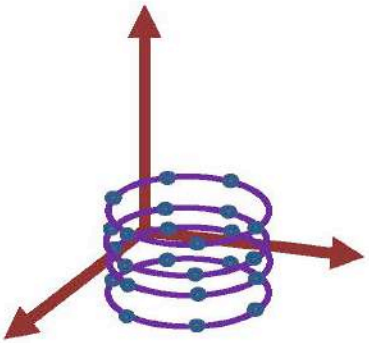
DMC_MoveCircularRelative_instance

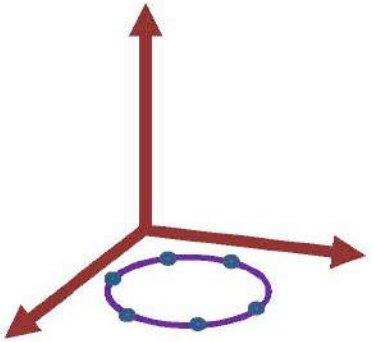
DMC_MoveCircularRelative	
AxesGroup	Done
Execute	Busy
CircMode	Active
AuxPoint	CommandAborted
EndPoint	Error
MultiTurn	ErrorID
PathChoice	
Velocity	
Acceleration	
Deceleration	
Jerk	
CoordSystem	
BufferMode	
TransitionMode	
TransitionParameter	

DMC_MoveCircularAbsolute_instance

DMC_MoveCircularAbsolute	
AxesGroup	Done
Execute	Busy
CircMode	Active
AuxPoint	CommandAborted
EndPoint	Error
MultiTurn	ErrorID
PathChoice	
Velocity	
Acceleration	
Deceleration	
Jerk	
CoordSystem	
BufferMode	
TransitionMode	
TransitionParameter	

3-х осевая интерполяция





Круговая интерполяция

СТОИК automation
+7 (495) 661-24-61
www.deltronics.ru

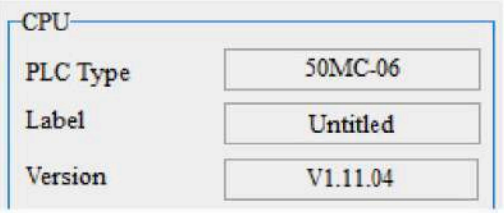
DELTA

Среда программирования

Редактор – **ISPSoft** (версия не ниже 3.10)

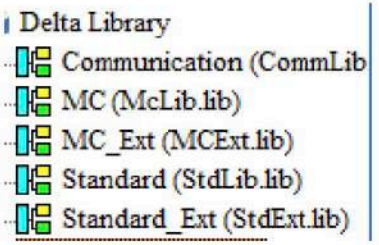
Утилита связи – **COMMGR** (версия не ниже 1.11)

Встроенное ПО (Firmware) – версия не ниже 1.11.04



Языки программирования – **LD** и **ST**

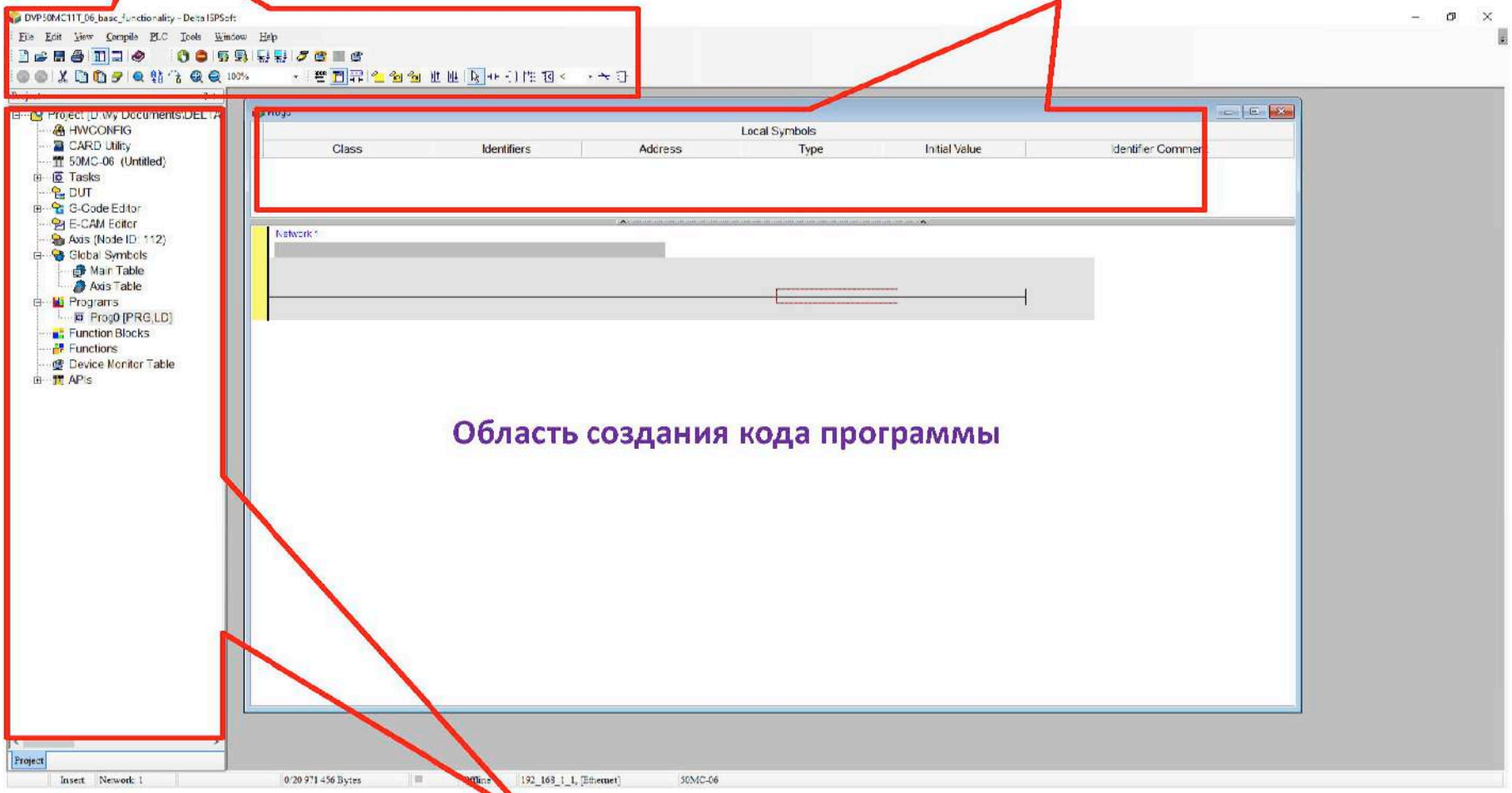
Подключаемые библиотеки – Delta Library: MC, MC_Ext, Standard, Standard_Ext, Communication



Инструменты программирования

Меню редактора ISPSoft.

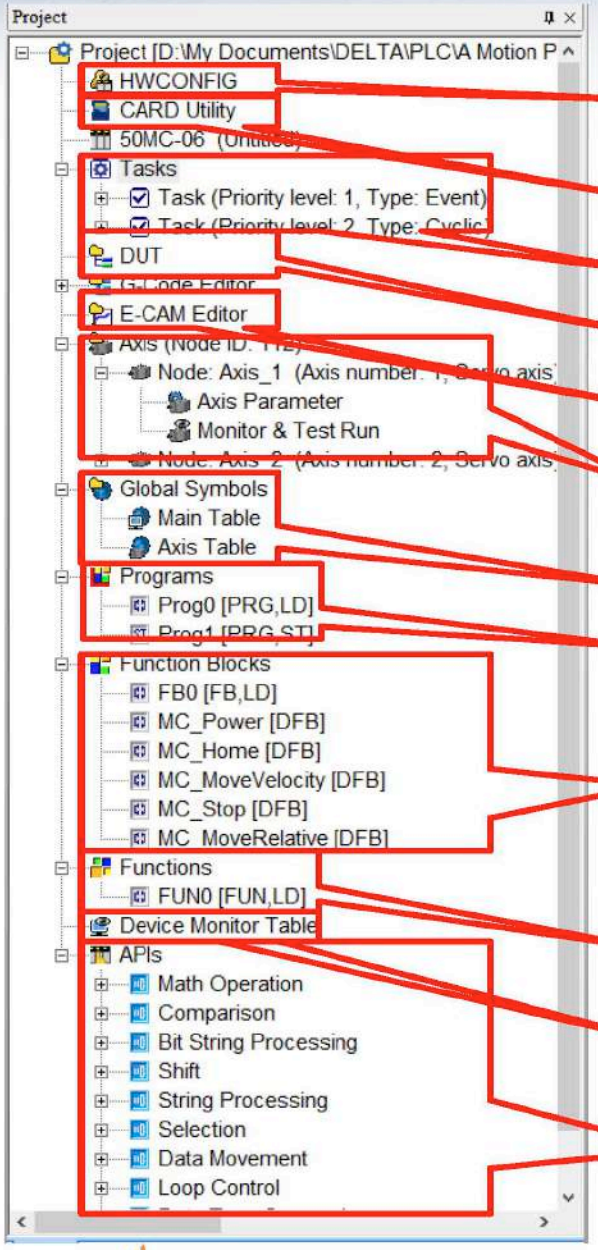
Таблица локальных переменных



Область создания кода программы

Древо проекта

Древо проекта



Конфигуратор аппаратной части, вызов конфигуратора протокола CANopen DS301

Утилита создания резервной копии проекта, в т.ч. в закрытом варианте

Менеджер задач. Определяет тип и порядок исполнения POU

Инструмент создания агрегированных типов данных: структур, объединений и перечислений

Инструмент создания кривых электронного кулачка

Список и параметры подключенных осей. Утилита Monitor & Test Run для проверки связи с сервоприводами и их работоспособности

Таблица глобальных переменных проекта

Список POU (Program Organization Unit - структурная единица программы со своими локальными переменными). Привязка POU к задаче осуществляется в Менеджере Задач

Список Функциональных Блоков как созданных прикладным разработчиком, так и подключенных из библиотек Delta. Функциональные блоки интегрируются в программу путём создания своих экземпляров, т.е. переменных типа ФБ

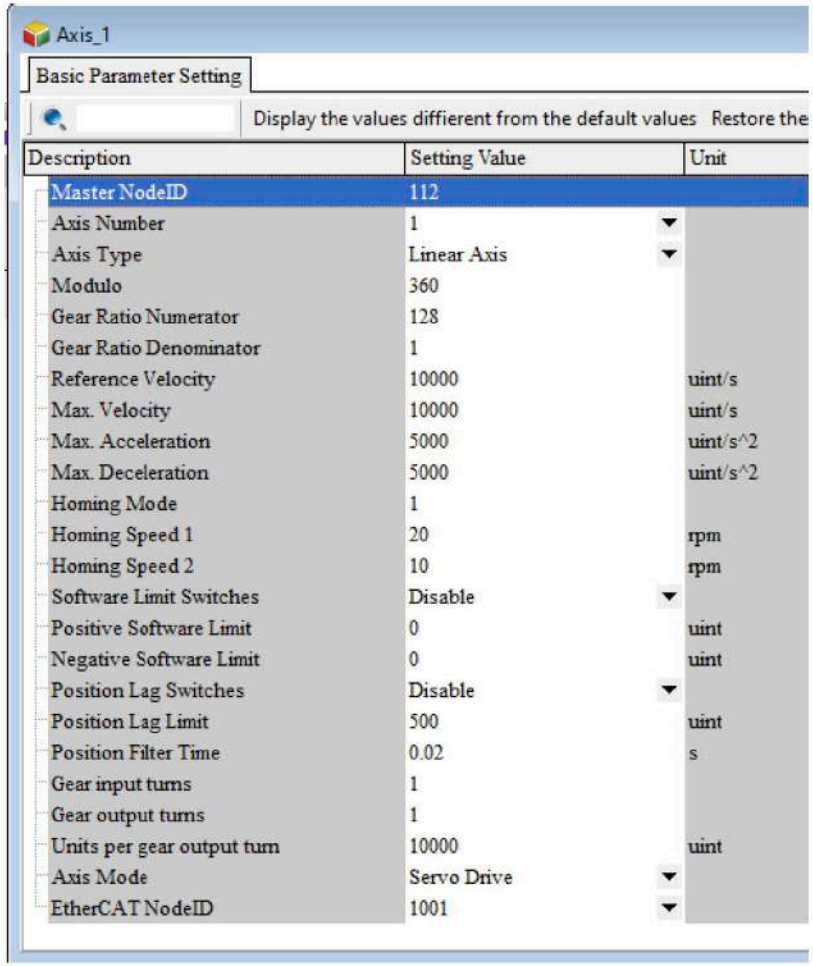
Список функций, созданных прикладным разработчиком. В программе вызываются как код, без создания экземпляров

Таблицы с группировкой переменных для мониторинга в онлайне

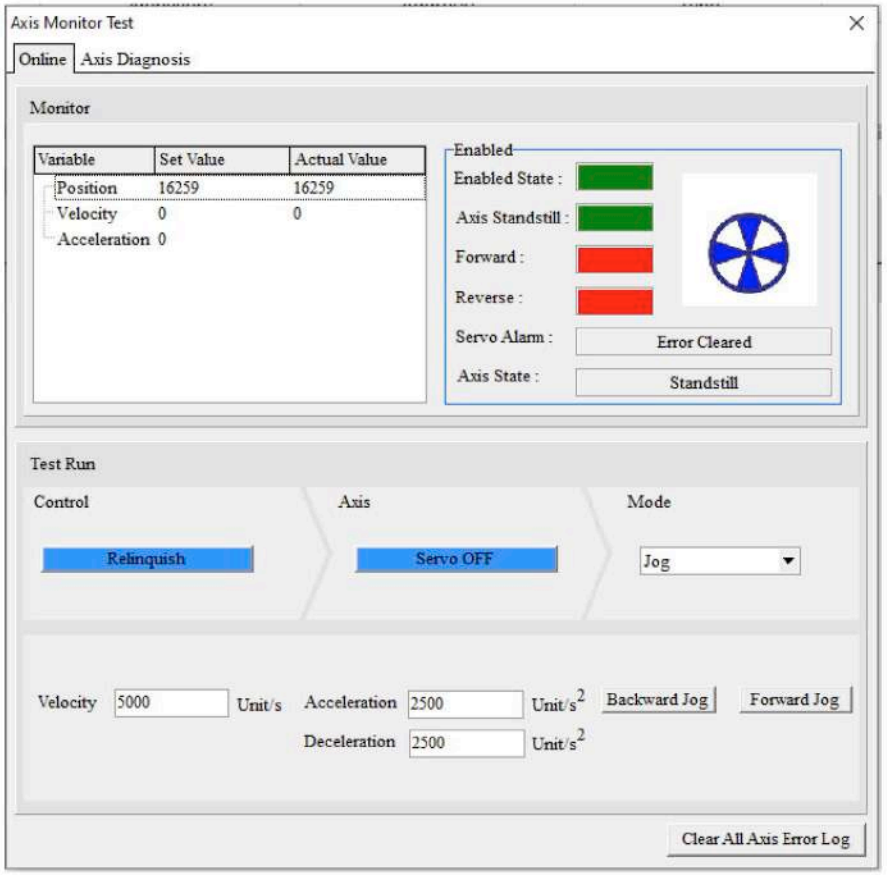
Список библиотечных функций. В программе вызываются как код, без создания экземпляров

Настройка осей

Параметры каждой оси задаются быстрым и удобным способом в отдельной оконной форме

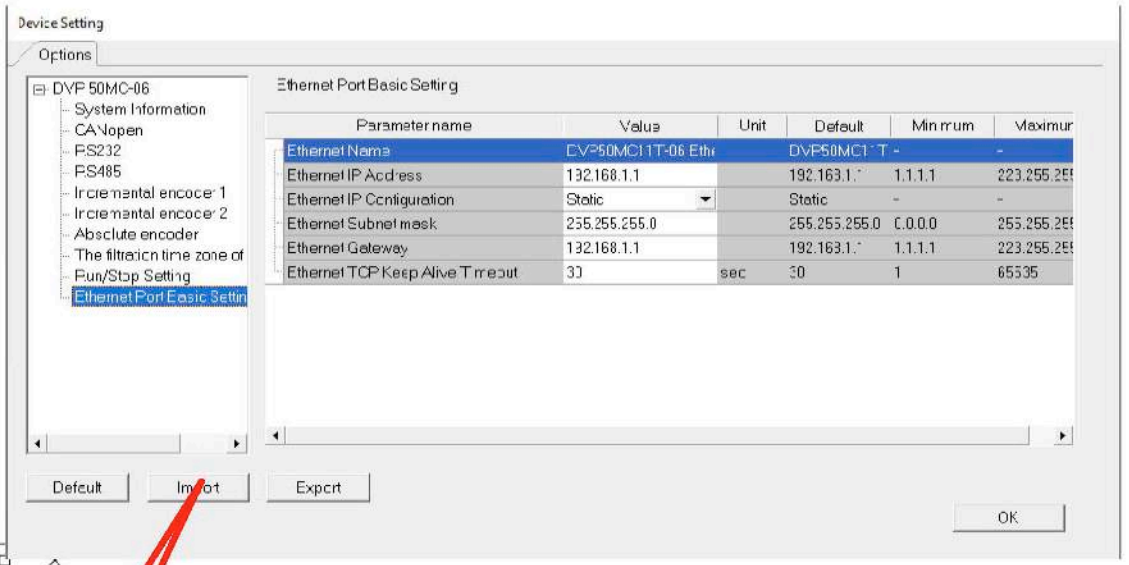


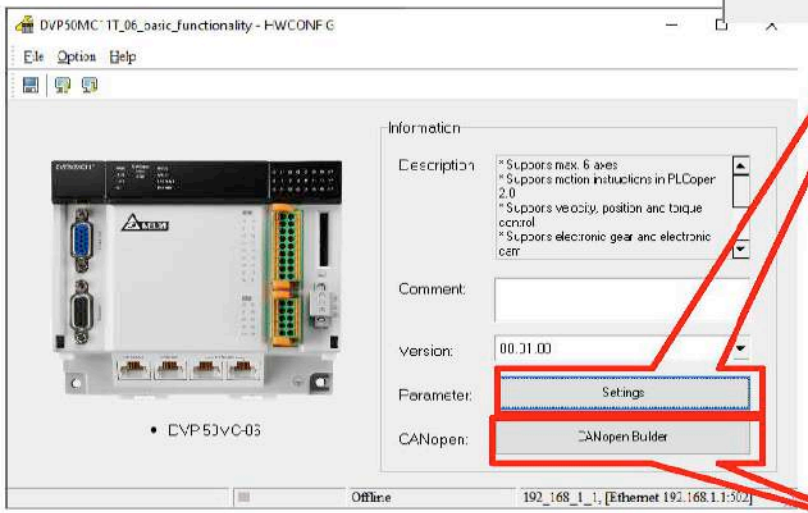
При помощи инструмента Monitor & Test Run можно не создавая программы проверить работоспособность каждой оси



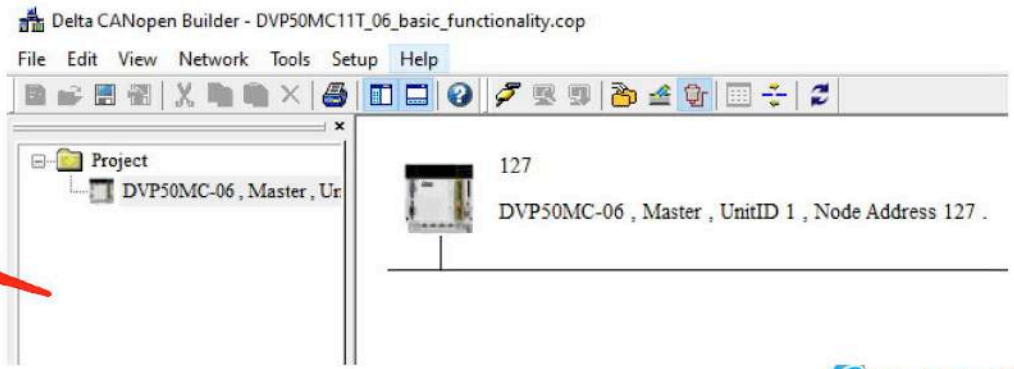
Конфигуратор аппаратной части HWCONFIG

Окно настройки аппаратных ресурсов: портов, энкодеров и т.д.



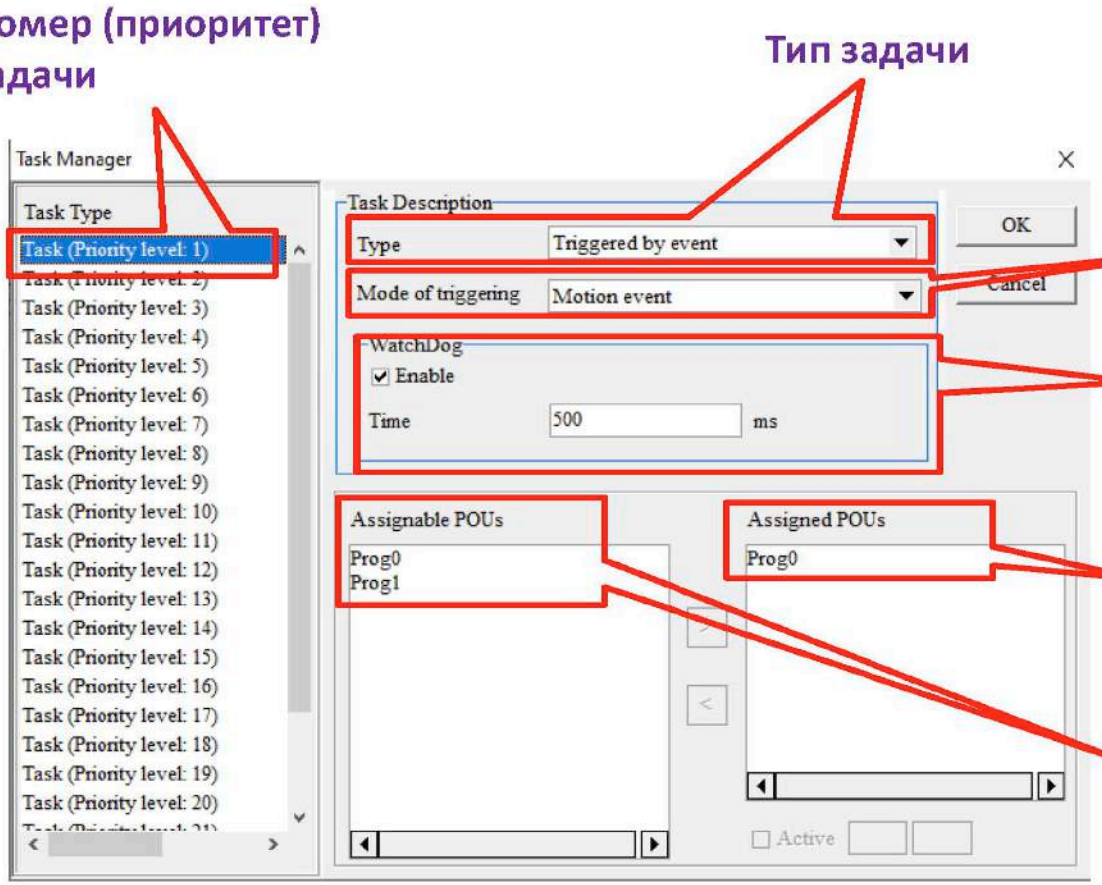


Конфигуратор настройки сети CANopen DS301



Менеджер задач Task Manager

Созданные программные единицы (POU) для своего исполнения должны быть привязаны к той или иной задаче (Task). Тип и номер задачи определяет приоритетность её исполнения. Программные единицы исполняются внутри задачи согласно их положению в списке: сверху вниз.



Общий порядок исполнения задач

Типы задач

- Cyclic task ： : исполняется циклически с заданным интервалом
- Freewheeling task : исполняется свободно, по мере освобождения процессора
- Task triggered by event: исполняется при наступлении заданного события

Номер задачи

Номер задачи определяет приоритет её исполнения. Необходимо учитывать, что если задачам с более высоким приоритетом будет выделено слишком мало времени для исполнения, то это приведёт к тому, что задачи с меньшим приоритетом не будут исполняться вообще

Сторожевой таймер

Если какая-то задача превышает выделенное время для исполнения, то сработает сторожевой таймер и контроллер перейдёт в состояние ошибки. Также, срабатывание сторожевого таймера может произойти при неправильном распределении времени исполнения между задачами с высоким и низким приоритетом. При слишком маленьком интервале исполнения высокоприоритетная задача будет занимать постоянно время процессора, что также вызовет срабатывание сторожевого таймера

Признаки вызова задачи по событию

- Motion_CYC – синхроимпульс шины управления движением
- CANopen_SYNC ： синхроимпульс CAN шины (DS301)
- Входы на ЦПУ (I0~I7 , I10~I17) - передний или задний фронт
- ABZ1_Z(Rising) ： Z импульс на энкодере 1(передний фронт)
- ABZ2_Z(Rising) ： Z импульс на энкодере 2 (передний фронт)

Распределение внутренней памяти

Место	Тип	Объём (ед.: БАЙТ)
Система (регистры с доступом по адресу)	Регистры Входов (%I)	128
	Регистры Выходов (%Q)	128
	Регистры Данных (%M)	128K
Разработчик	Энергозависимые переменные	20M
	Энергонезависимые переменные	128K
	Программа	20M
	Таблицы E-cam	8M
	G-коды (64 программы по 256K)	8M

Типы данных

Тип	Диапазон	Начальное значение	Длина в байтах
BOOL	TRUE or FALSE	FALSE	1
BYTE	16#00 ~ FF	16#00	1
WORD	16#0000 ~ FFFF	16#0000	2
DWORD	16#00000000 ~ FFFFFFFF	16#00000000	4
LWORD	16#0000000000000000 ~ FFFFFFFFFFFFFFFF	16#0000000000000000	8
USINT	0 ~ +255	0	1
UINT	0 ~ +65535	0	2
UDINT	0 ~ +4294967295	0	4
ULINT	0 ~ +18446744073709551615	0	8
SINT	−128 ~ +127	0	1
INT	−32768 ~ +32767	0	2
DINT	−2147483648 ~ +2147483647	0	4
LINT	−9223372036854775808 ~ +9223372036854775807	0	8
REAL	−3.402823e+38 ~ −1.175495e-38 , 0 , +1.175495e-38 ~ +3.402823e+38	0.0	4
LREAL	−1.79769313486231e+308 ~ −2.22507385850721e-308,0 , +2.22507385850721e−308 ~+1.79769313486231e+308,	0.0	8
TIME	T#XXXXXXdXXhXXmXXsXXX.XXXms, ед.: нс Диапазон: T # 0 мс ~ 213503 дней 23 часа 34 мин. 33 сек. 709.551 мс	T#0ms	8
DATE	D#Y-M-D. Диапазон: D#1970-01-01~D#2106-02-07. Ед.:.сек.	D#1970-01-01	4
TOD	TOD#H：M：S.MS. Диапазон：TOD#00:00:00~23:59:59:999. Ед.:мс Значение=0, TOD#00:00:00, значение=1, TOD#00:00:00.001, значение=86399999, TOD#23:59:59:999 , значение=86400000,TOD#00:00:00, значение = 4294967295, TOD#17:2:47:295.	TOD#00:00:00	4
DT	DT#Y-M-D-H-M-S. Диапазон : DT#1970-01-01-0:0:0~2106-02-07-6:28:15. Ед.: сек.	DT#1970-01-01-0:0:0	4

Система обозначений регистров по типам данных

В контроллерах DVP-MC используется следующая система обозначений физических регистров по типам и длине данных:

① : Фиксированный символ -- % (ставится обязательно)
② : Тип регистра данных -- I (Входы), Q (Выходы), M (Регистры данных)
③ : Длина данных -- X (бит)、B (байт)、W (2 байта), D (4 байта), L (8 байтов)
④ : Номер регистра поле 1 – первая цифра ставится всегда и обозначает номер регистра в зависимости от типа, определённого в ② и ③. Если ③ является битовым регистром типа X, то в данном случае данное поле обозначает номер байта
⑤ : Номер регистра поле 2 – фиксировано как «точка». Используется только в битовых регистрах типа X
⑥ : Номер регистра поле 3 -- Используется только в битовых регистрах типа X и обозначает номер бита в байте, определённом в ④

Группировка байтов в регистры по типам

В контроллерах DVP-MC реализована концепция распределение общего объёма памяти под требуемый набор регистров. Т.е. память представлена как непрерывная и общая для всех типов регистров последовательность байтов. При использовании регистра того или иного типа забирается соответствующее количество байтов в этом общем объёме памяти

Тип	Количество занимаемых байтов							
%MX	%MX0.0~0.7	%MX1.0~1.7	%MX2.0~2.7	%MX3.0~3.7	%MX4.0~4.7	%MX5.0~5.7	%MX6.0~6.7	%MX7.0~7.7
%MB	%MB0	%MB1	%MB2	%MB3	%MB4	%MB5	%MB6	%MB7
%MW	%MW0		%MW1		%MW2		%MW3	
%MD	%MD0				%MD1			
%ML	%ML0							

Device Name	Value (8bits)	Value (16bits)	Value (32bits)	Value (64bits)
%MB0	129	4737	1912279681	1234567893633
%MB1	18	64274	527563538	4822530834
%MW0	129	4737	1912279681	1234567893633
%MD0	129	4737	1912279681	1234567893633
%ML0	129	4737	1912279681	1234567893633

Например, при использовании регистра %ML0 он займёт 8 байтов, и все регистры меньшего размера, входящие по номерам байтов в его состав, также будут меняться при изменении %ML0

Диапазон регистров %MW0~%MW999 является энергонезависимым. Также при объявлении переменных без адреса можно поставить класс VAR_RETAIN, чтобы сделать переменную энергонезависимой

Local Symbols			
Class	Identifiers	Address	Type
VAR_RETAIN	word1	N/A [Auto]	WORD
VAR	bool2	N/A [Auto]	BOOL

Диапазоны номеров регистров по типам

Байты	Тип	Обозначение	Диапазон
бит	I (Input)	%IX0.0~%IX0.7	%IX0.0~%IX127.7
	Q (Output)	%QX0.0~%QX0.7	%QX0.0~%QX127.7
	M (Register)	%MX0.0	%MX0.0~%MX131071.7
байт	I (Input)	%IB0	%IB0~%IB127
	Q (Output)	%QB0	%QB0~%QB127
	M (Register)	%MB0	%MB0~%MB131071
2 байта	I (Input)	%IW0	%IW0~%IW63
	Q (Output)	%QW0	%QW0~%QW63
	M (Register)	%MW0	%MW0~%MW65535
4 байта	I (Input)	%ID0	%ID0~%ID31
	Q (Output)	%QD0	%QD0~%QD31
	M (Register)	%MD0	%MD0~%MD32767
8 байтов	I (Input)	%IL0	%IL0~%IL15
	Q (Output)	%QL0	%QL0~%QL15
	M (Register)	%ML0	%ML0~%ML16383

Адреса для протокола Modbus

Функция	Тип	Диапазон	адрес Modbus	тип адреса
I (Input)	Bit	%IX0.0~%IX0.7	0x6000~0x6007	Стандартный адрес Modbus
		%IX1.0~%IX1.7	0x6008~0x600F	
		.....	.....	
		%IX127.0~%IX127.7	0x63F8~0x63FF	
	Word	%IW0~%IW63	0x8000~0x803F	
Q (Output)	Bit	%QX0.0~%QX0.7	0xA000~0xA007	
		%QX1.0~%QX1.7	0xA008~0xA00F	
		.....	.....	
		%QX127.0~%QX127.7	0xA3F8~0xA3FF	
	Word	%QW0~%QW63	0xA000~0xA03F	
M (Register)	Bit	%MX0.0~%MX0.7	0x10000000~0x10000007	* Delta Extended Modbus Address
		%MX1.0~%MX1.7	0x10000008~0x1000000F	
		.....	.....	
		%MX131071.0~%MX131071.7	0x100FFFF8~0x100FFFFF	
	Word	%MW0~%MW32767	0x0000~0x7FFF	Стандартный адрес Modbus
	Word	%MW32768~%MW65535	0x20008000~0x2000FFFF	* Delta Extended Modbus Address

Стандартные адреса Modbus могут использоваться для опроса регистров контроллера DVP-MC любым стандартным Modbus Мастером

*** Расширенные адреса используются в панелях оператора Delta DOP-100**

Агрегированные типы данных (DUT)

Контроллеры DVP-MC поддерживают создание пользовательских типов данных (DUT) – структур и объединений, которые формируют экземпляры и могут объявляться как переменные соответствующего типа, а также перечислений

```
0001 TYPE Struct_1 :
0002 STRUCT
0003   b1 : BOOL;
0004   b2 : BOOL;
0005   b3 : BOOL;
0006   int1 : INT;
0007   int2 : INT;
0008   int3 : INT;
0009   Ar1_INT : ARRAY [0..2, 0..5] OF INT;
0010
0011 END_STRUCT
0012 END_TYPE
```

```
0002 TYPE Union_W_DW :
0003 UNION
0004   DW : DINT;
0005   AR_W :  ARRAY [0..1] OF INT;
0006 END_UNION
0007 END_TYPE
```

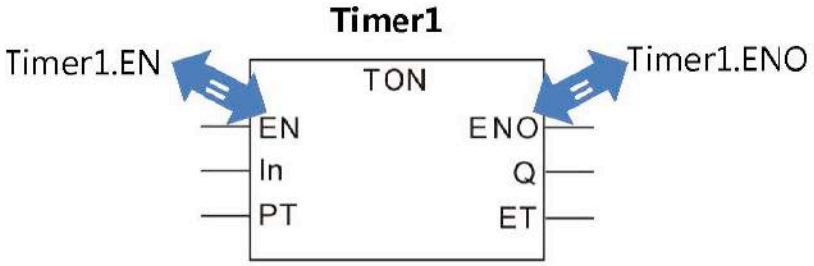
```
0001 TYPE VFD_1 :
0002 (
0003   _STOP := 0,
0004   _RUN_FWD := 3,
0005   _RUN_REV := 15
0006 );
0007 END_TYPE
```

Обращение к внутренним переменным ФБ

После создания экземпляра функционального блока (переменной типа ФБ), к его параметрам можно обращаться напрямую через точку:

Имя экземпляра ФБ. Имя формального параметра

Таким образом, аргументы можно передавать путём вызова всего экземпляра в программе, или путём обращения по отдельности к каждому параметру. Например: к параметрам экземпляра Timer1 ФБ TON можно обратиться двумя способами:



Прямая передача аргументов

```
Timer1 ( EN:= variable1 ,
         IN:= variable2 ,
         PT:= T#1S,
        ENO =>variable4,
        Q =>variable5,
        ET=>variable6 );
```

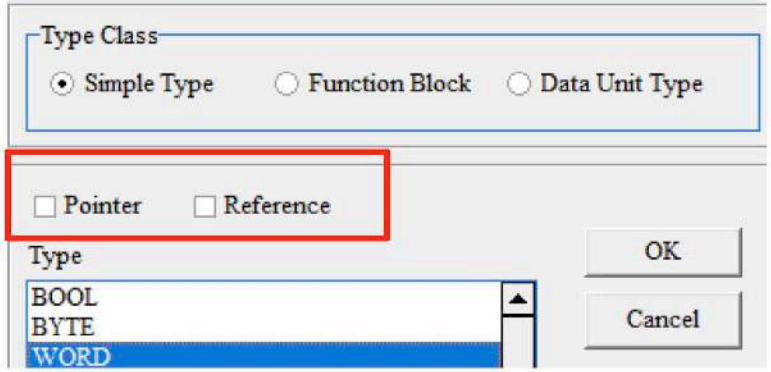
ИЛИ

Косвенная передача аргументов

```
Timer1 .EN:=  variable1; // Передача входного аргумента variable1 параметру EN
Timer1 . IN:=  variable2; //Передача входного аргумента variable2 параметру IN
Timer1 . PT:= T#1S; // Передача входного аргумента T#1S параметру PT
Timer1 ( );        //Вызов экземпляра Timer1 ФБ TON
variable 4 := Timer1 .ENO;//Выдача параметра ENO в выходной аргумент variable4
variable5  := Timer1 .Q ;  //Выдача параметра Q в выходной аргумент variable5
variable6  := Timer1 .ET; //Выдача параметра ET в выходной аргумент variable6
```

Язык ST совместно с языком LD

Начиная с версии среды программирования ISPSoft 3.10 можно использовать выражения, записанные на языке ST, в программных единицах, где программа пишется на языке LD, в качестве входных аргументов инструкций



Также, можно объявлять переменные как ссылки и указатели

СПАСИБО
ЗА ВНИМАНИЕ!

Компания «СТОИК»

+7(495) 661-24-41 / 661-24-61

www.deltronics.ru

г. Москва, ул. Семёновский Вал, д. 6А