NCT2xx
NCT3xx

Программирование PLC

От версии программного обеспечения № 14.9

Производитель и разработчик: **NCT Ipari Elektronikai kft**.
H1148 Budapest Fogarasi út 7
⊠ Почтовый адрес: 1631 Bp. pf.26
☎ Телефон: (+36 1) 467 63 00
☎ Телефакс:(+36 1) 467 63 09
Электронная почта: nct@nct.hu
Вебсайт: www.nct.hu

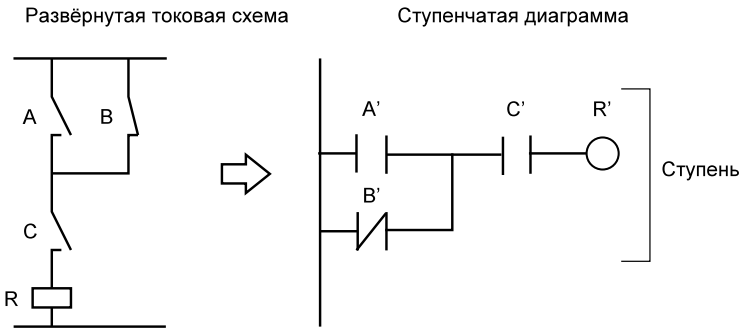
Содержание

17 Декабрь 2019 г.

1 Язык программы PLC

Язык программы PLC имеет формат **ступенчатой диаграммы**.

Ступенчатая диаграмма является упрощённой формой релейной **развёрнутой токовой схемы,** применённой в технике управления.



Приложенный рисунок показывает пример для изображения логической связи между развёрнутой токовой схемой и ступенчатой диаграммой: A и C замыкающий контакт, B нормально закрытий контакт, R реле.

Провода (логические линии) исходят на ступенчатой диаграмме из расположенной с левой стороны т.н. провода референции. За ними следуют контакты. Они могут быть управляемыми вводами / выводами, или замыкающими а также размыкающими контактами, принадлежащими к внутренним вспомогательным реле, к задерживающими реле, к реле времени. В правом конце логической линии находятся “обмотки” выводов, реле времени, счётчиков и т.д., или команды.

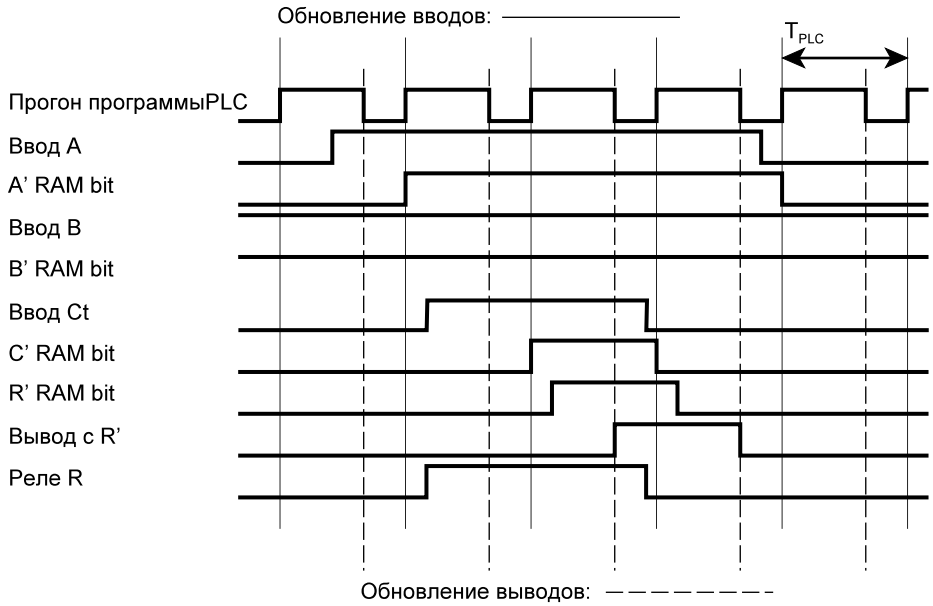
Ступенью (по-английски rang) называем контакты и провода, принадлежащие к одному выводу.

Важное правило, что в ступенчатой диаграмме и принадлежащей к ней программе данные выводы, реле, реле времени, счётчики и т.д. могут фигурироваться только один раз. Однако их рабочие контакты можно использовать в программе неограниченный раз.

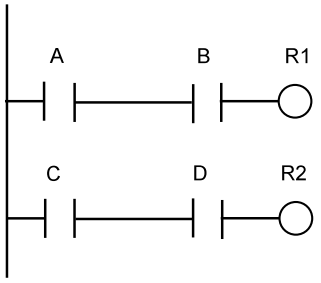
Между работой логической электрической цепи с аппаратным скелетом и ступенчатой программы PLC, осуществлённой программным обеспечением, имеются существенныи разницы.

1.1 Выборка проб, обращение с вводами и выводами

Программа PLC через определённые промежутки времени T_{PLC} совершает прогон, где $\mathbf{T_{PLC}}$- - это **время выборки проб**.
Остаёмся у примера предыдущего рисунка. Прежде чем программа PLC приступила бы к выполению команд, **берёт пробу** из сигналов A, B и C (из вводов PLC) и сохраняет их в память по адресу A', B' и C'.
След за этим начинается **прогон программы** PLC, которая подсчитает значение R' из значений A', B', C'. В ходе выполнения команды, значение R' сохраняется в памяти по адресу R'.
После полного прогона программы PLC, **выводы обновляются** из памяти, то есть значение R' в этот момент выводится из RAM на вывод.
По сравнению с этим, электрическая цепь, заложенная в аппаратном скелете, сразу реагирует на изменения. Это означает, что сигналы контактов A, B, или C готовы сразу активировать реле R.

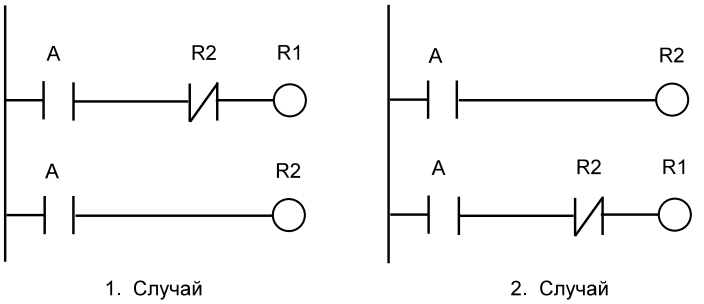


1.2 Порядок выполнения программы PLC



Программа PLC со ступенчатой диаграммой выполняется также, как любая компьютерная программа, со строчки в строчку, вернее со ступени на ступень.
На приложенном рисунке PLC сначала высчитает значение реле R1 в зависимости от состояния контактов A и B (1-я ступень), затем значение реле R2 - в зависимости от контактов C и D. Выполнение происходит всегда по порядку описания.
В полосе реле с аппаратным скелетом нет такого порядка

выполнения, каждое реле работает примерно в одно и то же мгновение времени.
Это особенность влечёт за собой то, что ступенчатая диаграмма не соответствует во всех случаях один к одному очерченному в полосе реле. Возьмём следующий пример.



– **В случае полосы реле**: Если оба случая, видного на рисунке, осуществить полосом реле, оба случая работают одинакого: если включается контакт “A”, и реле R1 и R2 срабатывает на секунду, потом после срабатывания R2 сработавшее R1 отпадает.
– **В случае ступенчатой диаграммы**: В 1-ом случае, если контакт “A” включается, включается R1, так как R2 не сработавшее. Потом включается и реле R2. В следующем цикле PLC, позже на время T_{PLC}, отпадает реле R1, так как реле R2 уже сработавшее. Значит, за время T_{PLC} и R1 и R2 сработавшие.
В 2-ом случае, если контакт “A” включается, вследствие этого сработает реле R2, поэтому в следующем строчке реле R1 уже даже не включается, так как R2 сработавшее.

Из примера выше видно, что решения с аппаратным скелетом можно переносить в программу PLC со ступенчатой диаграммой только после обдумывания.

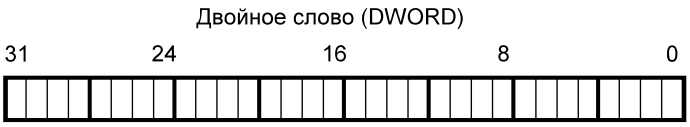
В программе PLC имеется возможность обусловливать изменение порядка выполнения: скачком, вызовом продпрограммы.

1.3 Редактирование программы PLC

Программу PLC, имеющую на основе ступенчатую диаграмму, надо редактировать графически, поэтому для составления программы требуется программное обеспечение, разработанное специально для этой цели. Описание редактора программы PLC не входит в тему настоящей книги.

2 Память, использованная программой PLC

Память, использованная программой PLC занимает связный объём с длиной 10000 двойного слова. Под двойное слово (DWORD) понимается единица памяти на 32 бит.



Объём памяти PLC делится на 4 главных частей:

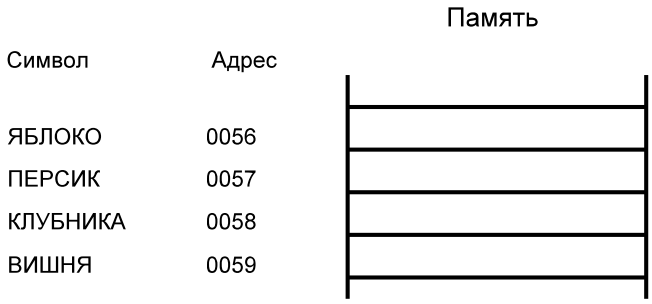
– на двойные слова с символическим адресом FLAGS, содержащие по адресу 0000 статусбиты, установленные командой программы PLC,
– на объёмы памяти, выполняющие коммуникацию между программой PLC и аппаратными блоками ввода и вывода, а также между программой PLC и системой,
– на рабочие объёмы, содержащие внутренние переменные программы PLC, сохраняющиеся и после выключения, начинающиеся со символического адреса PLCNVRAM, и
– на рабочие объёмы, содержащие внутренние переменные программы PLC, не сохраняющиеся после выключения, начинающиеся со символического адреса PLCRAM.

Границы адресов отдельных объёмов могут меняться по типам.

Програмист PLC должен все переменные команд занести в этот объём памяти. Например, если требуется таймер в программе PLC, переменное таймера, измеряющее течение времени, нужно декларировать в этой памяти.
Значение статусных битов рассматривается в отдельном разделе этой книги.
Разпределение адресов и их значение для битов и регистров по объёму памяти, выполняющих коммуникацию между программой PLC и блоками ввода/вывода, а также между программой PLC и системой, изложены в другой книге, относящейся к данному типу.
Основной единицей памяти является двойное слово (DWORD, 32 битов). Для адресовки меньшей этой единицы, как слова (WORD, 16 битов) или байт (8 битов), нет возможности. Однако, любая часть памяти достигаемая по битам.
Программа PLC обеспечивает широкую возможность к символическому доступу к памяти. Любая часть объёма памяти, достигаемая адресовкой, достигается и символично.

2.1 Адресовка двойного слова (DWORD)

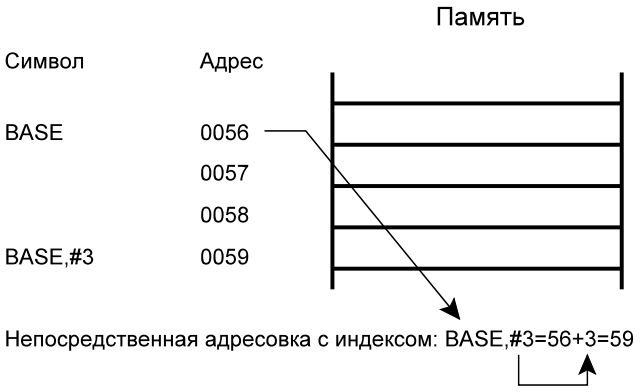
Можно ссылаться на любое двойное слово памяти
числом и
а также **символично**.
В случае ссылки числом, всегда надо задавать 4 десятичных чисел, записав **ведущие нули**.
В случае ссылки символично, символ предварительно нужно декларировать в магазине символов.



На основании вышего примера можно ссылаться на 56-ю ячейку памяти
0056, или записыванием
ЯБЛОКО

2.2 Индексированная адресовка двойного слова (DWORD) оператором “,”

Индексированная адресовка состоит из двух частей:
из **базового адреса** и
из **смещения**.
Эти две части разделены друг от друга оператором
, (запятая).
Адрес формируется таким образом, что к базовому адресу добавляется значение смещения:
адрес=базовый адрес+смещение
Для **базового адреса** действительны правила, относящие к адресовке двойного слова:
можно задать на 4-х десятичных числах, или
символично.
Значение смещения можно задать
непосредственно числом, или постоянным символом, или
косвенной ссылкой на регистр.
– В случае задачи **непосредственного** смещения нужно пользоваться десятичным оператором для ввода чисел **“#”**. При желании сместить адрес на 3 по отношению базовому адресу BASE, надо записать ссылку
0056,#3, или
BASE,#3,или,
предпологая, что символ BASE декларирован к адресу 0056.



Тот же результат получим, если в таблице символов декларировать символ BIAS постоянным символом: BIAS #3. Тогда в программе надо записать ссылку

BASE,BIAS.

– В случае задачи **косвенного**, индиректного смещения через регистр, после оператора "," (запятая)надо записать адрес того регистра, который содержит значение смещения.

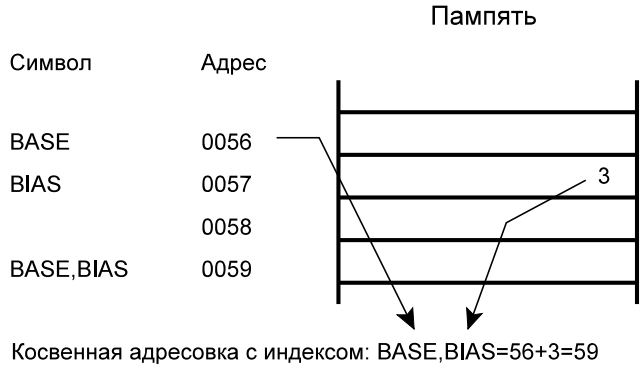
К адресу регистра, содержащего смещения, действительны нормальные правила адресовки: их можно задавать и числом и символом.

Если к базовому адресу BASE декларировать регистра BIAS, адрес можно задвать ссылкой:

BASE,BIAS или

0056,0057

при условии, что символ BASE декларировать к адресу 0056, а символ BIAS к адресу 0057. Значение смещение берётся из регистра по адресу BIAS (0057).



2.3 Непосредственная адресовка бита и оператор “.”

К любому биту памяти PLC можно адресовать. На биты можно ссылаться с использованием оператора “.” (точка), а также символом, декларированным на данный бит.
Если применить **оператор “.”**, адресовка с битом состоит из двух частей:

из **адреса двойного слова** и
из **адреса бита** внутри двойного слова

Эти две части разделены друг от друга оператором

. (точка).

На адрес двойного слова можно ссылаться 4-мя десятичными числами, а также символично.
После оператора “.” на адрес бита можно ссылаться только

со значением.

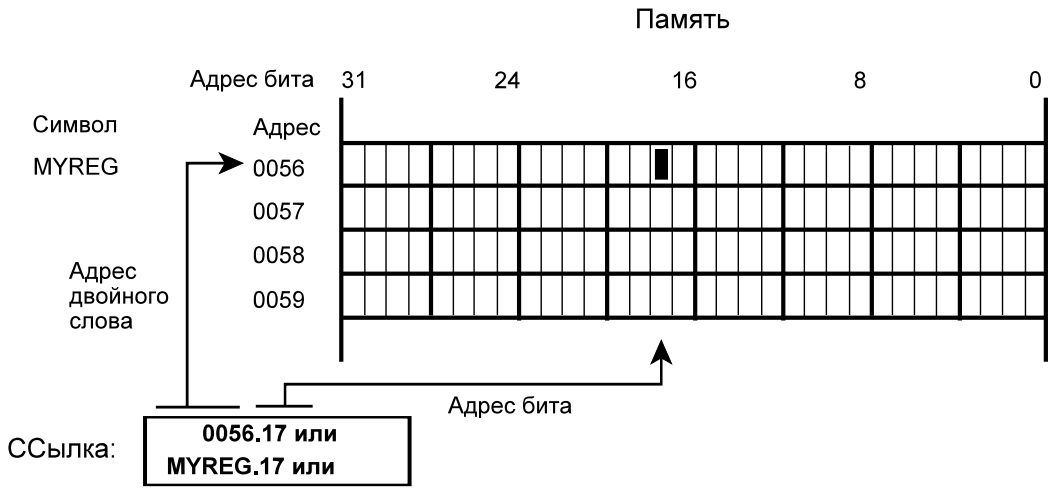
После “.” надо задавать всегда 2 цифра, то есть надо написать и ведущие нули. Область значения адреса бита:

00 ... 31

Если желаем ссылаться например на 17-й бит двойного слова с адресом 0056, тогда можно использовать ссылку

0056.17 или
MYREG.17,

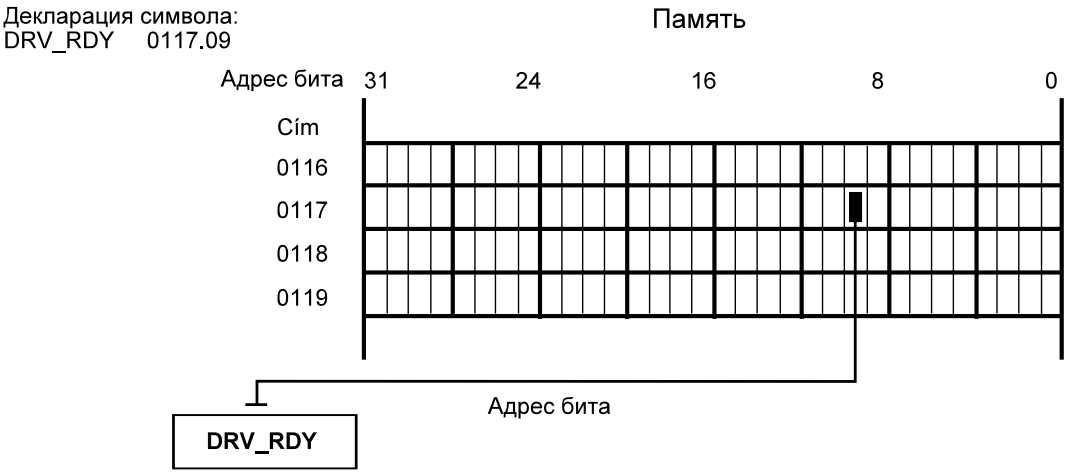
при условии, что MYREG декларировали предварительно на адрес 0056.



Если на адрес бита декларировать символ, **символом** можно ссылаться непосредственно на бит. Если например символ DRV_RDY декларировали на адрес бита 0117.09, тогда символом

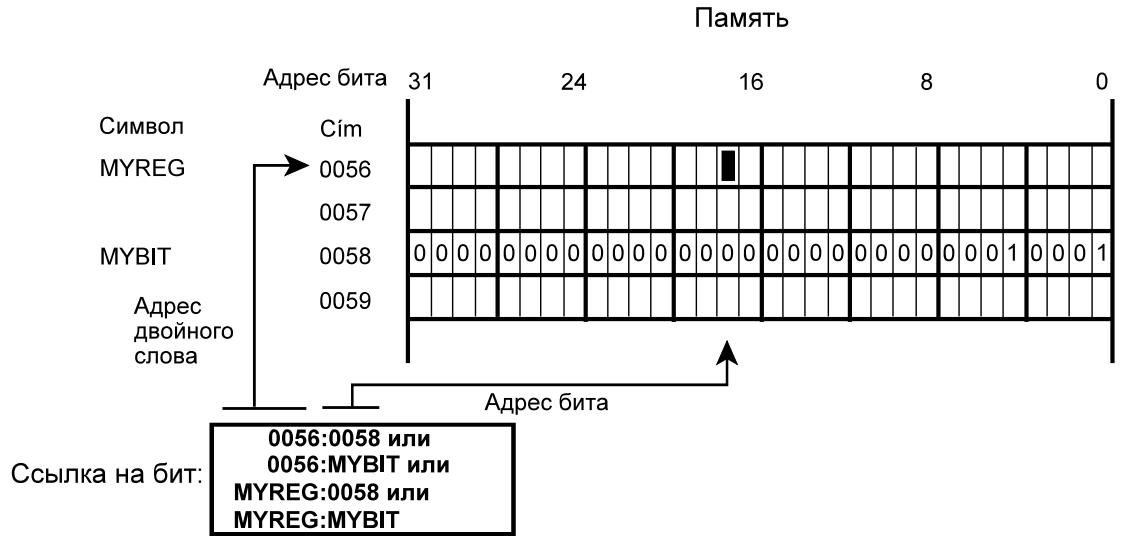
DRV_RDY

можно достичь непосредственно бит 0117.09.



2.4 Косвенная адресовка бита оператором “:”

Если использовать **оператор “:”**, адресовка с битом состоит из двух частей:
из **адреса двойного слова** и
из **адреса бита** внутри двойного слова
Эти две части разделены друг от друга оператором
: (двоеточие).
На адрес двойного слова можно ссылаться
4-мя десятичными числами,
а также символично.
На адрес бита после оператора “:” можно ссылаться
адресом регистра, заданным 4-мя десятичными числами,
или символичным адресом, указывающим на регистр.
Например, ссылка
0056:0058
относится к тому по порядкому номеру биту регистра 0056, сколько содержание регистра 0058.
Если декларировать
символ MYREG на адрес 0056, а
символ MYBIT на адрес 0058,
можно и так писать, что
MYREG:MYBIT.



2.5 Индексированная адресовка бита оператором “,”

Индексированная адресовка бита подобно к адресовке двойного слова. Адресовка состоит из двух частей:

из **базового адреса бита** (**база.бит**) и

из **смещения**.

Базовый адрес бита относится уже к конкретному биту (бит) двойного слова (база). Значение смещения смещает всегда адрес двойных слов.

Две части разделяется друг от друга оператором

, (запятая).

Адрес формируется так, что к базовому члену базового адреса бита прибавится значение смещения:

адрес бита=(база+смещение).бит

К базовому адресу бита действительны правила, относящиеся к адресовке бита:

символично

можно задавать оператором точки “.”, или

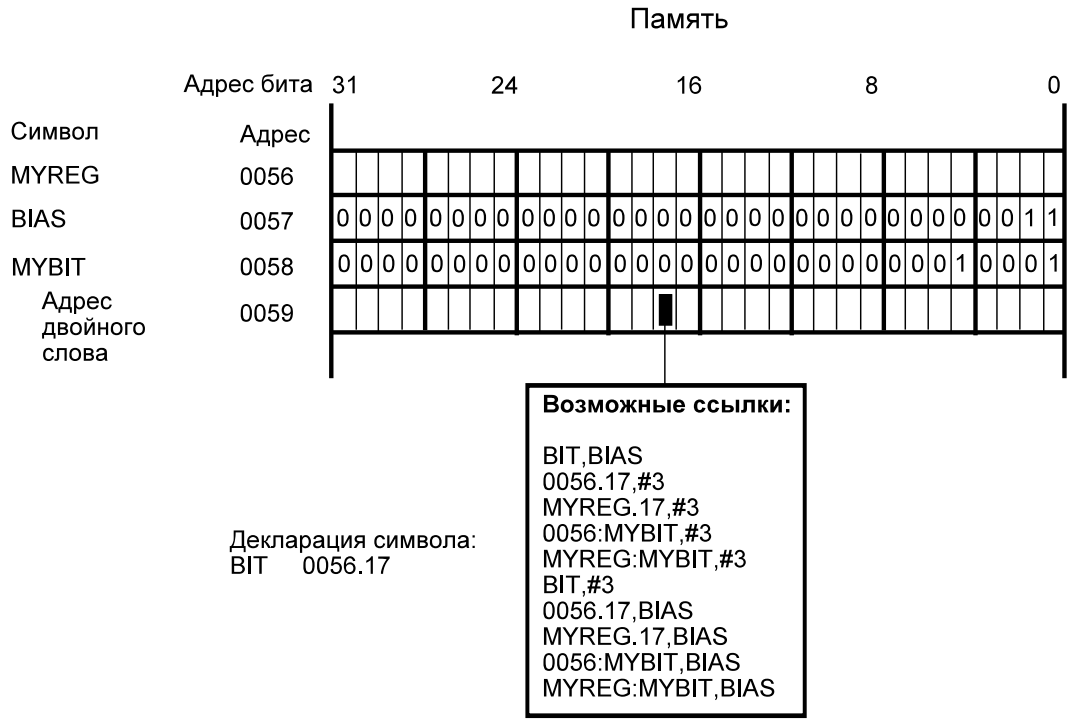
косвенно оператором “:”.

Значение смещения можно задавать

непосредственно числом, или

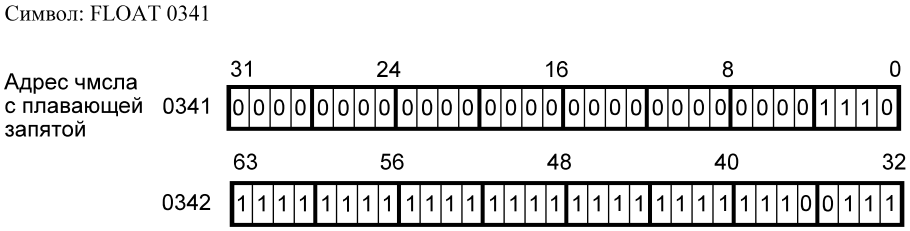
косвенно ссылкой на регистр.

При желании переместить бит с адресом 0056.17 на 3,то это можно поступить таким образом, как это видно на рисунке:



2.6 Адресовка чисел с плавающей запятой (double)

Команды программы PLC обращаются числами с плавающей запятой (double) на 64 битах. Числа с плавающей запятой занимают два последующего друг за другом регистра (DWORD) объёма памяти.



На число с плавающей запятой можно ссылаться

числом и также

символично.

В случае ссылки числом надо задавать 4 десятичных цифер, всегда записав **ведущие нули**.
В случае ссылки символично необходимо символ предварительно декларировать в магазине символов.
При ссылке на объём памяти, содержащий число с плавающей запятой, необходимо задавать всегда адрес первого слова (это содержит нижние 32 бита числа с плавающей запятой).
В примере выше можно ссылаться на число с плавающей запятой следующим образом:

0341 или

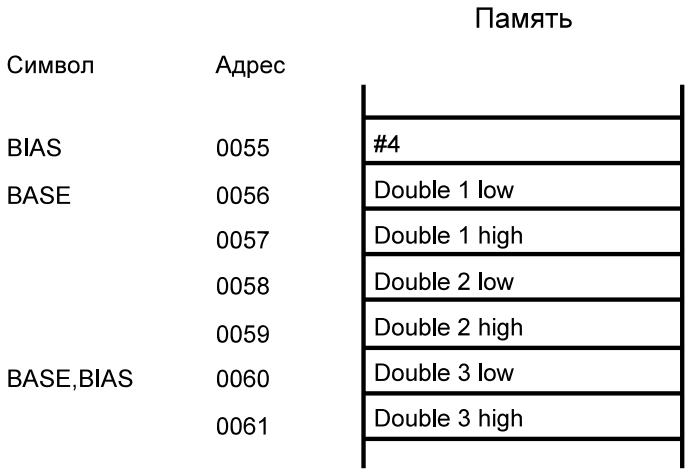
FLOAT.

2.7 Индексированная адресовка чисел с плавающей запятой (double) оператором “,”

Для индексированной адресовке чисел с плавающей запятой действительны правила, изложенные при индексированной адресовке регистров DWORD.

☞ **Внимание!**

Числа с плавающей запятой надо индексировать всегда чётными числами, так как они занимают два последующих друг за другом регистра!



Непосредственная адресовка с индексом: BASE,#4=56+4=60

Косвенная адресовка с индексом: BASE,BIAS=56+4=60

3 Модули программы PLC

Приограмма PLC состоит из **2-х модулей**:

из **главной программы** и

из модули **Int0**.

Оба модуля представляет собой независимую друг от друга ступенчатую сетку и выполняютсясамостоятельно. Установить связь между модулями возможно через память PLC. Разница между модулями имеется в частоте их прогона.

3.1 Главная программа

Главная программа выполняется согласно циклу времени T_{PLC}. Значение цикла времени зависит от его типа. Его значение видно на управлениях NCT201 встрочке PlcPeriod Диагностического окна в мсек-ах (например: 10мсек). В этом модуле описаны все деятельности PLC, кроме тех, которые требуют вмешательство, быстрее времени цикла T_{PLC}.

3.2 Модуль Int0

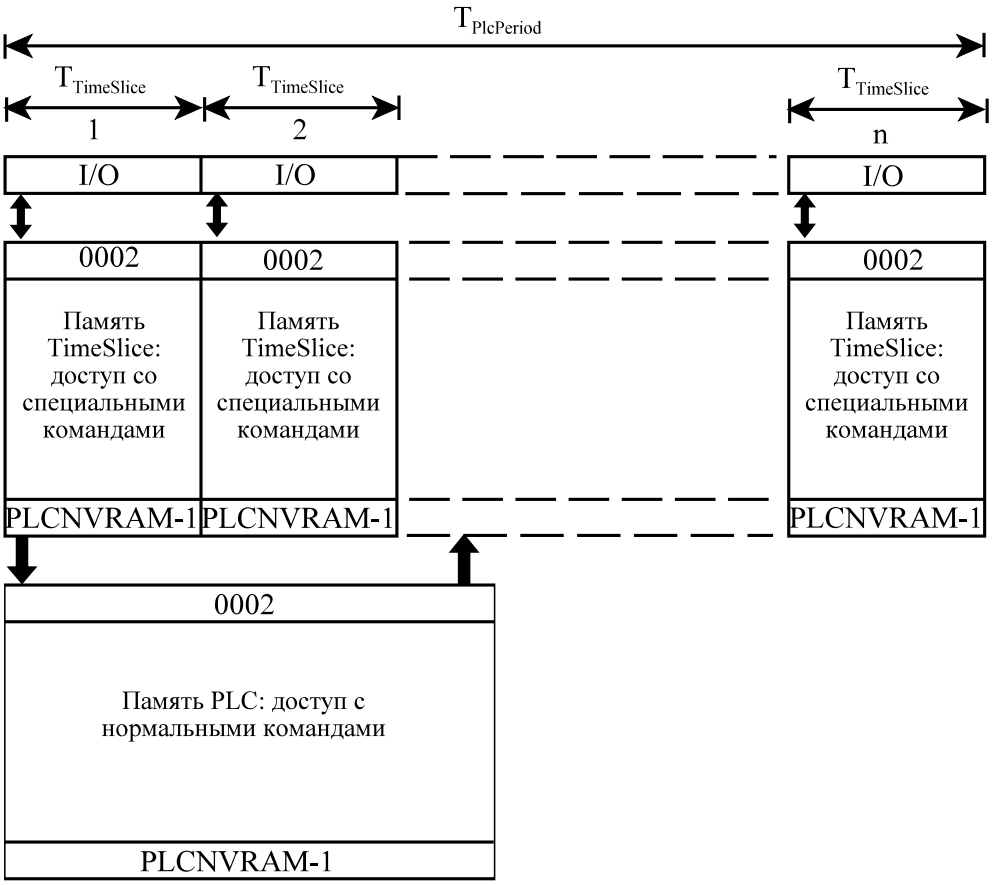
Модуль Int0 выполняется чаще T_{PLC} времени цикла. Его значение видно на управлениях NCT201 встрочке TimeSlice Диагностического окна µсек-ах (например: 2000 µсек). В этом модуле можно описать только те деятельности PLC, которые требуют быстрое вмешательство: например к сигналу ввода интерфейса быстро надо включить вывод интерфейса. Если слышком перегрузить модуль, получим сообщение свыше времени PLC.

3.3 Обновление памяти PLC

Вводы и выводы интерфейса, а также программа PLC **и объём памяти, выполняющий коммуникацию между системами,** зачитаются и выводятся согласно модулю **Int0**, то есть **согласно частоты времени TimeSlice**. Этот объём памяти длится от адреса 0002 до адреса PLCNVRAM-1.

Физические вводы зачитываются (например: сигналы ввода интерфейса) в RAM, а физические выводы (например. сигналы вывода интерфейса) выписываются из RAM на аппаратную программу. Программа PLC может выписывать или зачитать эту **память TimeSlice,** входящую в объём RAM, обновляемый частотой TimeSlice, только с помощью **специальных команд**. Этими командами имеет смысл пользоваться только в модуле Int0! **Объём памяти ввода PLC, достигаемый с** помощью нормальных команд PLC обновляется, то есть **синхронизируется перед пуском главной программы PLC** из памяти TimeSlice. Главная программа PLC может выполняться через несколько промежутков времени TimeSlice, таким образом в ходе прогона может работать из связанных друг с другом картин ввода. **Выводы**, соединенные с нормальными командами программы PLC, **обновляются после выполнения главной программы PLC** в памяти TimeSlice.

Нормальные команды обоих модулей, то есть модуля Главной программы и модуля Int0 видят тот же самый синхронизированный объём памяти.


T PlcPeriod
T TimeSlice
1
T TimeSlice
2
T TimeSlice
n
I/O
I/O
I/O
0002
0002
0002
Память TimeSlice: доступ со специальными командами
Память TimeSlice: доступ со специальными командами
Память TimeSlice: доступ со специальными командами
PLCNVRAM-1
PLCNVRAM-1
PLCNVRAM-1
0002
Память PLC: доступ с нормальными командами
PLCNVRAM-1

4 Данные, обработанные программой PLC

Программа PLC может обращаться данными с фиксированной запятой, с плавающей запятой и бинарными данными.
Данные с фиксированной запятой, сохранившиеся в памяти, могут быть и в бинарной форме и в форме BCD (бинарно кодированный, десятичный).
Бинарные данные можно запрашивать и для анализа изменения.

4.1 Обработка бинарных данных

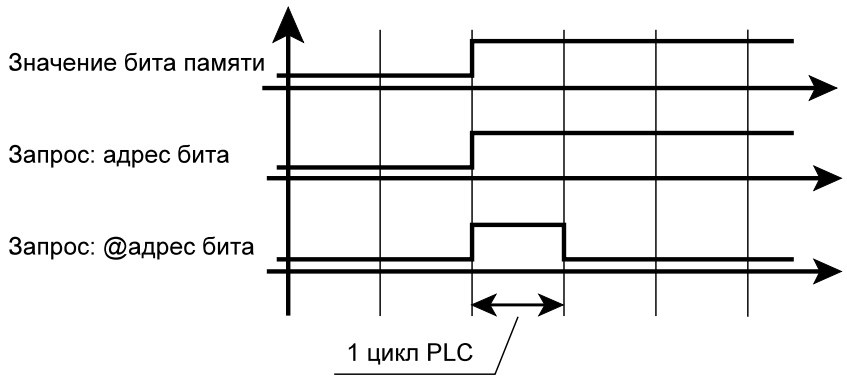
Чаще всего использованной единицей данных программы PLC являются бинарные данные. Можно запрашивать любой бит из всего объёма памяти PLC, то есть можно дефинировать контакт на адрес бита, и можно назначить любой бит вывода, то есть можно использовать например в качестве обмотки реле. Кроме того, для любого бита всего объёма памяти можно запустить анализ изменений.

4.2 Запрос набегающего фронта бита памяти оператором @

Изменение с 0 в 1 любого бита памяти, достигаемого с помощью программы PLC, можно запрашивать, если перед адресом бита памяти написать оператор @:

@ адрес бита: запрос набегающего фронта

Задачу адреса бита можно выполнить любым известным образом: непосредственно, или косвенно, символично, или индексированно.
Запрашиванные таким образом данные будут ВЕРНО в силу одного цикла PLC:

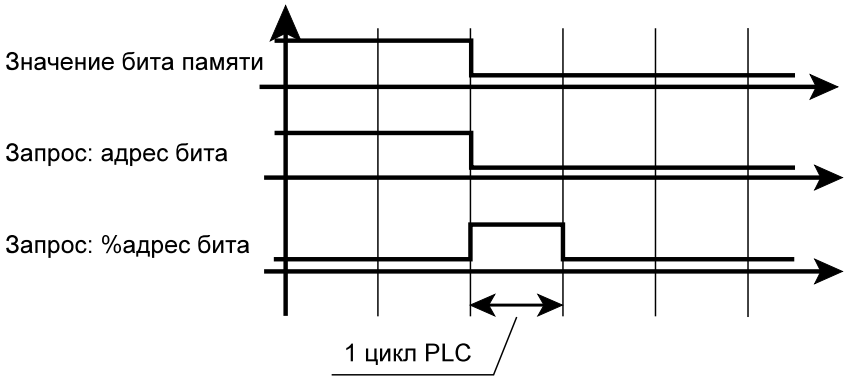


4.3 Запрос заднего фронта бита памяти оператором %

Изменение с 0 в 1 любого бита памяти, достигаемого с помощью программы PLC, можно запрашивать, если перед адресом бита памяти написать оператор %:

% адрес бита: запрос заднего фронта

Задачу адреса бита можно выполнить любым известным образом: непосредственно, или косвенно, символично, или индексированно.
Запрашиванные таким образом данные будут ВЕРНО в силу одного цикла PLC:



4.4 Немедленный запрос ввода, немедленная выдача вывода оператором "!"

Любой бит памяти, достигаемый с помощью программы PLC, можно немедленно запрашивать из памяти TimeSlice, а также можно немедленно записать в память TimeSlice, если перед адресом бита памяти написать оператор !:

! адрес бита: бинарное чтение, запись памяти TimeSlice

Задачу адреса бита можно выполнить любым известным образом: непосредственно, или косвенно, символично, или индексированно. Целесообразно его использовать в модуле Int0 для вмешательства, требующего быструю реакцию.

☞ ***Внимание!*** *Оператор ! Нельзя использовать вместе с оператором @, или %, образующим фронт!*

4.5 Задача десятичных чисел со знаком оператором "#"

Оператором # можно задавать десятичное постоянное со знаком в области

$-2147483648 \le$ постоянное ≤ 2147483647.

Редактор PLC сообщает об ошибке, если внесённое значение выходит за заданными пределами.

Внесённое число попадает в магазин **бинарно,** изображено **в комплементе 2**.

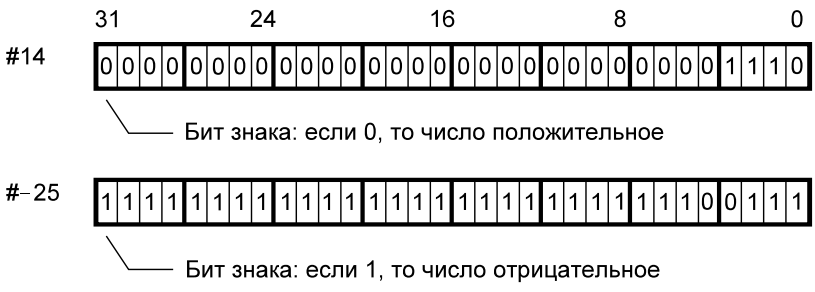
31-й бит является битом знака. Знак + (положительный) не нужно выставить.

Значит, при желании записать в магазин +14, то надо записать

#14,

если −25, тогда

#−25.



4.6 Задача гексадецимальных чисел оператором #$

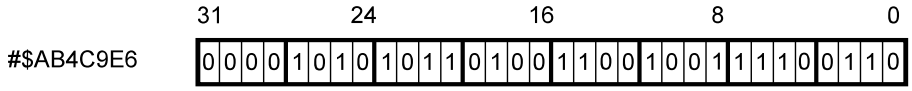
Оператором #$ можно задавать гексадецимальное постоянное, длиной не более 8 цифр в области

0≤постоянное≤FFFFFFFF.

Редактор PLC сообщает об ошибке, если внесённое значение выходит за заданными пределами. Ведущие нули можно пропустить.

Например, при желании внести в магазин гексадецимальное значение 0AB4C9E6, надо писать

#$AB4C9E6.



4.7 Задача чисел BCD без знака оператором #$

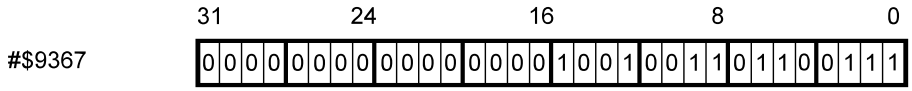
Оператором #$ можно задавать постоянное BCD без знака (бинарно кодированное десятичное), с длиной не более 8 цифр в области

0≤постоянное≤99999999.

Значит, при желании записать в магазин данные BCD, то с ним обращаемся как ексадецимальным числом, но используем только цифры 0, 1, ..., 9. Ведущие нули можно пропустить.

Например, при желании внести в магазин десятичное число 9367, кодировано в BCD, надо писать

#$9367.



4.8 Задача чисел с плавающей запятой оператором "*"

Изображение числа с плавающей запятой в программе PLC выполняется согласно стандарту IEEE754 для изображения числа с плавающей запятой с двойной точностью. Эти числа изображаются на 64 битах. Значит, при желании внести в магазин данные с плавающей запятой, надо всегда занимать место

два двойное слово (64 бит)

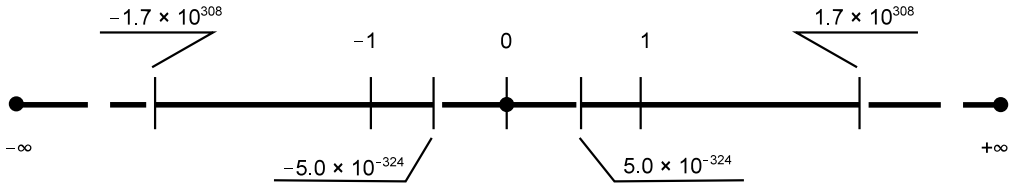
Если данные с плавающей запятой записать по адресу n, данные размещаются по адресу **n** и **n+1**.

Из-за этого нельзя записать данные с плавающей запятой по адресу 9999!

Оператором * можно изображать числа с плавающей запятой примерно в порядке

от $\pm 5.0 \times 10^{-324}$ до $\pm 1.7 \times 10^{308}$

и 0, с точностью до цифр 15-16. При вводе надо использовать **десятичную запятую** (.).



Если надо занести например число −124.753, то надо написать

*−124.753.

Не нужно вносить ни ведущие, ни последующие за ним нули. Знак + (положительный) можно пропустить.

4.9 Изображение чисел с плавающей запятой с двойной точноттью по стандарту IEEE754

В целях информации кратко опишем построение формата IEEE754 с плавающей запятой с двойной точностью (double precision floating point format), однако с *этим програмисту PLC совершенно не нужно заниматься.*

Число с плавающей запятой с двойной точноттью состоит из трёх частей:

– **Бит знака** (s): при его значении 0 число положительное, при 1 - число отрицательное.

– **Показатель степени** или **экспонент** (e): его длина 11 бит, и его основание 2. В этом поле нужно изображать и положительные и отрицательные показатели степени. Для достижения этого к фактическому показателю степени добавим **смещение**, значение которого **1023**. Значит, если значение сохраненного показателя степени e=1201, фактический показатель степени можно получить из зависимости 1201−1023=178. Значение экспонента чисто 0 (000h) и чисто 1 (7FFh) сохранено для специального использования.

– **Мантисса**: Длина мантиссы 53 бита. Она состоит из двух частей: из **однобитной целой части** и из **дробной части** (f), имеющей длину 52 бита. Значение целой части для нормализованных чисел равно 1, а для денормализованных чисел - 0. Следовательно, мантисса имеет 53 бита, но при изображении чисел занимает только 52 бита место, так как изображаем только дробную часть.

Значение показателя степени (e) и дробной части мантиссы (f) влияет и на изображение чисел:

– **Нормализованное число**: если показатель степени не чисто 0 (e≠000h) и не чисто 1 (e≠7FFh), число является нормализованным. При этом для целой части мантиссы предполагаем 1 и число получим из следующей зависимости:

$(-1)^s \times 1.f \times 2^{e-1023}$, где “s”- бит знака, “f” - дробная часть мантиссы, “e” - показатель степени.

И так максимальное изображаемое абсолютное значение, 1. Подставим $f=2-2^{-52}$:

$(-1)^s \times (2-2^{-52}) \times 2^{1023} \approx \pm 1.7 \times 10^{308}$

– **Переполнение**: Если результат операции с плавающей запятой нельзя изображать потому, что абсолютное значение полученного числа превышает изображаемый максимум, то операция выводит статусный бит переполнения **FL_OF** (overflow).

– **Денормализованное число**: если показатель степени чисто 0 (e=000h) но дробная часть мантиссы не равно 0, $f \neq 0$, то число является денормализованным. Этот формат используется для изображения очень малых чисел. При этом для целой части мантиссы предполагаем 0 и число получим из следующей зависимости:

$(-1)^s \times 0.f \times 2^{-1022}$, где “s”- бит знака, “f” - дробная часть мантиссы.

И так минимальное изображаемое абсолютное значение, 0. Подставим $f=2^{-52}$:

$(-1)^s \times 2^{-52} \times 2^{-1022} = (-1)^s \times 2^{-1074} \approx \pm 5.0 \times 10^{-324}$

– **Недополнение**: Если результат операции с плавающей запятой нельзя изображать потому, что абсолютное значение полученного числа меньше изображаемого минимума, операция выводит статусный бит недополнения **FL_UF** (underflow).

– **Ноль**: Нулю является то число, экспонент и дробная часть мантиссы которого 0:

e=0 и f=0

По стандарту различается в зависимости от бита знака (s=0, или s=1) +0 и −0. Значит, ноль является специальным денормализованным числом.

– **Бесконечное**: Если показатель степени чисто 1 и дробная часть мантиссы 0

e=2047(=7FFh) и f=0

число ±∞, в зависимости от бита знака s.

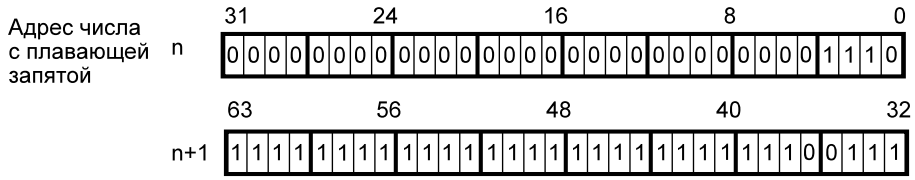
– **Не Число**, **NaN** (Not a Number): изображенное значение не считается действительным числом тогда, если показатель степени чисто 1 и дробная часть мантиссы не 0:

e=2047(=7FFh) и $f \neq 0$

– **Сообщение об ошибке на значение Не Число**: Если одно из чисел, участвующих в операции с плавающей запятой представляет собой Не Число (NaN), например, в следствие пропущенной инициализации, управлениемвыводится статусный бит **FL_ER**.

Расположение числа с плавающей запятой в памяти

Нижние 32 бита числа с плавающей запятой расположены по заданному адресу, пока его верхние 32 бита занимают двойное слово после заданного адреса:



5 Статусные биты, ставленные командами PLC

Сообщения, ставленные командами PLC, служат для показа, статуса результата или ошибки какой-то операции.
Статусные биты находятся в двойном слове с физическим адресом 0000, а со символическим адресом FLAGS. Ссылаться на это слово можно из программы PLC **только чтением, писать нельзя**.

5.1 Сообщение об ошибке FL_ER (error)

Система напишет сообщение в 1, если в выполненной программе PLC находит ошибку. Эти могут быть слежующие:

заданный адрес выходит за пределами значения 0000....9999,
заданный адрес выходит за возможными пределами значения,
заданные данные имеются не в правильном формате,
одны из данных операции с плавающей запятой являются NaN (Не Число)
вводные параметры команды с ошибкой,
команда выполнимая.

Обращением сообщения FL_ER отдельно будем заниматься при изложении отдельных команд.

5.2 Сообщение о недополнении FL_UF (underflow)

Система напишет сообщение в 1, если:

– если сумма, полученная после сложения двух отрицательных чисел с фиксированной запятой попадает в область положительных чисел 00000000 ... 07FFFFFF, а в прочем 0,
– если после вычитания положительного числа из отрицательного числа с фиксированной запятой, разность попадает в область положительных чисел 00000000 ... 07FFFFFF, а в прочем 0,
– если при операции с плавающей запятой, абсолютное значение результата будет настолько мало, что нельзя его изображать с изображением чисел с плавающей запятой с двойной точностью.

5.3 Сообщение о переполнении FL_OF (overflow)

Система напишет сообщение в 1, если:

– если сумма, полученная после сложения двух положительных чисел с фиксированной запятой попадает в область отрицательных чисел 80000000...FFFFFFFF, а в прочем 0,
– если после вычитания отрицательного числа из положительного числа с фиксированной запятой, разность попадает в область отрицательных чисел 80000000 ... FFFFFFFF, а в прочем 0,
– если при операции с плавающей запятой, абсолютное значение результата будет настолько велика, что нельзя его изображать с изображением чисел с плавающей запятой с двойной точностью.

5.4 Сообщение о переносе FL_CY (carry)

Система напишет сообщение в 1, если:
– если при сложении с фиксированной запятой образуется перенос, то есть сумма не вмещается на 32 битах,
– если при вычитании с фиксированной запятой образуется долг, а в прочем 0.

5.5 Сообщение FL_GT (greater than) больше, чем

Система напишет сообщение в 1,
– если при сравнении двух чисел значение левой стороны больше, чем значения правой стороны: A>B.

5.6 Сообщение о равенстве FL_EQ (equal)

Система напишет сообщение в 1,
– если при сравнении двух чисел, эти два числа равны друг другу: A=B.

5.7 Сообщение FL_LT (lower than) меньше, чем

Система напишет сообщение в 1,
– если при сравнении двух чисел значение левой стороны меньше, чем значения правой стороны: A<B.

6 Команды программы PLC

6.1 Команды, выполняющие бинарные операции

Команды, выполняющие бинарные операции, работают с двумя значениями, с 0 и 1. Значение 0 считаем состоянием ФАЛЬШ, а значение 1 - ВЕРНО. С помощью бинарных команд выполняются разные Боолалгебраические операции, такие как И, ИЛИ и ИСКЛЮЧЕНО ИЛИ. Вводы логических операций относятся всегда к соответствующим битам памяти, а результаты операций пишут соответствующие биты памяти.

6.1.1 Замыкающий контакт: запрос бита памяти

Символ замыкающего контакта: 

Вводный параметр замыкающего контакта:

– **Address of Bit**:

Предел значения (n.i): на адрес двойного слова: n=0000...9999, на адрес бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

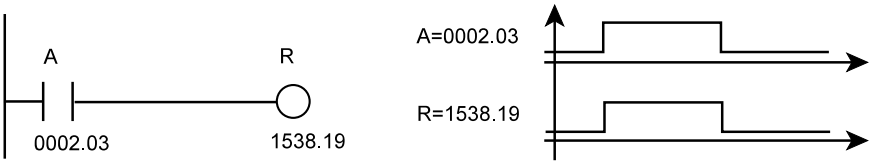
Возможные члены для поправки:

перед адресом: @ обращение с набегающим фронтом или % обращение с задним фронтом,

после адреса: оператор индексирования “,”.

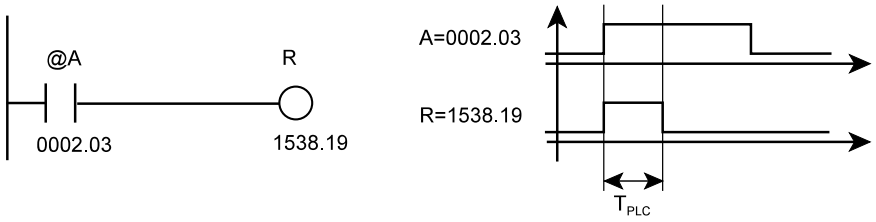
Работа замыкающего контакта:

Через замыкающий контакт проходит ток (контакты замкнуты), если значение бита памяти ВЕРНО (1), не проходит ток (контакты разомкнуты), если значение бита памяти ФАЛЬШ (0). Обмотка реле, подключенная за контактом, согласно этому срабатывает или отпускает.



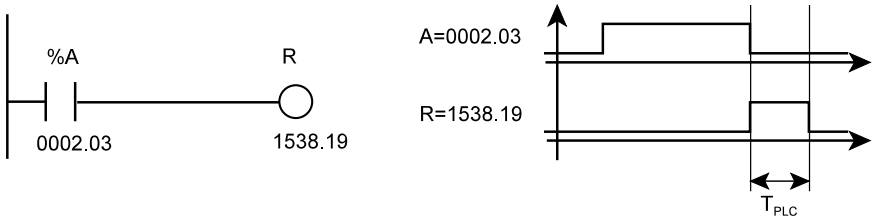
Работа замыкающего контакта с оператором набегающего фронта @

Если перед адресом замыкающего контакта написать оператор @, через замыкающие контакты проходит ток только после сигнала набегающего фронта (смена бита памяти из 0 в 1) в течение 1 цикла PLC, то есть реле R будет сработавшийся в течение времени T_{PLC}.



Работа замыкающего контакта с оператором заднего фронта %

Если перед адресом замыкающего контакта написать оператор %, через замыкающие контакты проходит ток только после сигнала заднего фронта (смена бита памяти из 1 в 0) в течение 1 цикла PLC, то есть реле R будет сработавшийся в течение времени T_{PLC}.



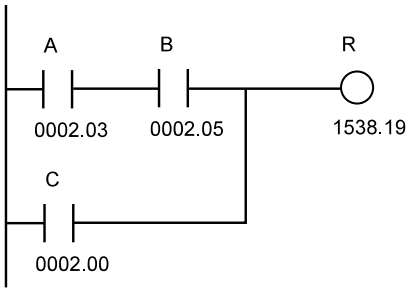
Работа замыкающего контакта с оператором немедленного запроса !

Если перед адресом замыкающего контакта написать оператор !, состояние замыкающего контакта берётся из памяти TimeSlice.

Пример для применения замыкающих контактов

Замыкающими контактами, соединёнными последовательно, можно осуществить логическую связь И, а соединёнными параллельно - логическую ИЛИ. Приложенный рисунок приводит пример для следующей логической всязи:

(A AND B) OR C = R



6.1.2 Замыкающий контакт: запрос двойного слова

Символ замыкающего контакта: ⊣ ⊢

Вводной параметр замыкающего контакта:

– **Address of Bit**:

Предел значения: n=0000...9999

Адрес двойного слова задаётся численно или символично. Возможно задача индек-

сированно.

Возможные члены для поправки:

перед адресом: @ обращение с набегающим фронтом или % обращение с задним фронтом,

после адреса: оператор индексирования “,”.

Работа замыкающего контакта:

Через замыкающий контакт проходит ток (контакты замкнуты), если значение двойного слова >0, не проходит ток (контакты разомкнуты), если значение двойного слова=0. Обмотка реле, подключенная за контактом, согласно этому срабатывает или отпускает.

Команда используется для анализа таймеров, счётчиков и т.д. Например, для решения того, что работает ли ещё таймер (>0), то есть проходит ли ток через контакты, или уже сработал (=0), то есть через контакты не проходит ток.

6.1.3 Размыкающий контакт: отказ запроса бита памяти

Символ размыкающего контакта: 

Вводной параметр размыкающего контакта:

– **Address of Bit**:

Предел значения (n.i): по адресу двойного слова: n=0000...9999, по адресу бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

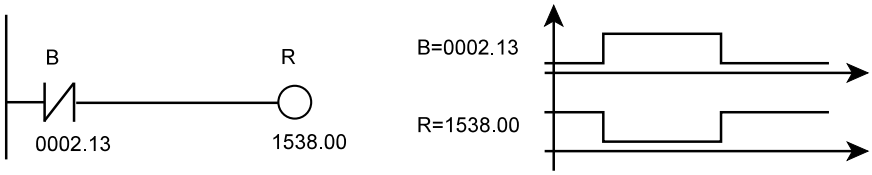
Возможные члены для поправки:

перед адресом: @ обращение с набегающим фронтом или % обращение с задним фронтом,

после адреса: оператор индексирования “,”.

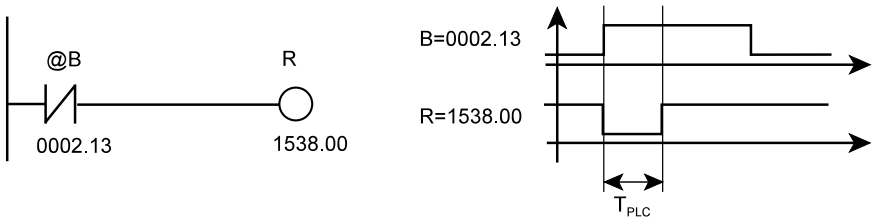
Работа размыкающего контакта:

Через размыкающий контакт проходит ток (контакты замкнуты), если значение бита памяти ФАЛЬШ (0), не проходит ток (контакты разомкнуты), если значение бита памяти ВЕРНО (1). Обмотка реле, подключенная за контактом, согласно этому срабатывает или отпускает.



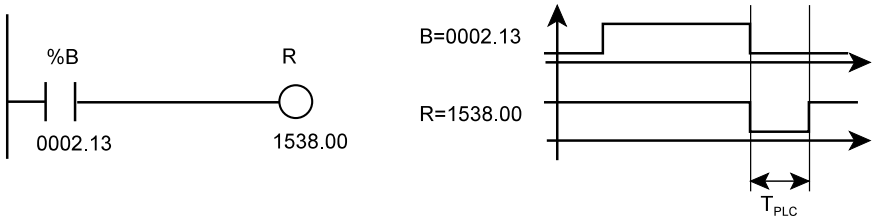
Работа размыкающего контакта с оператором набегающего фронта @

Если перед адресом размыкающего контакта написать оператор @, через размыкающие контакты не проходит ток после сигнала набегающего фронта (смена бита памяти из 0 в 1) в течение 1 цикла PLC, то есть реле R будет отпадавшееся в течение времени T_{PLC}.



Работа размыкающего контакта с оператором заднего фронта %

Если перед адресом размыкающего контакта написать оператор %, через размыкающие контакты не проходит ток после сигнала заднего фронта (смена бита памяти из 1 в 0) в течение 1 цикла PLC, то есть реле R будет отпадавшееся в течение времени T_{PLC}.



Работа размыкающего контакта соператором немедленного запроса !

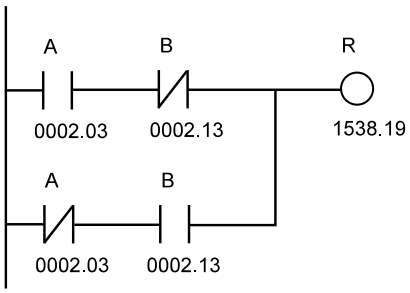
Если перед адресом размыкающего контакта написать оператор !, состояние размыкающего контакта берётся из памяти TimeSlice.

Пример для применения размыкающих контактов

Применением размыкающих и замыкающих контактов можно осуществить связь ИСКЛЮЧЕНО ИЛИ. Как известно, связь ИСКЛЮЧЕНО ИЛИ можно записать в следующей форме:

A XOR B = (A AND (NOT B)) OR ((NOT A) AND B)

Приложенный рисунок показывает осуществление этой связи с размыкающими и замыкающими контактами.



6.1.4 Размыкающий контакт: отказ запроса двойного слова

Символ размыкающего контакта: 

Вводной параметр размыкающего контакта:

– **Address of Bit**:

Предел значения: n=0000...9999

Адрес двойного слова задаётся численно или символично. Возможно задача индексированно.

Возможные члены для поправки:

перед адресом: @ обращение с набегающим фронтом или % обращение с задним фронтом,

после адреса: оператор индексирования “,”.

Работа размыкающего контакта:

Через размыкающий контакт проходит ток (контакты замкнуты), если значение двойного слова=0, не проходит ток (контакты разомкнуты), если значение двойного слова >0. Обмотка реле, подключенная за контактом, согласно этому срабатывает или отпускает.
Команда используется для анализа таймеров, счётчиков и т.д.. Например, для решения того, что работает ли ещё таймер (>0), то есть через контакты не проходит ток, или уже сработал (=0), то есть через контакты проходит ток.

6.1.5 Обмотка реле: запись бита памяти

Символ обмотки реле: 

Вводной параметр обмотки реле:

– **Address of Bit**:

Предел значения (n.i): по адресу двойного слова: n=0000...9999, по адресу бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

Возможные члены для поправки:

после адреса: оператор индексирования “,”.

– Remark:

Замечание: текст, записанный в Remark будет комментарием ступени.

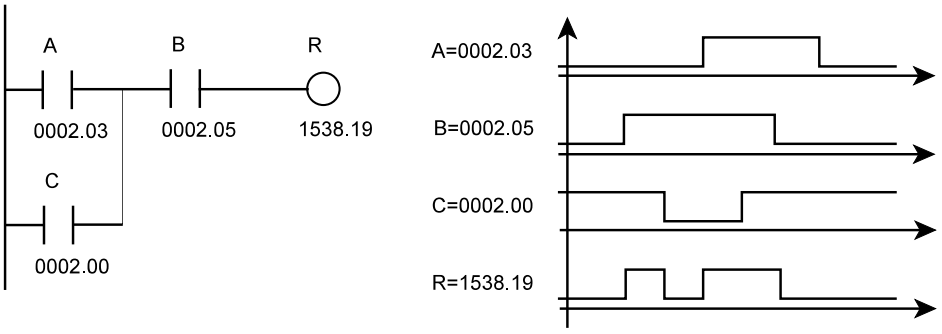
Работа обмотки реле с оператором немедленной выдачи !

Если перед адресом обмотки реле написать оператор !, состояние обмотки реле передаётся в память TimeSlice.

Работа обмотки реле

Если через ввод обмотки реле проходит “ток”, то есть, если состояние совокупности условий перед обмоткой ВЕРНО, реле срабатывает, то есть бит памяти, относящийся к обмотке, команда запишет в 1.
И наоборот, если через ввод обмотки реле не проходит “ток”, то есть, если состояние совокупности условий перед обмоткой ФАЛЬШ, реле отпадает, то есть бит памяти, относящийся к обмотке, команда запишет в 0.

Обмотка реле является так называемым запирающим элементом, то есть не имеет вывода, за ней уже нельзя что-нибудь подключить.



6.1.6 Отверженная обмотка реле (отказ записи бита памяти)

Символ отверженной обмотки реле: 

Вводной параметр отверженной обмотки реле:

– **Address of Bit**:

Предел значения (n.i): по адресу двойного слова: n=0000...9999, по адресу бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

Возможные члены для поправки:

после адреса: оператор индексирования “,”.

– Remark:

Замечание: текст, записанный в Remark будет комментарием ступени.

Работа отверженной обмотки реле с оператором немедленной выдачи !

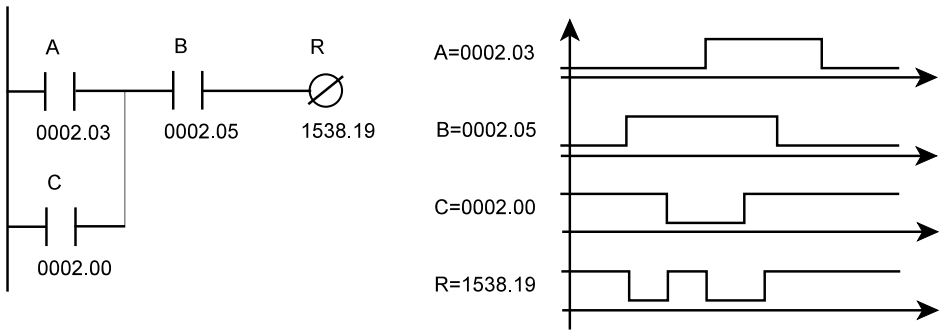
Если перед адресом отверженной обмотки реле написать оператор !, состояние отверженной обмотки реле выдаётся в память TimeSlice.

Работа отверженной обмотки реле

Если через ввод отверженной обмотки реле проходит “ток”, то есть, если состояние совокупности условий перед обмоткой ВЕРНО, реле отпадает, то есть бит памяти, относящийся к обмотке, команда запишет в 0.

И наоборот, если через ввод обмотки реле не проходит “ток”, то есть, если состояние совокупности условий перед обмоткой ФАЛЬШ, реле срабатывает, то есть бит памяти, относящийся к обмотке, команда запишет в 1.

Отверженная обмотка реле является так называемым запирающим элементом, то есть не имеет вывода, за ней уже нельзя что-нибудь подключить.



6.1.7 Установка бита памяти: команда SET

Символ команды SET: 

Вводной параметр команды SET:

– **Address of Bit**:

Предел значения (n.i): по адресу двойного слова: n=0000...9999, по адресу бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

Возможные члены для поправки:

после адреса: оператор индексирования “,”.

– Remark:

Замечание: текст, записанный в Remark будет комментарием ступени.

Работа команды SET с оператором немедленной выдачи !

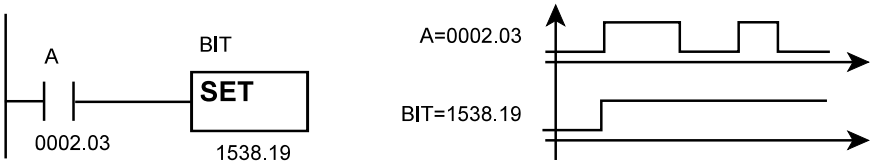
Если перед адресом команды SET написать оператор !, он запишет бит в память TimeSlice

Работа команды SET

Если совокупность условий до команды включается в состояние ВЕРНО, команда запишет относящийся бит памяти в 1.

Если после этого совокупность условий до команды переключается в состояние ФАЛЬШ, бит памяти остаётся неизменным.

Команда SET является так называемым запирающим элементом, то есть не имеет вывода, за ней уже нельзя что-нибудь подключить.



6.1.8 Удаление бита памяти: команда RST

Символ команды RST: 

Вводной параметр команды RST:

– **Address of Bit**:

Предел значения (n.i): по адресу двойного слова: n=0000...9999, по адресу бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

Возможные члены для поправки:

после адреса: оператор индексирования “,”.

– Remark:

Замечание: текст, записанный в Remark будет комментарием ступени.

Работа команды RST с оператором немедленной выдачи !

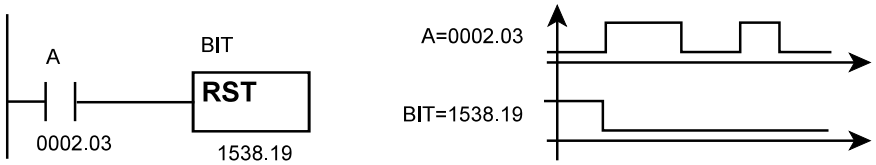
Если перед адресом команды RST написать оператор !, он удаляет бит из памяти TimeSlice.

Работа команды RST

Если совокупность условий до команды включается в состояние ВЕРНО, команда запишет относящийся бит памяти в 0.

Если после этого совокупность условий до команды переключается в состояние ФАЛЬШ, бит памяти остаётся неизменным.

Команда RST является так называемым запирающим элементом, то есть не имеет вывода, за ней уже нельзя что-нибудь подключить.



6.1.9 Создание импульса на набегающий фронт: команда DIFU

Символ команды DIFU: 

Вводной параметр команды DIFU:

– **Address of Bit**:

Предел значения (n.i): по адресу двойного слова: n=0000...9999, по адресу бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

Возможные члены для поправки:

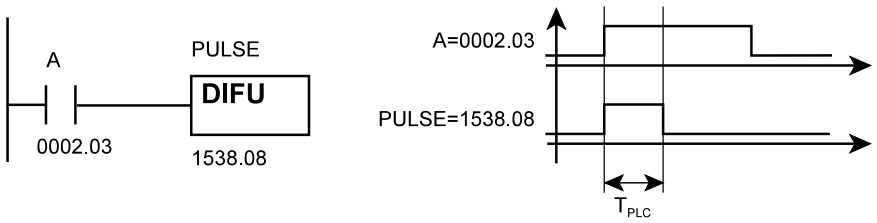
после адреса: оператор индексирования “,”.

– Remark:

Замечание: текст, записанный в Remark будет комментарием ступени.

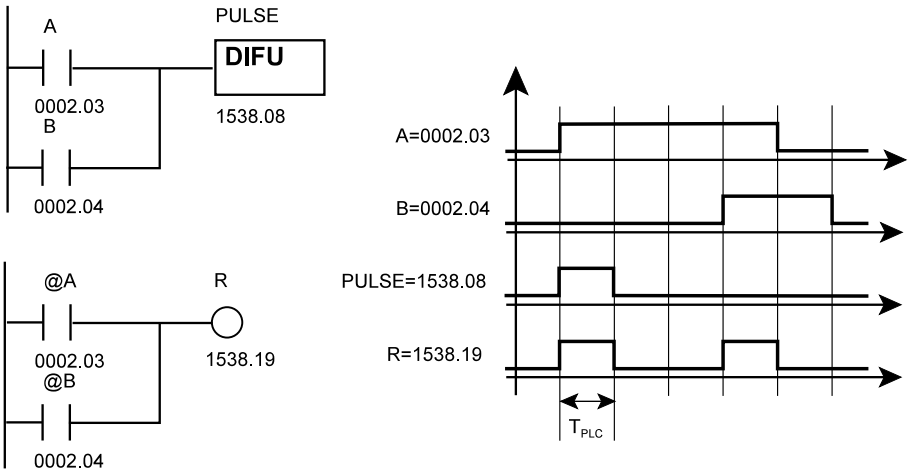
Работа команды DIFU

Если совокупность условий до команды переключается из состояния ФАЛЬШ (0) в состояние ВЕРНО (1), команда запишет относящийся бит памяти в 1 в силу 1 цикла PLC (T_{PLC} время), то есть дифференцирует на набегающий фронт сигнала.
Команда DIFU является так называемым запирающим элементом, то есть не имеет вывода, за ней уже нельзя что-нибудь подключить.



Применение команды DIFU

Приведенная ниже пример обращает внимание на разницу между применением команды DIFU и оператора @. Если дифференцировать результат зависимости (A OR B) командой DIFU, получим другой результат, как будто рассматривать альтернативно набегающий фронт A и набегающий фронт B и результат сохраним на реле R.



6.1.10 Создание импульса на задний фронт: команда DIFD

Символ команды DIFD: 

Вводной параметр команды DIFD:

– **Address of Bit**:

Предел значения (n.i): по адресу двойного слова: n=0000...9999, по адресу бита: i=00..31

Адрес бита задаётся численно или символично. Возможно задача индексированно.

Возможные члены для поправки:

после адреса: оператор индексирования “,”.

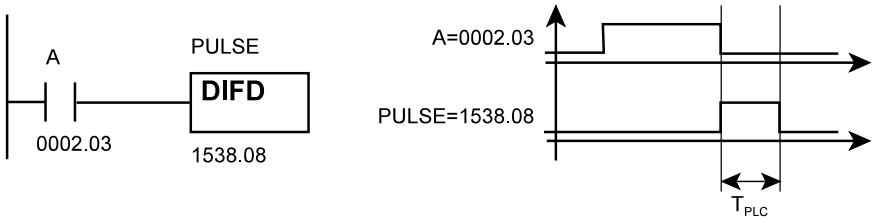
– **Remark:**

Замечание: текст, записанный в Remark будет комментарием ступени.

Работа команды DIFD

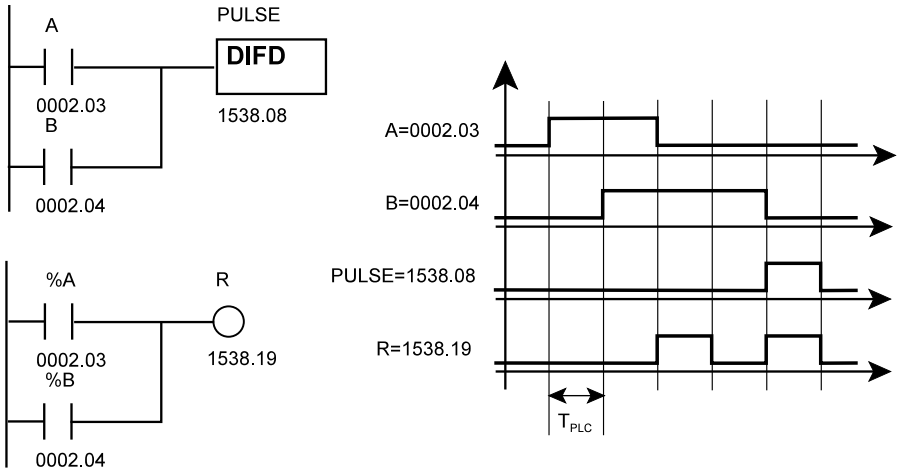
Если совокупность условий до команды переключается из состояния ВЕРНО (1) в состояние ФАЛЬШ (0), команда запишет относящийся бит памяти в 1 в силу 1 цикла PLC (T_{PLC} время), то есть дифференцирует на задний фронт сигнала.

Команда DIFD является так называемым запирающим элементом, то есть не имеет вывода, за ней уже нельзя что-нибудь подключить.



Применение команды DIFD

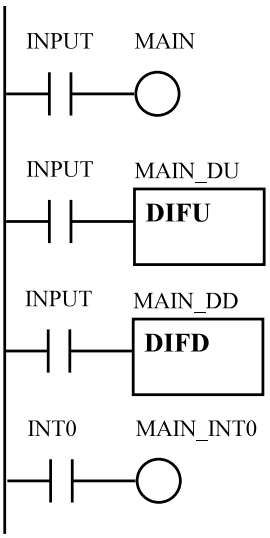
Приведенная ниже пример обращает внимание на разницу между применением команды DIFD и оператора %. Если дифференцировать результат зависимости (A OR B) командой DIFD, получим другой результат, как будто рассматривать альтернативно задний фронт A и задний фронт B и результат сохраним на реле R.



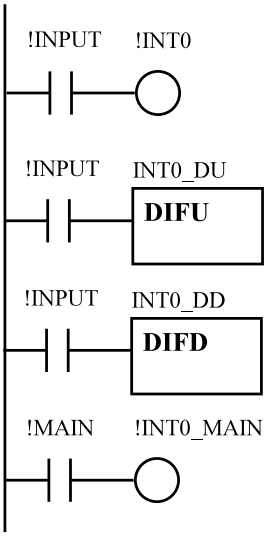
6.1.11 Команды, выполняющие бинарных операций и оператор ! в двух модулях

На основании двух фрагмента программы рассматриваем команды, выполняющие бинарные операции в Главной программе и в модуле Int0.

Главная программа



Модуль Int0



Ввод под названием INPUT

– Главная программа читает из памяти PLC,

– пока модуль Int0 с адресовкой !INPUT из памяти TimeSlice.

Вывод MAIN

– Главная программа запишет в память PLC,

пока INT0 вывод

– модуль Int0 запишет в память TimeSlice с адресовкой !INT0.

Команды DIFU, DIFD используем в обоих модулях.

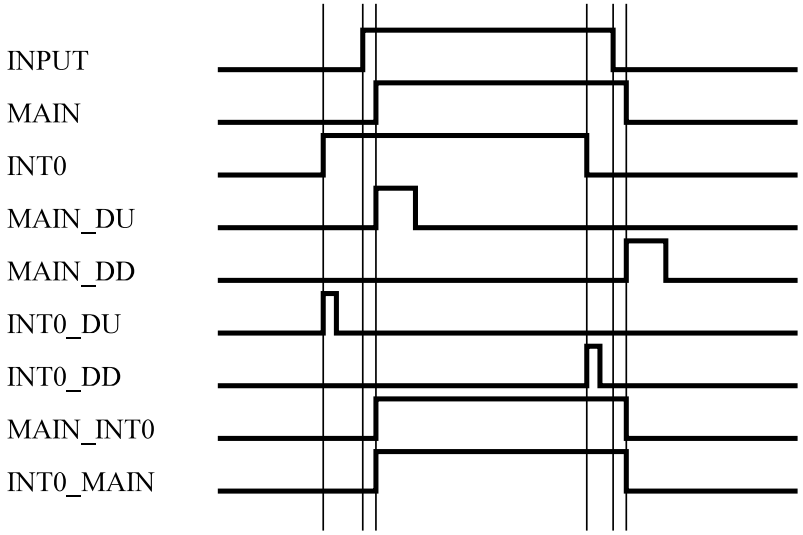
Выводы сохраняем в Главной программе на адресах MAIN_DU, MAIN_DD, в модуле Int0 на адресах INT0_DU, INT0_DD.

На приведенной ниже программе времени видно прохождение по времени отдельных сигналов.

Сигнал INPUT обновляется из памяти PLC, пока сигнал INT0 берёт свой ввод из памяти TimeSlice и туда запишет вывод. Это является причиной того, что сигнал INT0 опережает по времени сигнал INPUT.

На выводе команд DIFU, DIFD хорошо видно,что модуль Int0 выполняется чаще, чем Главная программа, поэтому выводы INT0_DU, INT0_DD находятся включенно короче по времени,чем выводы MAIN_DU, MAIN_DD.

Между сигналами MAIN_INT0 и INT0_MAIN нет разницы. Сигнал MAIN_INT0 берётся из памяти PLC (команда INT0), что обновляется через время T_{PLC}. Зря записать сигнал INT0_MAIN в память TimeSlice, зря берётся ввод из памяти TimeSlice, так как сигнал MAIN обновляется через время T_{PLC}, поэтому эти два сигнала MAIN_INT0 и INT0_MAIN протекают одинакого по времени.


INPUT
MAIN
INT0
MAIN_DU
MAIN_DD
INT0_DU
INT0_DD
MAIN_INT0
INT0_MAIN

6.2 Основные правила создания ступенчатой сетки

Контакты и команды ступенчатой диаграммы надо соединить. Соединительные элементы - это не команды, они не имеют никакого параметра ввода и вывода, они являются толко “проводами“, служащими для обозначения “пути тока”, графическими элементамы, которые занимают место одной ячейки. Эти элементы видны на рисунке ниже, где одна данная клетка относится к одной данной строчке или столбце.
Графические элементы формируются в строчки и столбцы. Одна ступень представляет собой одну логическую единицу, которая может состоять из одной или из нескольких строчек. В программе PLC

число ступеней,

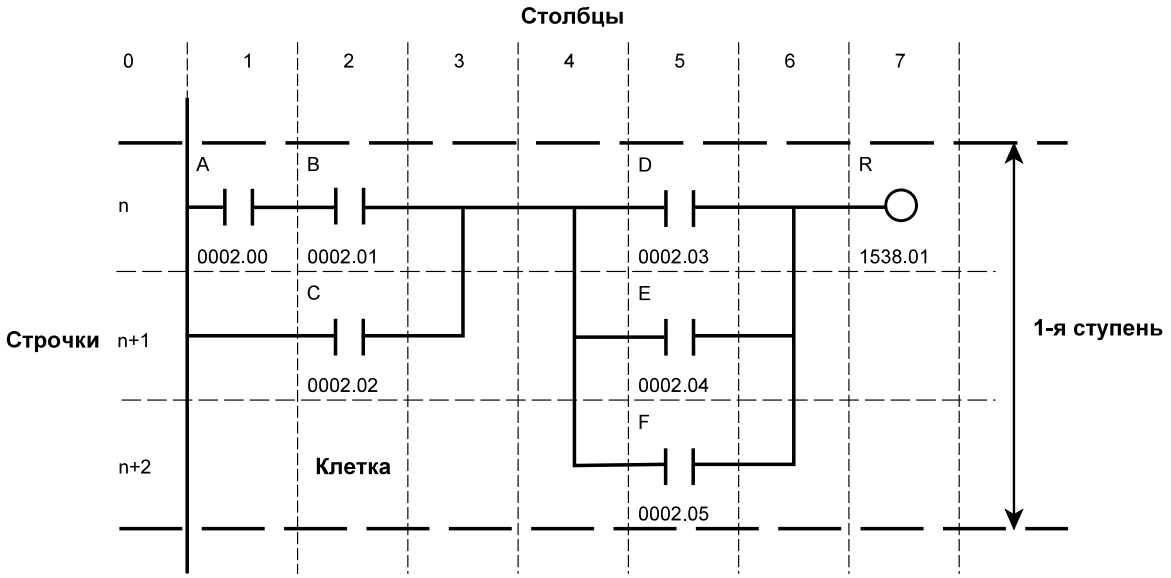
внутри одной ступени

число строчек

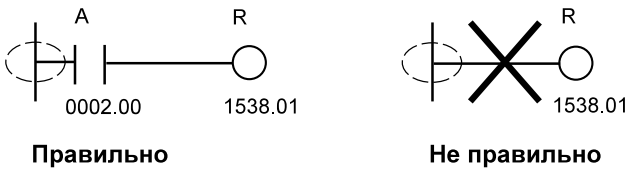
внутри одной строчки

число столбец

то есть число графических элементов, записанных в одну строчку (линии, контакты, команды) не ограничено.

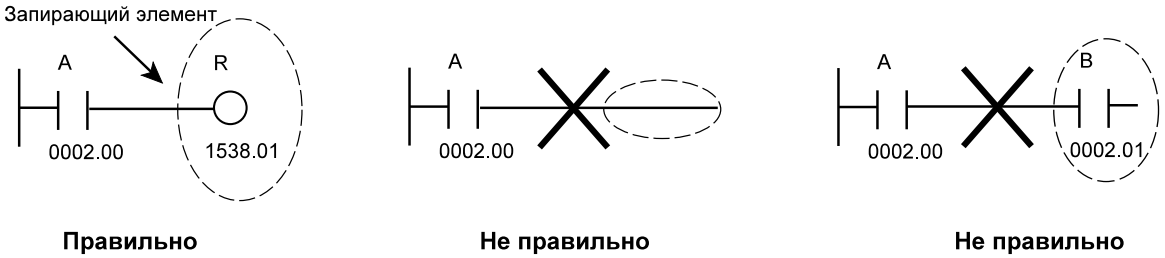


Большинство **команд**, за исключением пару команд, **нельзя записать непосредственно в первый столбец**, только после одного контакта. Значит, нельзя подключить непосредственно например обмотку реле, или команду SET к вертикальному проводу с левой стороны.



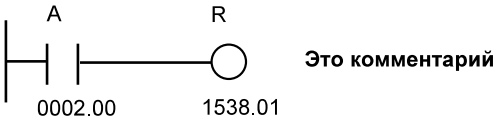
Последний элемент ступени, правую крайную команду, уже никуда не подключаем.

Последний элемент называется
запирающим элементом.
Запирающий элемент не может быть графическим элементом, контактом, или такой командой, который имеет выход.



Правильно **Не правильно** **Не правильно**

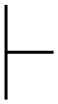
В случае наличия запирающего элемента всегда можно задавать **комментарий** с заполнением параметра Remark. Комментарий запишется в строчку запирающего элемента, начиная со столбца вслед за запирающим элементом.



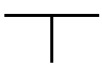
6.2.1 Соединительные элементы

Соединительные элементы - это не команды, они не имеют никакого вводного или выводного параметра, они являются толко “проводами“, служащими для обозначения “пути тока”, графическими элементамы, которые занимают место одной ячейки.
PLC со ступенчатой диаграммой использует следующие соединительные элементы:

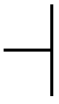
Вертикальный-Правый:



Горизонтальный-Вниз:



Левый-Вертикальный:



Горизонтальный-Верх:

Вертикальный:

Горизонтальный:

Верх-Правый: 

Левый-Верх: 

6.2.2 Комментирование логических секторов ступенчатой сетки: команда SEC

Как уже видели, после запирающих элементов можно писать и комментарий. Существует среди команд такая, которая служит исключительно для задачи комментария.

Символ команды SEC: 

☞ ***Внимание!*** *Команда SEC, хоть она является запирающим элементом, можно записать исключительно в 1-й столбец!*

Вводной параметр команды SEC:

– Remark:

Замечание: текст, записанный в Remark будет комментарием ступени.

Описание команды SEC

Команда SEC не выполняет никакую операцию, она служит исключительно для того, чтобы отделить один логический сектор программы PLCот другого. В качестве вводного параметра можно задавать только текстовый комментарий.

6.3 Команды для перемещения данных

Командами для перемещения данных можно задавать значение для любого регистра памяти PLC, или любой регистр можно копировать на другой адрес.
При задаче значений всегда надо задавать формат вводимых данных:

#: десятичное целое число со знаком,
#$: гексадецимальное число, или
*: число с плавающей запятой.

Для адресовки регистров(символческая, числовая, или индексированная) действительны изложенные раньше.

Ввод и вывод команд для перемещения данных

Каждая команда для перемещения данных имеет

разрешающий ввод.

Разрешающий ввод в состояние ВЕРНО выполняет перемещение.
Для команд для перемещения данных *можно конфигурировать*

вывод.

Вывод примет состояние ВЕРНО, если ввод команды ВЕРНО и выполнил операцию перемещения. Вывод примет состояние ФАЛЬШ, если ввод ФАЛЬШ, или, если команду нельзя выполнить, то есть если команда выводит сообщение об ошибке FL_ER!
К выводу можно подключить хоть и новую команду для перемещения данных.
Все команды для перемещения данных имеют общие вводные параметры.

Вводные параметры команд для перемещения данных

– **Address of Result**:
  Предел значения: n...9999 (n: где начинаются адреса пользователей)
  Адрес целевого регистра можно задавать численно, или символично. Возможна индексированная задача.
  Заданное значение, или содержание перемещаемого регистра попадает на этот адрес.
– **Operand**:
  Значение перемещаемых данных, или адрес перемещаемого регистра.
  При задаче данных надо записать данные в формате, соответствующем команде.
  В случае адреса регистра, адрес может быть символичным, числовым, или индексированным.
– **Output**:
  В разрешённом состоянии к выводу коробки команд можно подключить дальнейшие команды.
– **Remark**:
  Если вывод не разрешён, записанное сюда замечание выводится в виде комментария, далее, если ничего не запишем сюда, то сюда запишется комментарий символа, заданного в поле Address of Result.

6.3.1 Перемещение двойного слова (DWORD): команда MOV и MVN

Команда

MOV,

если выполняется условие на вводе с длиной DWORD, то запишет в целевой регистр

заданное оператором # десятичное число со знаком, или

заданное оператором #$ гексадецимальное число, или

содержание регистра с длиной DWORD.

Команда

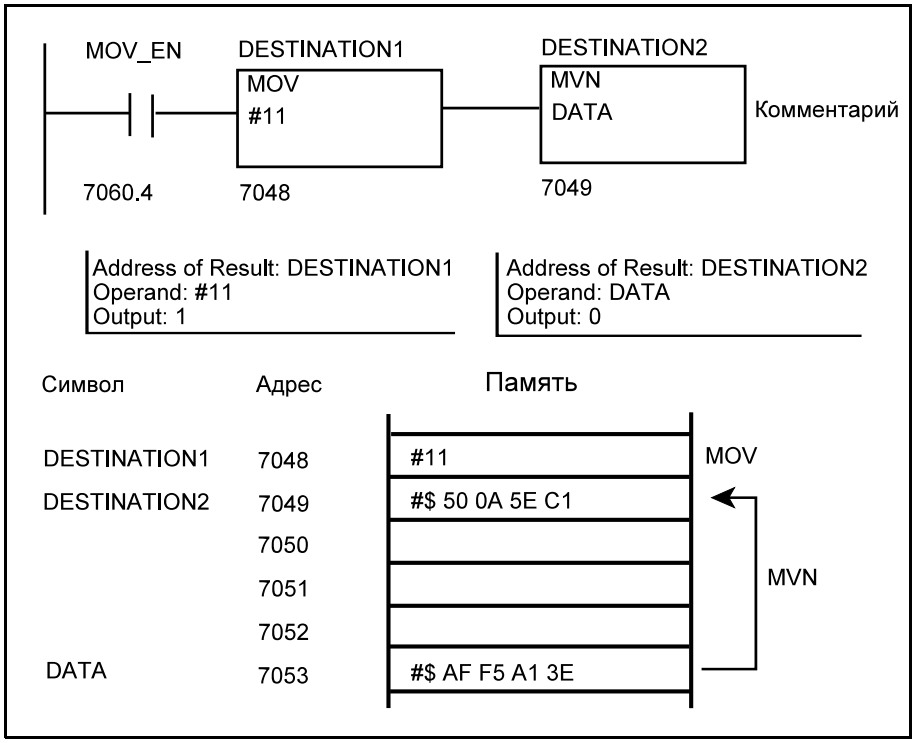
MVN

если выполняется условие на вводе с длиной DWORD,*отвергает по битам* и запишет в целевой регистр

заданное оператором #$ гексадецимальное число, или

содержание регистра с длиной DWORD.

Если разрешён параметр команды Output, за ним можно писать дальнейшие команды.



6.3.2 Перемещение данных с плавающей запятой (double): MOVF

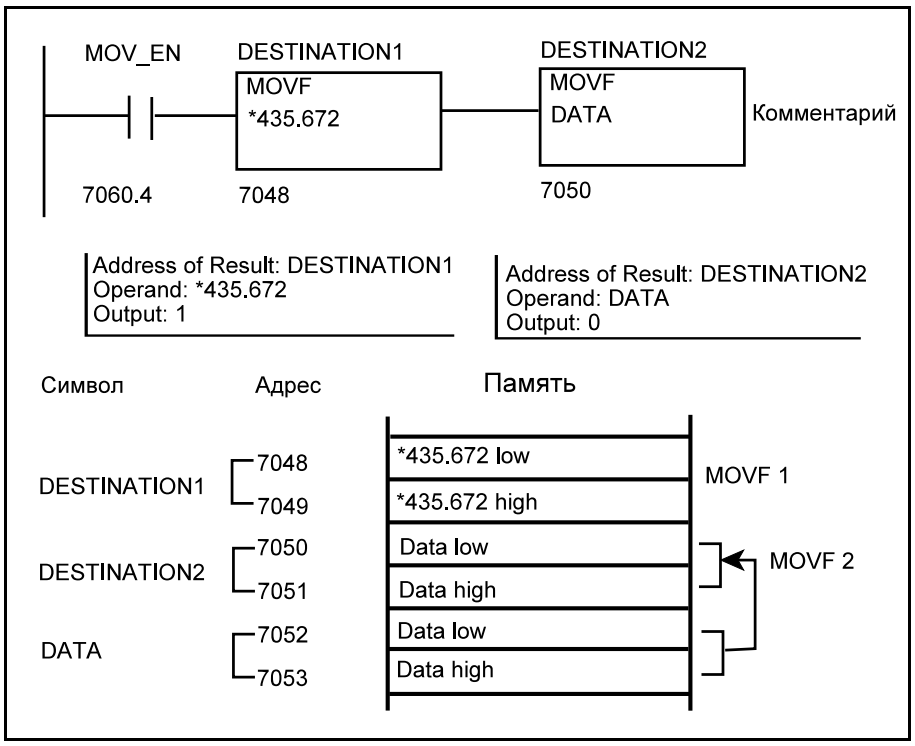
Команда

MOVF

если выполняется условие на вводе с длиной 2 DWORD, запишет в целевой регистр заданное оператором *число с длиной double с плавающей запятой, или содержание 2 штуки регистра с длиной DWORD

☞ ***Внимание!*** *Для Address of Result и для Operand также надо занимать регистр с длиной 2 DWORD.*

Если разрешён параметр команды Output, за ним можно писать дальнейшие команды.



6.4 Таймеры

С помощью таймеров можно установить задержки различного типа и импульсы с различной шириной в программе PLC.

Ввод и вывод таймеров

Каждый таймер имеет один

разрешающий ввод.

Разрешающий ввод управляет работой таймера.

Все таймеры имеют

вывод.

Сигнал на выводе появится согласно работе таймера, с задержкой.

Все таймеры имеют общие вводные параметры.

Вводные параметры таймеров

– **Address of Timer**:

Предел значения: n...9999 (n: где начинаются адреса пользователей)

Адрес таймера можно задавать численно, или символично. Возможна индексированная задача.

Таймер должен дефинировать програмист в области магазина для пользователя.

Адрес таймера указывает на регистр, содержащий мгновенное значение таймера, где таймер считает, и из которого адреса можно вычитать мгновенное значение таймера. Когда таймер на вводе получит набегающий фронт, в этот регистр записывается значение, заданное параметром Set Value и изменённое параметром Time Basis. В любое время, хоть во время работы таймера, этот регистр можно изменить.

Отсчёт вниз в масштабе миллисекунд происходит всегда от значения, записанного в этот регистр. После окончания отсчёта значение регистра остаётся 0 до тех пор, пока не получит на вводе снова разрешение.

☞ ***Внимание:*** *Адрес таймера, или его символичное имя можно использовать в команде, проверяющей условия, то есть и в качестве контакта. Значение условия:*

ПРАВДА: если вычитанное из адреса значение ≠0

ФАЛШ: если вычитанное из адреса значение =0.

– **Set Value:**

Предел значения: $0...2^{31}$

Значение выбранного времени.

Число с фиксированной запятой, или ссылка на регистр. Возможно задача индексированно.

Если таймер разрешить на вводе, это число запишется в регистр, заданный параметром Address of Timer.

– **Time Basis:**

Предел значения: 0, 1, 2, 3

Определяет истолкование значения, заданного параметром Set Value:

0: мсек (миллисекунд)

1: сек (секунд)

2: мин (минута)

3: ч (час)

Может быть число с фиксированной запятой, или постоянный символ (Например. MSEC #0, SEC #1, MIN #2, HOURS #3), но может быть и ссылка на регистр. Возможно задача индексированно.

Предел значение, заданный на параметр Set Value, относится на выбранное время, заданное в мсек. Таймеры считают всегда фондом времени в мсек-ах, заданным в регистре по адресу Address of Timer. Если значение, заданное параметром Time Basis таймера, отличается от мсек-ов, можно задавать следующие пределы значения:

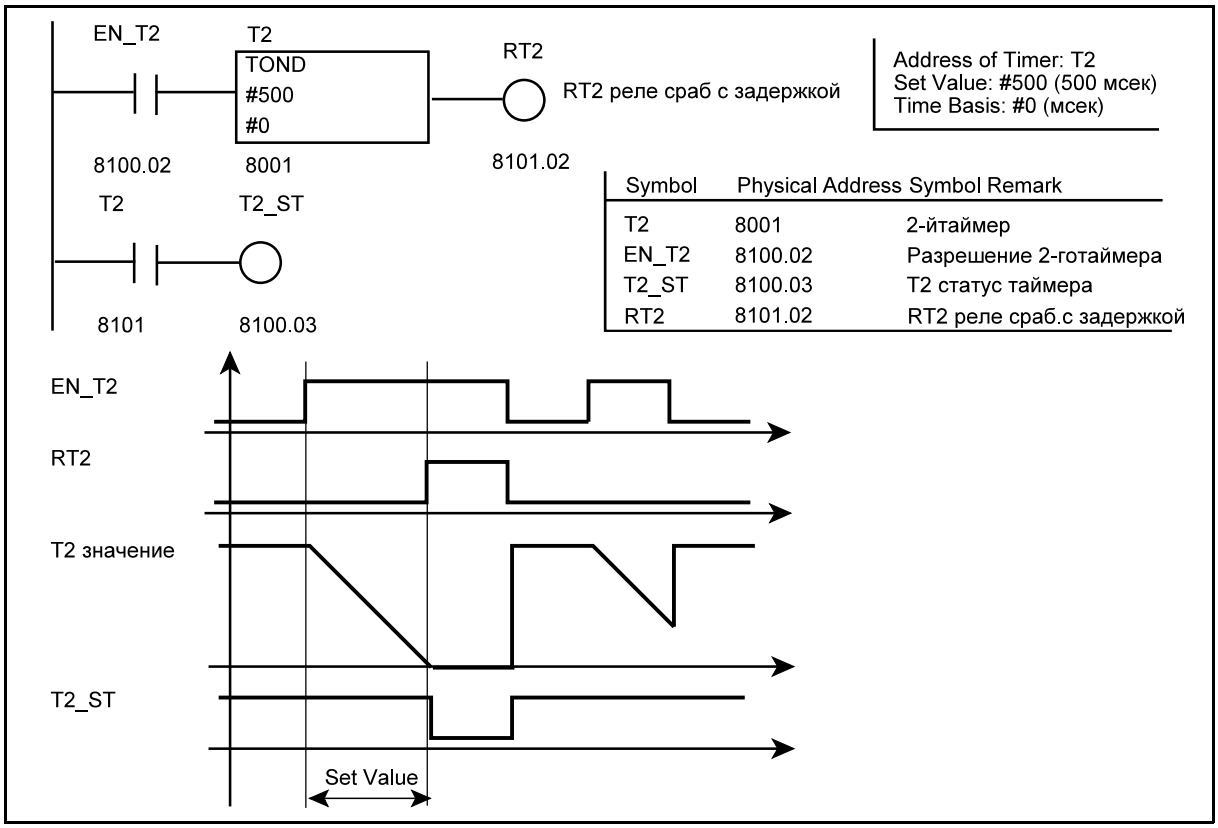
=1 сек: 2147483
=2 мин: 35791
=3 час: 596

– Remark:

Не используется.

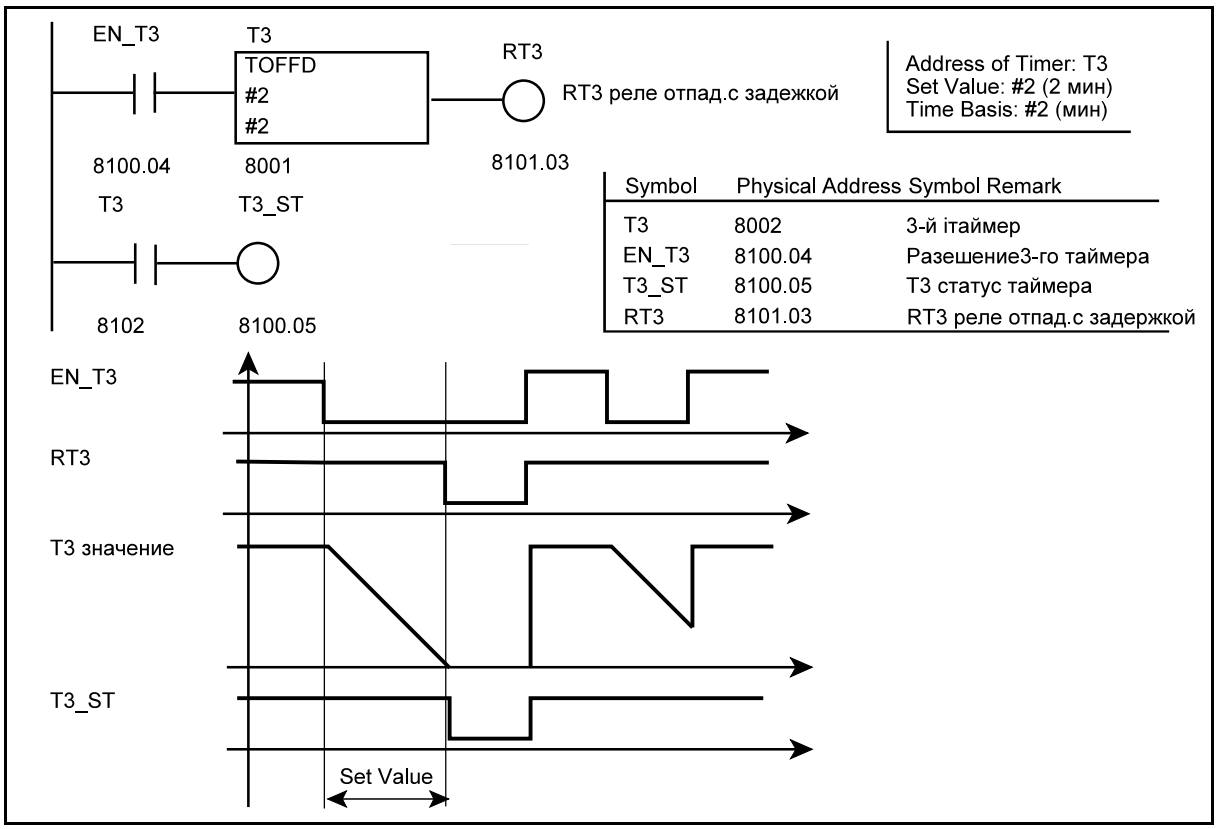
6.4.1 Таймер замыкания с задержкой TOND

На выводе таймера замыкания с задержкой (Timer, ON Delay) появится значение ВЕРНО, видное на вводе, после установленной в таймере задежки.
Если значение разрешающего ввода ВЕРНО, таймер начинает отсчёт вниз, и после выполнения отсчёта значение вывода переключается на ВЕРНО. Если значение разрешающего ввода переключается на ФАЛЬШ, содержание регистра, заданного по адресу Address of Timer, снова примет заданное параметром Set Value значение, изменённое параметром Time Basis. Вывод таймера примет значение ФАЛШ.
На регистр, заданный по адресу Address of Timer можно ссылаться и проверкой условий: если значение регистра ≠0, то значение условия будет ВЕРНО, если =0 - то значение условия ФАЛЬШ.



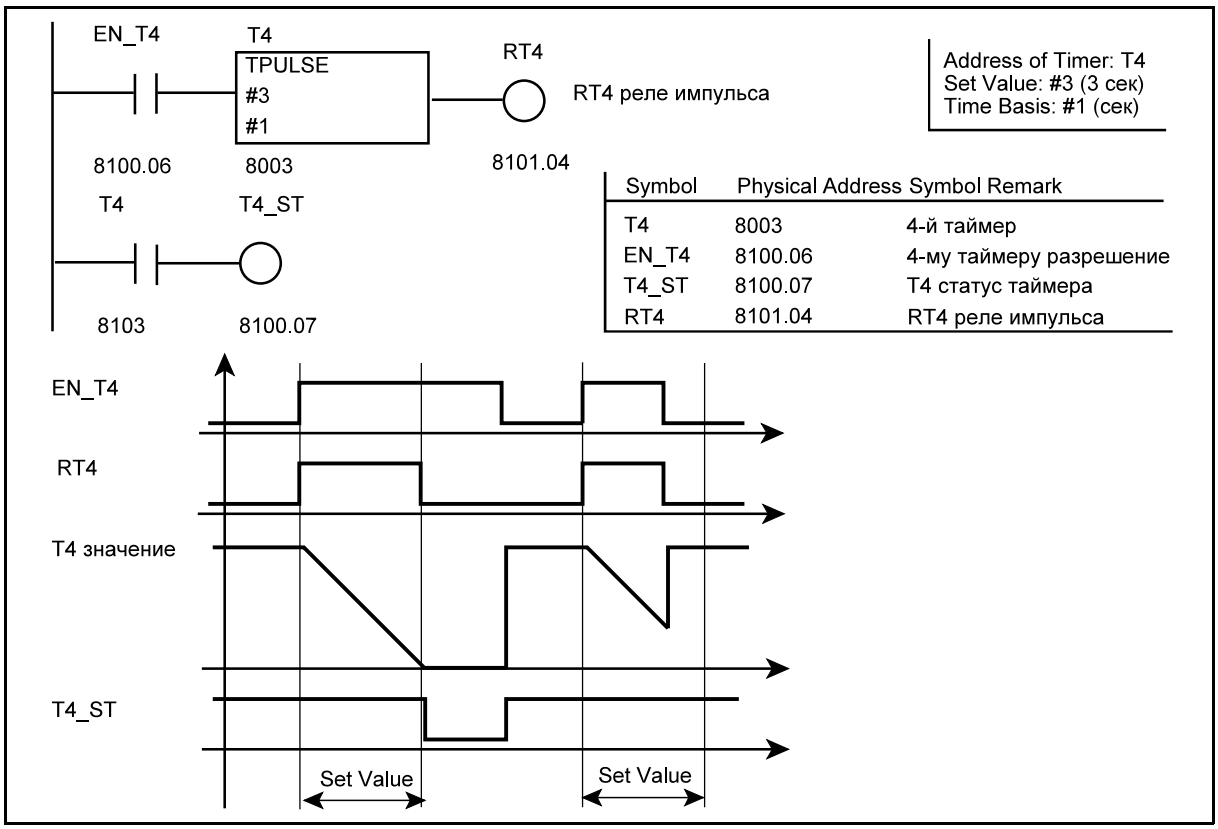
6.4.2 Таймер размыкания с задержкой TOFFD

На выводе таймера размыкания с задержкой (Timer, OFF Delay) появится значение ФАЛЬШ, видное на вводе, после установленной в таймере задежки
Если значение разрешающего ввода ФАЛЬШ, таймер начинает отсчёт вниз, и после выполнения отсчёта значение вывода переключается на ФАЛЬШ. Если значение разрешающего ввода переключается на ВЕРНО, содержание регистра, заданного по адресу Address of Timer, снова примет заданное параметром Set Value значение, изменённое параметром Time Basis. Вывод таймера примет значение ВЕРНО.
На регистр, заданный по адресу Address of Timer можно ссылаться и проверкой условий: если значение регистра ≠0, то значение условия будет ВЕРНО, если =0 - то значение условия ФАЛЬШ.



6.4.3 Импульс с программируемой шириной TPULSE

На выводе таймера импульса сразу появится значение ВЕРНО, видное на вводе, потом вывод примет состояние ФАЛЬШ после установленной в таймере задержки.
Если значение разрешающего ввода ВЕРНО, вывод включается, будет ВЕРНО, и таймер начнёт отсчитывать вниз. Если он выполнил отсчёт, вывод переключается на значение ФАЛЬШ. Если значение разрешающего ввода переключается на ФАЛЬШ, содержание регистра, заданного по адресу Address of Timer снова примет заданное параметром Set Value значение, изменённое параметром Time Basis и вывод тоже будет ФАЛЬШ.
На регистр, заданный по адресу Address of Timer можно ссылаться и проверкой условий: если значение регистра ≠0, то значение условия будет ВЕРНО, haесли =0 - то значение условия ФАЛЬШ.



6.5 Счётчики

С помощью счётчиков можно подсчитать импульсы в программе PLC.

Вводы и вывод счётчиков

Каждый счётчик имеет один ввод
для сигнала часов (CK),
для одного направления (U/D) и одного
приводящего в исходное положение (R).

– **Ввод CK**: Счётчик приводится в действие от набегающего фронта импульса, поступающего на ввод CK.
– **Ввод U/D**: Импульс, поступающий на ввод CK счётчика,
увеличивает значение счётчика, если ввод U/D ФАЛЬШ, далее
уменьшает, если ввод U/D ВЕРНО.
– **Ввод R**: Если ввод R ВЕРНО, счётчик примет первоначальное значение. Первенство ввода R выше ввода CK: если его состояние ВЕРНО счётчик просчёт не выполняет, даже при поступлении импульсов на ввод CK.
– **Вывод** счётчика примет значение ВЕРНО, если содержание счётчика будет равно числу, установленному заранее.

Параметры ввода являются общими для всех счётчиков.

Вводные параметры счётчиков

– **Address of Counter**:
Предел значения: n...9999 (n: где начинаются адреса пользователей)
Адрес счётчика задаётся численно или символично. Возможно задавать индексированно.
Счётчик должен дефинировать програмист в области магазина для пользователя.
Адрес счётчика указывает на регистр, содержащий мгновенное значение счётчика, где счётчик считает, и из которого адреса можно вычитать мгновенное значение счётчика. Когда ввод R счётчика ВЕРНО, в этот регистр записывается значение, заданное параметром Starting Value. В любое время, хоть во время работы счётчика, этот регистр можно изменить.

☞ ***Внимание:*** *Адрес счётчика, или его символичное имя можно использовать в команде, проверяющей условия, то есть и в качестве контакта. Значение условия:*
ВЕРНО: если вычитанное из адреса значение ≠0
ФАЛЬШ: если вычитанное из адреса значение =0.

– **Starting Value:**
Предел значения: $0...2^{31}$
Число с фиксированной запятой, или ссылка на регистр. Возможно задача индексированно.
Начальным значением счётчика является то, что примет под действием состояния ВЕРНО, видное на вводе R. Это является нижним пределом переворачивания счётчика для счётчиков кольцевого типа.

– **Ending Value:**
Предел значения: $0...2^{31}$
Число с фиксированной запятой, или ссылка на регистр. Возможно задача индексированно.
Простые счётчики считаютдо этого значения, это является верхним пределом переворачивания счётчика для счётчиков кольцевого типа.

– **Compare Value:**

Предел значения: $0...2^{31}$

Число с фиксированной запятой, или ссылка на регистр. Возможно задача индексированно.

Вывод счётчика примет состояние ВЕРНО тогда, если содержание счётчика совпадает заданным здесь числом.

– Remark:

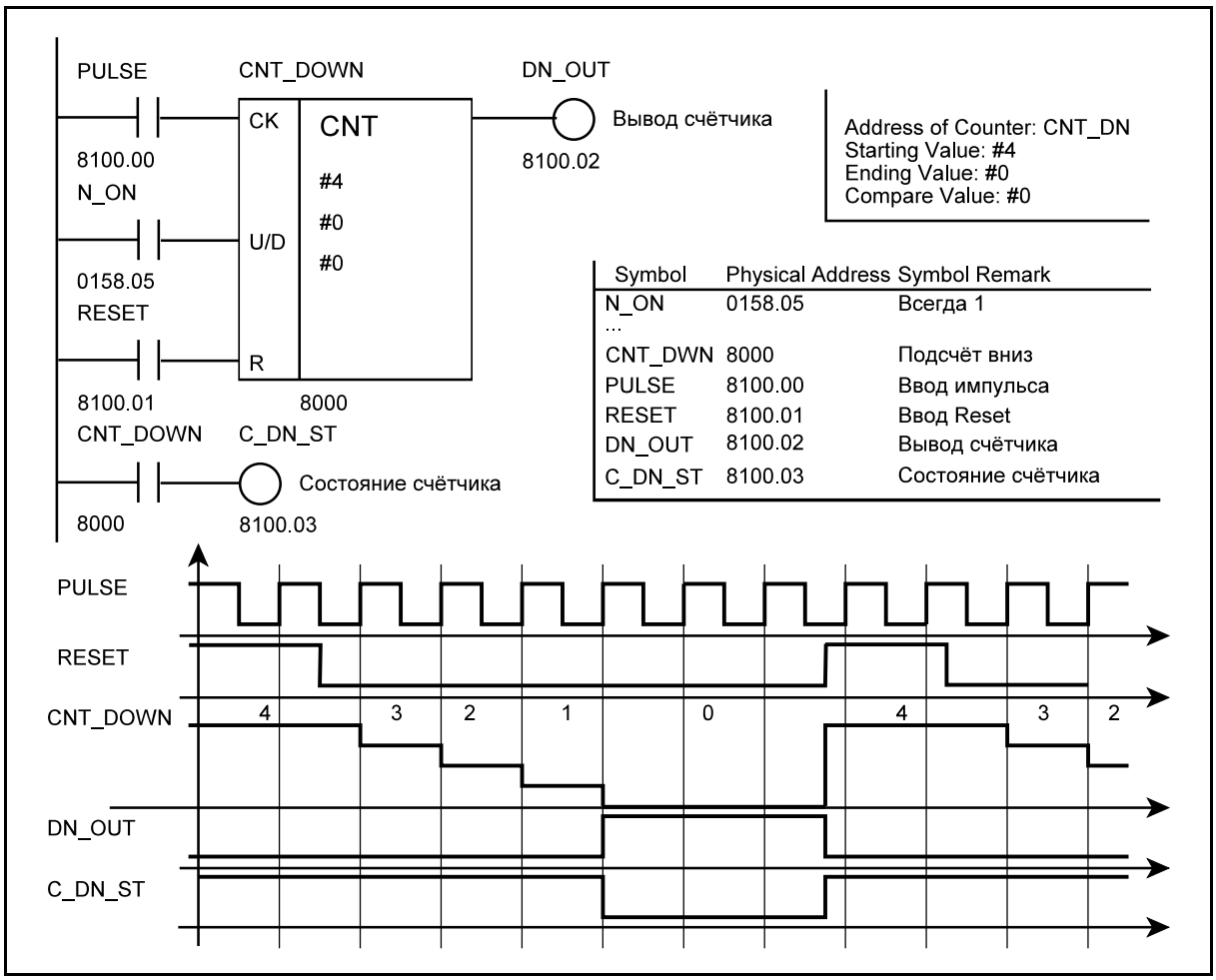
Замечание.

6.5.1 Простой счётчик CNT

Простой счётчик считает до тех пор, пока достигает значение, заданное параметром Ending Value. После этого остановится, даже при поступлении новых импульсов на ввод CK. Счётчик можно сново задействовать с переключением ввода reset в состояние ВЕРНО, приводящее в первоначальное положение.
Ниже приводим два примера для пользования счётчиком.

Подсчёт вниз

Ввод U/D счётчика включим неподвижно в 1 с использованием сигнала N_ON, указывающего всегда состояние ВЕРНО, чтобы выполнил подсчёт вниз.
Счётчик запускается под действием сигнала RESET с 4, так как значение параметра Starting Value 4, и считает до 0, так как параметр Ending Value 0. Счёт ведётся на набегающий фронт ввода CK.
Вывод счётчика попадает в состояние ВЕРНО, если его содержание 0, так как пареметр Compare Value 0.
Под действием сигнала RESET счётчик снова запускается.
На регистр, заданный по адресу Address of Counter можно ссылаться и проверкой условий: если значение регистра ≠0, то значение условия будет ВЕРНО, haесли =0 - то значение условия ФАЛЬШ.



Подсчёт вверх

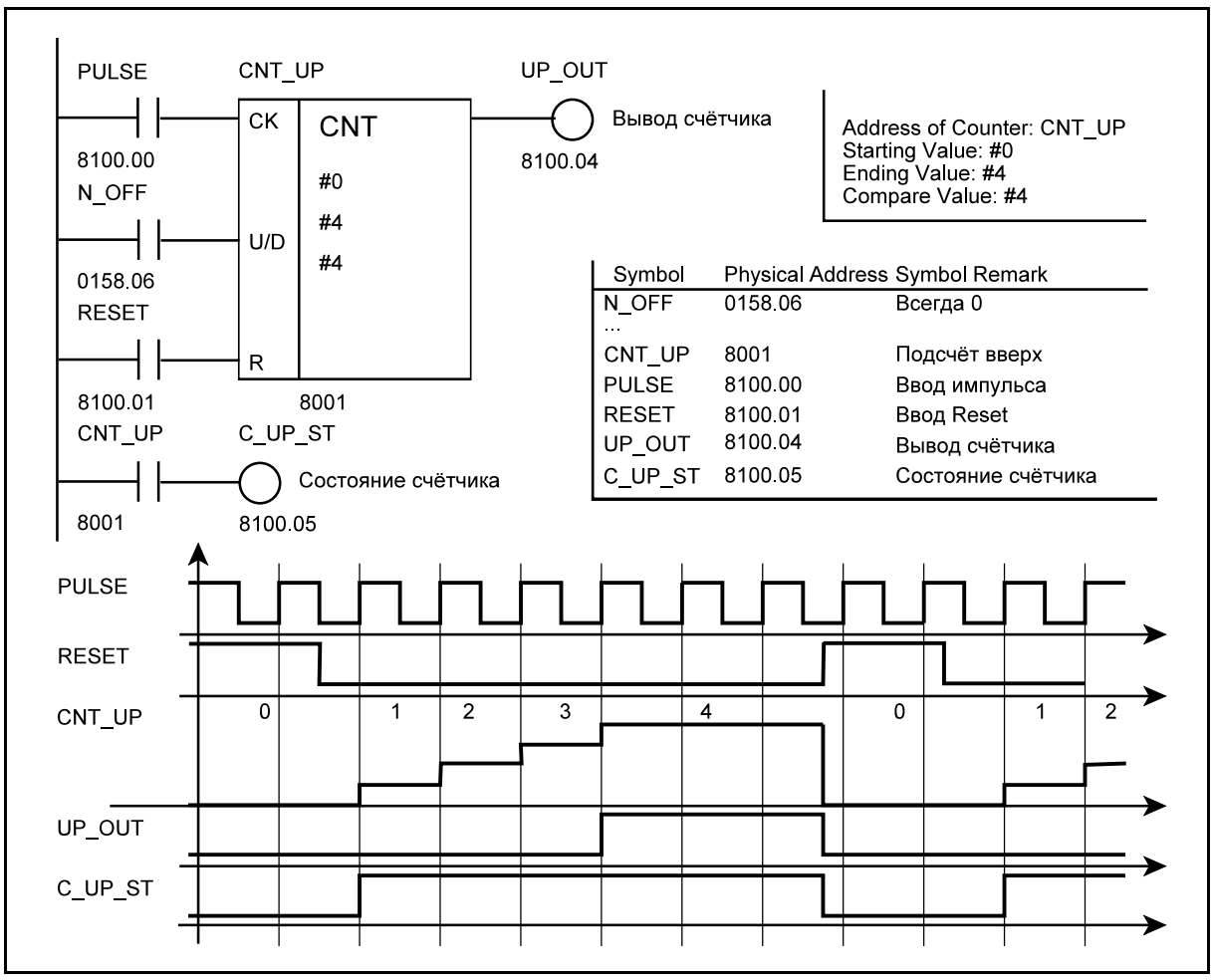
Ввод U/D счётчика включим неподвижно в 0 с использованием сигнала N_OFF указывающего всегда состояние ФАЛЬШ, чтобы выполнил подсчёт вверх.

Счётчик запускается под действием сигнала RESET с 0, так как значение параметра Starting 0, и считает до 4, так как параметр Ending Value 4. Счёт ведётся на набегающий фронт ввода CK.

Вывод счётчика попадает в состояние ВЕРНО, если его содержание 4, так как пареметр Compare Value 4.

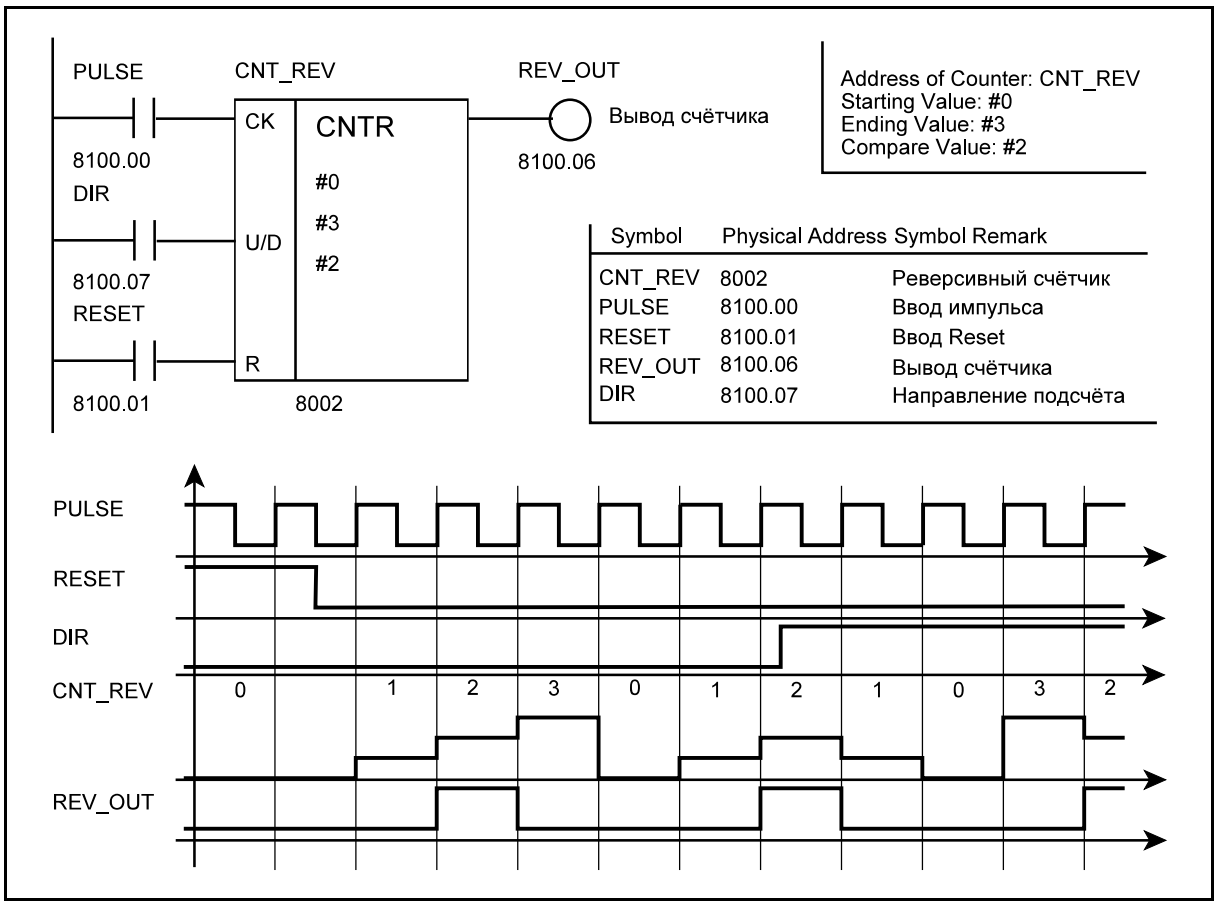
Под действием сигнала RESET счётчик снова запускается.

На регистр, заданный по адресу Address of Counter можно ссылаться и проверкой условий: если значение регистра ≠0, то значение условия будет ВЕРНО, haесли =0 - то значение условия ФАЛЬШ.



6.5.2 Реверсивный счётчик CNTR

Если реверсивный счётчик во время подсчёта вверх достигает значение, заданное параметром Ending Value, под действеим следующего импульса переварачивается и продолжает подсчёт со значения, заданного параметром Starting Value.
И наоборот, если счётчик во время подсчёта вниз достигает значение, заданное параметром Starting Value, под действеим следующего импульса переварачивается, и продолжает подсчёт со значения, заданного параметром Ending Value.
С переключением ввода reset (R) счётчика, приводящего в исходное положение, в состояние ВЕРНО, он примет значение, определённое параметром Starting Value.
Вывод счётчика управляется значением, заданным параметром Compare Value. Если значение счётчика совпадает параметром, его вывод попадает в состояние ВЕРНО.
На рисунке ниже видна работа реверсивного счётчика:



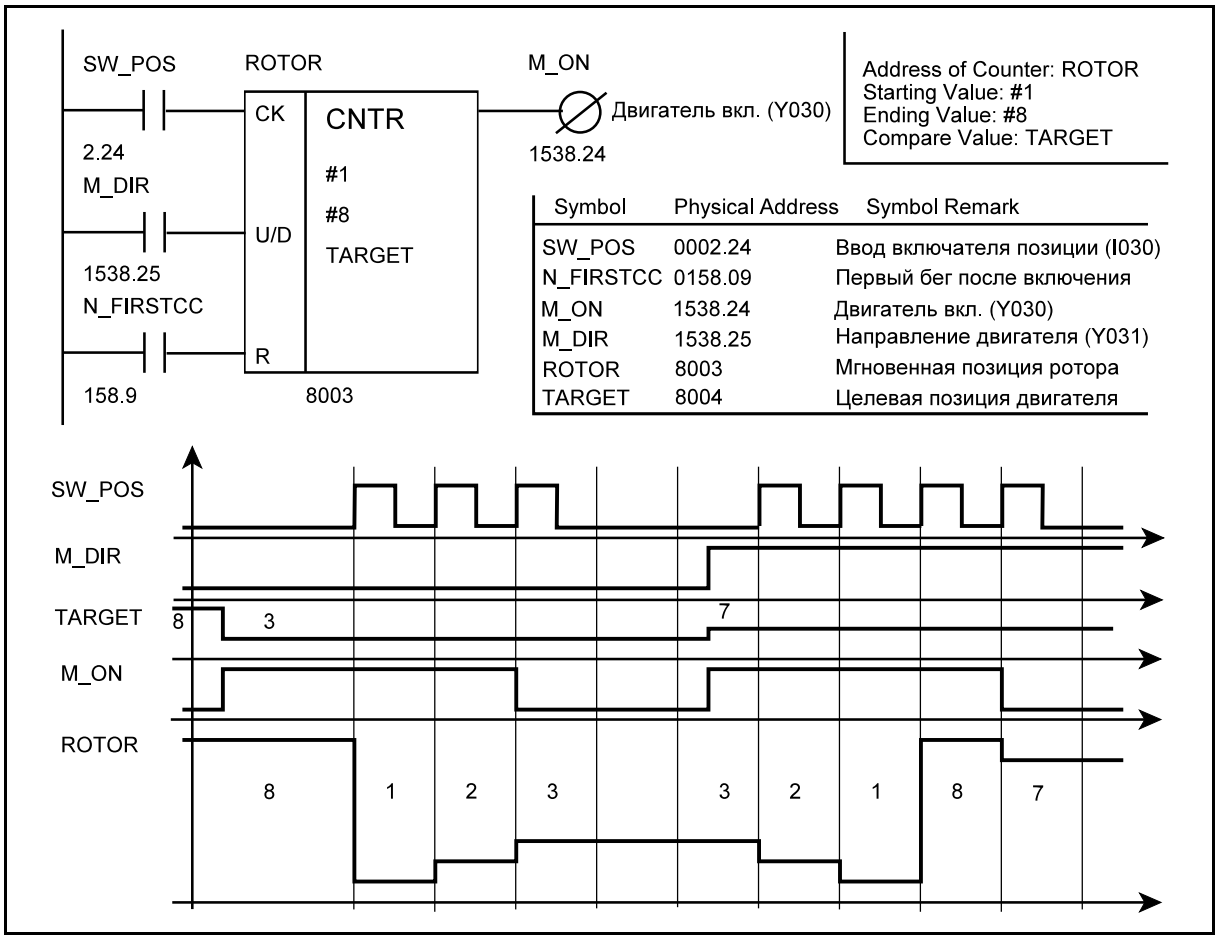
Пример для применения реверсивных счётчиков

Часто втречается задача, что необходимо перемещать вращающееся средство в определённую позицию и взять на учёт его позицию в программе PLC. Таким средством может быть револьверная головка, магазин или делительный стол.

Пусть будет наше вращающееся средство 8-ми позиционным и его позиции пронумерируем с 1 до 8. Значит, параметр Starting Value реверсивного счётчика 1, его параметр Ending Value 8. Пусть будет параметром Compare Value счётчика символический регистр TARGET, в который запишем целевую позицию. Название счётчика пусть будет ROTOR.

Если PLC получит новую команду, изменит значение регистра TARGET. Поскольку значение счётчика не совпадает с целевой позицией, его вывод попадает в состояние ФАЛЬШ, что после инвертирования включает вывод двигателя M_ON. Вывод будет включён до тех пор, пока мгновенное значение счётчика станет равным целевой позицией, заданной в регистре TARGET.

К вводу направления U/D счётчика подключаем сигнал включателя M_DIR, определяющий направление вращения двигателя, а к вводу часов CK - ввод включателя, показывающего позицию вращающегося средства.



6.6 Команда для управления вращением ROT

Команда ROT может подсчитать на основании заданных вводных параметров и соответственно установленных вводов, что в какое направление и на сколько шагов надо повернуть вращающееся средство (например:револверную головку, магазин или делительный стол), чтобы попасть в желаемую позицию.

Вводы и вывод команды ROT

Команда ROT имеет 5 вводов. Эти следующие:

– **Ввод EN**: на основании вводных параметров и вводов команд, для ***набегающего фронта*** разрешающего ввода EN подсчитает значение достигаемой позиции и устанавливает направление вращения на выводе.

– **Ввод BID**: в состоянии ФАЛЬШ ввода вдухстороннего выбора вращения BID, вычисляется значение достигаемой позиции на основании заданного направления на вводе DIR, а вывод устанавливается согласно этому. В состоянии ВЕРНО ввода вычисляется значение достигаемой позиции для вращения по самой короткой пути и выключает вывод, если направление вращения, принадлежащее к короткой пути положительным будет, далее включает, если нужное направление вращения отрицательным будет.

– **Ввод DIR:** в состоянии ФАЛЬШ ввода вычисляется значение достигаемой позиции на основании заданного направления на вводе DIR, а вывод устанавливается согласно этому: если DIR=0 и вывод будет ноль, в прочем 1.

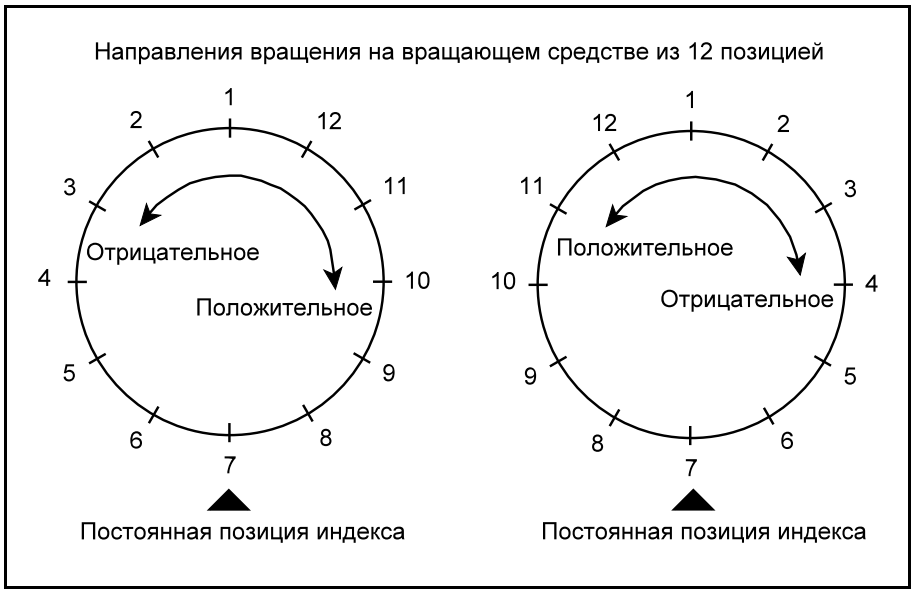
– **Ввод INC**: если значение ввода ФАЛЬШ, достигаемая позиция вычисляется в абсолютное значение, если ВЕРНО - то инкрементально, то есть вычистяется, что на сколько шагов надо повернуть средство.

– **Ввод PRE**: если значение ввода ФАЛЬШ, достигаемая позиция совпадает целевой позицией в положении INC=0, далее в положении INC=1 равняется разностью достигаемой целевой и актуальной позиции. В состоянии ввода ВЕРНО достигаемой позицией будет первая позиция перед целевой позицией.

Команды **ROT**

– **Вывод** показывает, что в каком направлении надо вращать средство в зависимости от состояния вводов BID и DIR:
если вывод ФАЛЬШ - в положительное направление,
если вывод ВЕРНО - в отрицательное направление.

Если команда не выполнима, командой устанавливается сообщение об ошибке FL_ER и вывод команды и значение рагистра Position to Go остаётся неизменным!



Вводные параметры команды ROT

– **Position to Go**:

Предел значения: 0...9999

Адрес регистра задаётся численно, или символично. Возможно задавать индексированно.

Счётчик должен дефинировать програмист в области магазина для пользователя.

Адрес того регистра, куда попадает значение ***достигаемой позиции***, вычисленное командой, является выводным регистром команды.

Если ввод BID BEPHO, достигаемая позиция вычисляется по короткой пути. Если ввод INC BEPHO, достигаемая позиция вычисляется инкрементально, то есть вычисляет, сколько шагов надо совершить до позиции. Если ввод PRE BEPHO, достигаемой позицией будет первая позиция перед целевой позицией.

– **Current Position**:

Предел значения: 0...9999

Адрес регистра задаётся численно, или символично. Возможно задавать индексированно.

Счётчик должен дефинировать програмист в области магазина для пользователя.

Адрес того регистра, который содержит ***мгновенную позицию*** вращающего средства.

– **Goal Position**:

Предел значения: 0...9999

Адрес регистра задаётся численно, или символично. Возможно задавать индексированно.

Счётчик должен дефинировать програмист в области магазина для пользователя.

Адрес того регистра, который показывает на ***целевую позицию***. Это обычно является программированным значением, например значение номера инструмента T, принятое от NC. Расчётное значение выходного регистра Position to Go совпадает значением Goal Position, если параметры INC и “1" фальшивые.

– **Starting Value**:

Предел значения: $0...2^{31}$

Число с фиксированной запятой, или ссылка на регистр. Возможно задавать индексированно.

Начальная позиция вращающего средства, её наименьшая позиция. Например 1.

– **Ending Value**:

Предел значения: $0...2^{31}$

Число с фиксированной запятой, или ссылка на регистр. Возможно задавать индексированно.

Конечная позиция вращающего средства, её наибольшая позиция. Например 12.

☞ ***Внимание!***

Если заданные адресы выходят за пределами области определения, командой установится сообщение об ошибке FL_ER и вывод команды и значение регистра. Position to Go остаются неизменными.

Пример для применения команды ROT

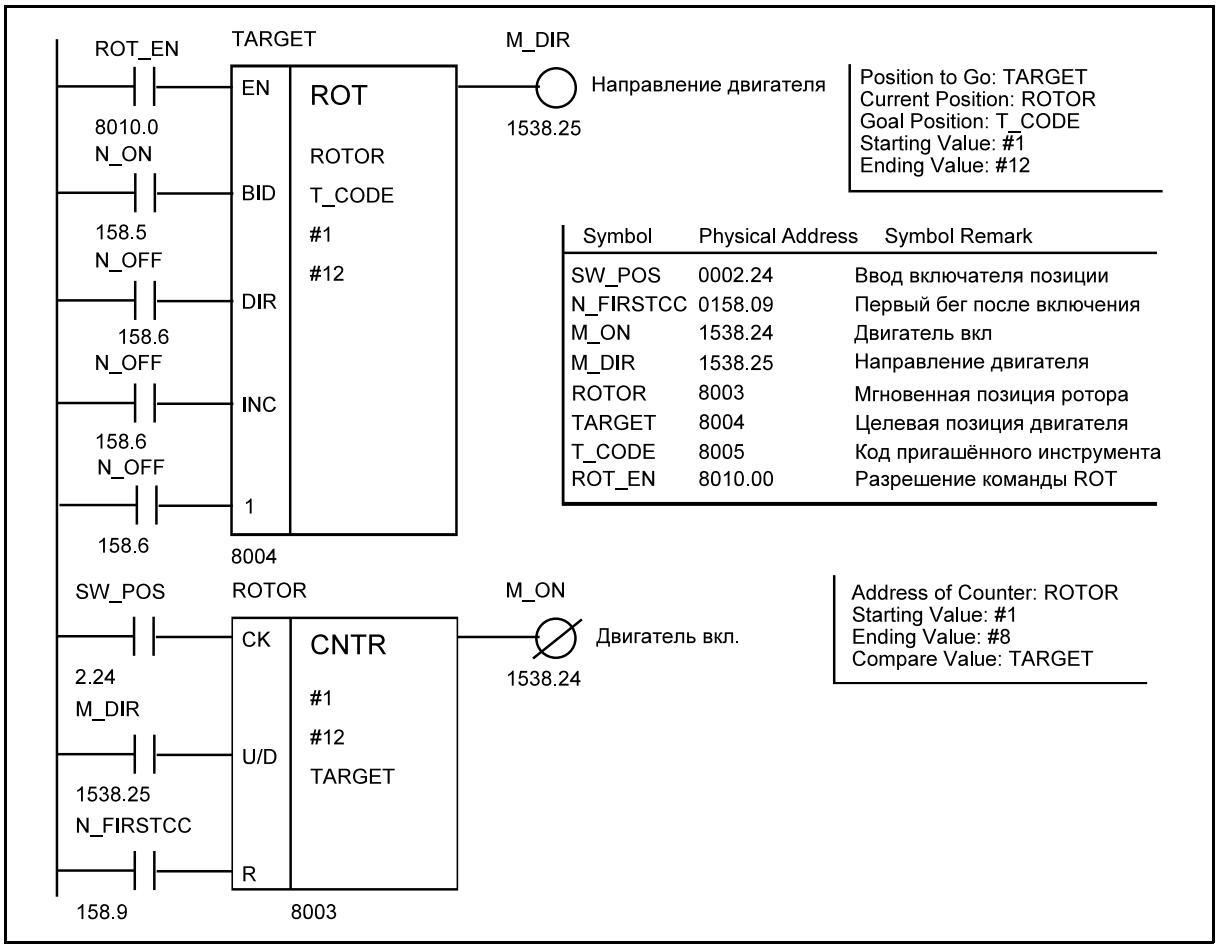
Пусть будет револьверная головка с 12-ти позицией, которую можно вращать в оба направления. Код T, поступающий от NC, попадает в регистр T_CODE.

Достигаемая позиция, вычисляемая командой ROT попадает в регистр TARGET. Мгновенная позиция получается от значения счётчика имени ROTOR. Значение целевой позиции является регистром T_CODE. Для расчёта задавали ещё начальное и конечное значения вращающего средства: #1 и #12.

Средство вращается в оба направления, значит, ввод BID будет ВЕРНО. Достигаемую позицию запросим в абсолютном значении, поэтому ввод INC будет ФАЛЬШ. Не хотим остановится перед достигаемой позиции на одну раньше, поэтому и ввод PRE будет ФАЛЬШ.

Револьверное вращение запускается, записав указатель ROT_EN в 1. Команда ROT вычисляет и запишет в регистр TARGET значение достигаемой позиции и установит соответствующее направление двигателя M_DIR.

Поскольку значение Compare Value (TARGET) счётчика имени ROTOR изменилось, его вывод будет ФАЛЬШ, это включает вывод двигателя M_ON. Двигатель начинает вращаться в направление, определённое выводом M_DIR, и будет вращаться до тех пор, пока содержание счётчика ROTOR достигает значение TARGET. Тогда вывод счётчика примет состояние ВЕРНО, выключает вывод M_ON, что остановит двигатель.



Symbol	Physical Address	Symbol Remark
SW_POS	0002.24	Ввод включателя позиции
N_FIRSTCC	0158.09	Первый бег после включения
M_ON	1538.24	Двигатель вкл
M_DIR	1538.25	Направление двигателя
ROTOR	8003	Мгновенная позиция ротора
TARGET	8004	Целевая позиция двигателя
T_CODE	8005	Код пригашённого инструмента
ROT_EN	8010.00	Разрешение команды ROT

6.7 Команды смещения и вращения

6.7.1 Регистр Shift SHTR

С помощью регистра shift можно перемещать образец бита направо, или налево в программе PLC.

Вводы и вывод команды SHTR

У регистра shift имеется один ввод
для сигнала часов (CK),
для одного направления (L/R) и
одни для данных (D).

– **Ввод CK**: Регистра shift CK перемещает образец бита, имеющийся в регистре на ***набегающий фронт*** импульса, поступающего на его ввод.
– **Ввод L/R**: Импульс, поступающий на ввод CK регистра shift, перемещает образец бита, имеющийся в регистре:
на один ***налево***, если его ввод ***L/R ФАЛЬШ***, далее
на один ***направо***, если его ввод ***L/R ВЕРНО***.
– **Ввод D**: Состояние ввода данных (D) определяет значение ***вступающего бита***. Если ввод ФАЛЬШ, то вступающий бит 0, если
ВЕРНО, то вступающий бит 1.

Состояние ввода данных D попадает
при смещении налево - на место самого нижнего 0-вого бита,
при смещении направо - на место самого верхнего 31-го бита

– **Вывод** регистра shift примет:
при смещении налево бита, выступающего из регистра - состояние его самого верхнего 31-го бита,
при смещении направо бита, выступающего из регистра - состояние его самого нижнего 0-вого бита.

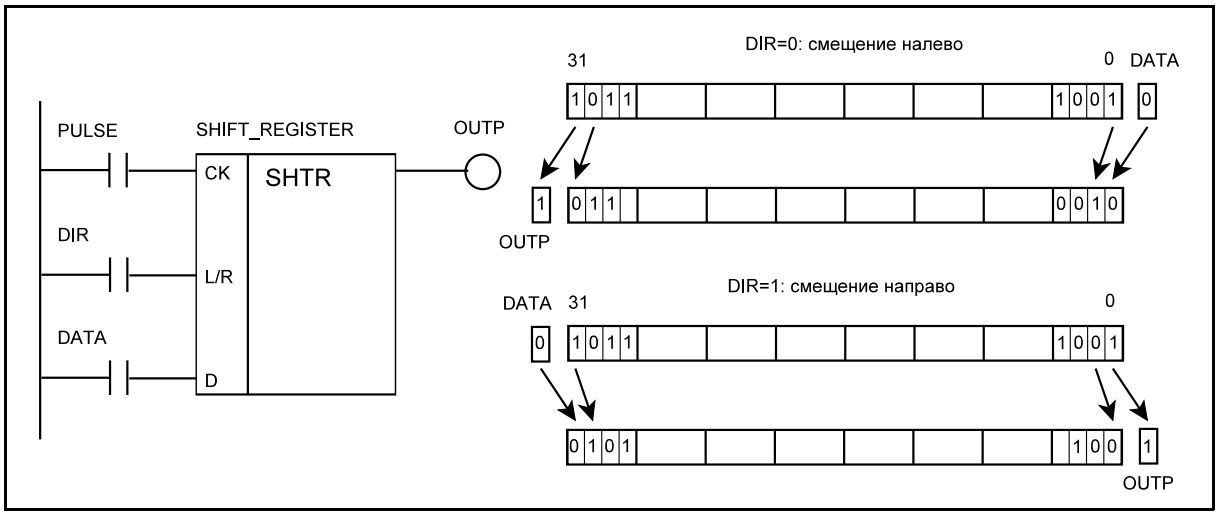
Его значение ФАЛЬШ, если выходящий бит 0, или ВЕРНО, если выходящий бит 1.

Вводный параметр регистра shift

– **Address of Shift Register**:
Предел значения: 0...9999
А Адрес регистра задаётся численно, или символично. Возможно задавать индексированно.
Регистр shift должен дефинировать програмист в области магазина для пользователя.
Адрес shift регистра указывает на адрес, содержащий мгновенное значение регистра, где регистр работает, и с какого адреса можно прочитать его мгновенное значение. То же самое относится к любому биту регистра. В любое время, хоть во время работы регистра можно изменить.

☞ ***Внимание!***
Если адрес, заданный на параметре Address of Shift Register выходит за пределами области определения, командой установится сообщение об ошибке FL_ER.

– Remark:
Замечание.



6.7.2 Арифметические смещения: ASHL, ASHR

Из чисел команд, выполняющих арифметические смещения, ***ASHL*** перемещает выделённый образец бита ***налево***, а ***ASHR - направо***.
Оба команды имеют один

статический разрешающий ввод, и один

вывод**,** подающий сообщение **о безошибочном выполнении.**.

В состоянии ВЕРНО разрешающего ввода, намеченное смещение выполняется.
Если ***разрешающий ввод ВЕРНО*** и ***операция выполнена без ошибки***, ***вывод*** команды ***принимает состояние ВЕРНО***, в противом случае его вывод будет ФАЛЬШ. Вывод можно использовать для разрешения выполнения следующей, например арифметической операции.
С помощью команд можно выделить в магазине связный блок, состоящий из n штук двойных слов, задавая начальный и конечный адрес блока. Кроме этого, можно задавать, что команда за один раз на сколько битов должна перемещать образец бита. В ходе смещения выходящие биты теряются, на освобождающееся место вступают 0-ы.
При смещении налево выходяшими битами являются верхные биты последнего двойного слова блока, которые теряются, а входящие 0-ы попадают на нижный бит первого двойного слова блока.
При смещении направо выходяшими битами являются нижные биты первого двойного слова блока, которые теряются, а входящие 0-ы попадают на нижный бит последнего двойного слова блока.

Вводные параметры арифметрического смещения

– **Starting Address**:

Предел значения: 0...9999

Начальный адрес выделяемого блока задаётся численно, или символично. Возможно задавать индексированно.

– **Ending Address:**

Предел значения: 0...9999

Конечный адрес выделяемого блока задаётся численно, или символично. Возможно задавать индексированно.

☞ ***Внимание!***

Начальный и конечний адреса должны отвечать следующему условию:

Staring Address ≤ Ending Address

Если условие выше не выполняется, или выходят за пределами области определения, командой установится сообщение об ошибке FL_ER и вывод команды примет состояние ФАЛЬШ.

Если значение начального и конечнего адреса совпадает, операция выполняется только для одного двойного слова.

– **Shift Value:**

Предел значения: $0...2^{31}$

Число с фиксированной запятой, или ссылка на регистр. Возможно задавать индексированно.

Этим параметром нужно задавать, что на сколько битов нужно перемещать образец бита, выделённый блоком.

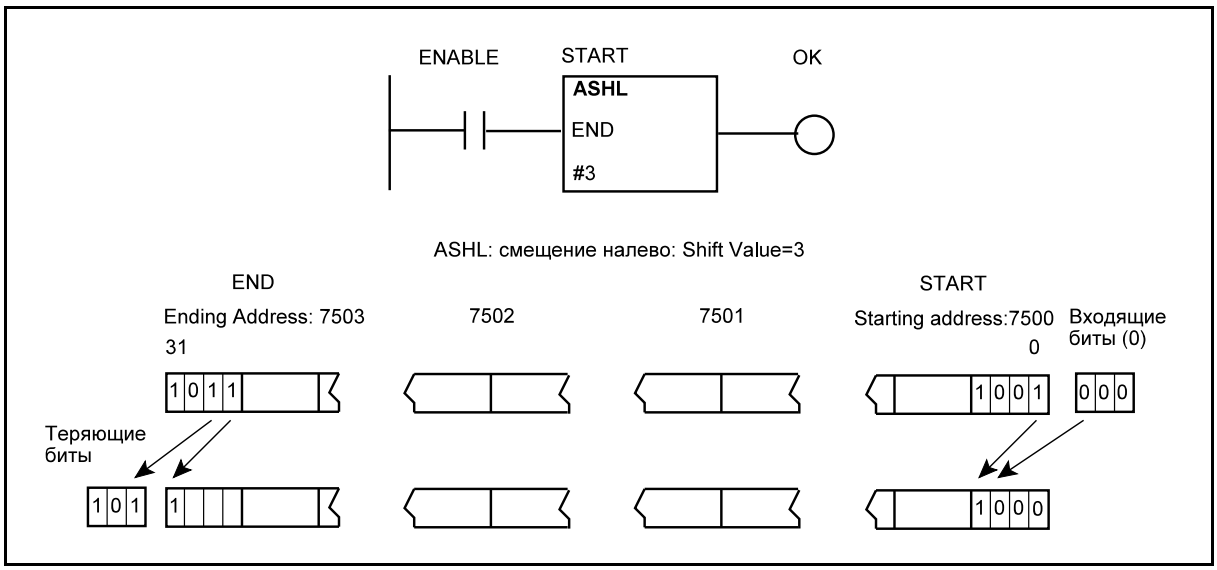
– **Output**:

В разрешенном состоянии на вывод коробки команд можнопривязать дальнейшие команды.

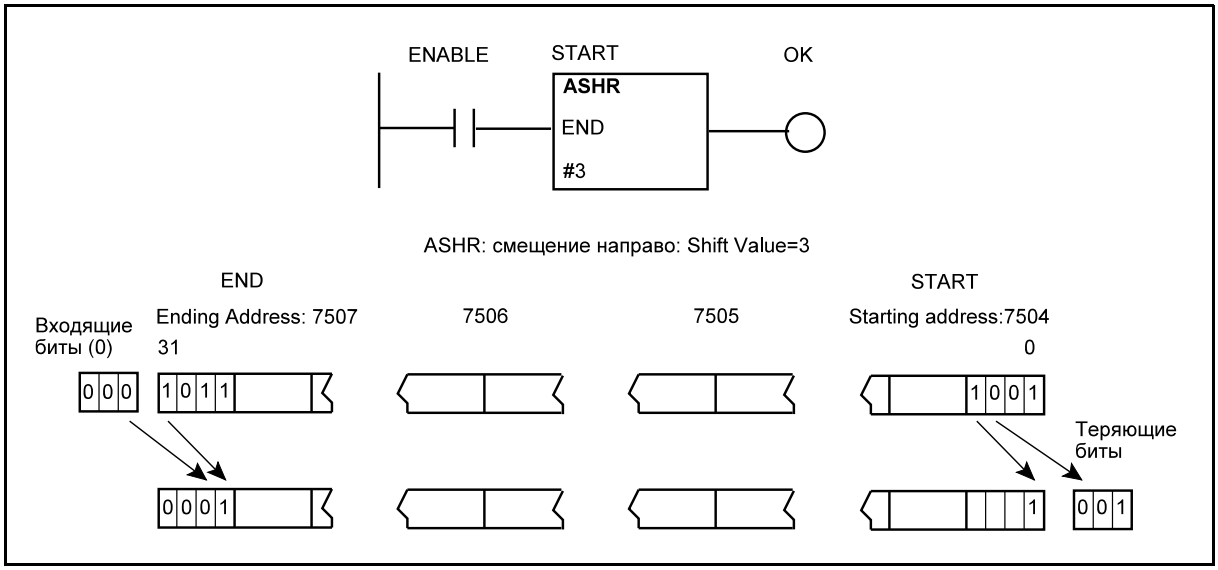
– Remark:

Замечание.

ASHL, смещение налево на произвольное количество битов, работа команды:



ASHR, смещение направо на произвольное количество битов, работа команды:



6.7.3 Арифметические вращения: ARTL, ARTR

Из числа команд, выполняющих арифметическое вращение, ***ARTL*** поворачивает выделённый образец бита ***налево***, а ***ARTR - направо***.
Оба команды имеют один

статический разрешающий ввод, и один

вывод**,** подающий сообщение **о безошибочном выполнении**.

В состоянии ВЕРНО разрешающего ввода, намеченное вращение выполняется.
Если ***разрешающий ввод ВЕРНО*** и ***операция выполнена без ошибки, вывод*** команды ***принимает состояние ВЕРНО***, в противом случае его вывод будет ФАЛЬШ. Вывод можно использовать для разрешения выполнения следующей, например арифметической операции.
С помощью команд можно выделить в магазине связный блок, состоящий из n штук двойных слов, задавая начальный и конечный адрес блока. Кроме этого, можно задавать, что команда за один раз на сколько битов должна повернуть образец бита. В ходе вращения выходящие биты вступают на другой стороне.
При вращении налево выходяшими битами являются верхные биты последнего двойного слова блока, они попадают на нижный бит первого двойного слова блока.
При вращении направо выходяшими битами являются нижные биты первого двойного слова блока, они попадают на нижный бит последнего двойного слова блока.

Вводные параметры арифметрических вращений

– **Starting Address**:

Предел значения: 0...9999

Начальный адрес выделяемого блока задаётся численно, или символично. Возможно задавать индексированно.

– **Ending Address:**

Предел значения: 0...9999

Конечный адрес выделяемого блока задаётся численно, или символично. Возможно

задавать индексированно.

☞ ***Внимание!***

Для начального и конечнего адреса должны быть верно следующее условие:

Staring Address ≤ Ending Address.

Если условие выше не выполняется, или выходят за пределами области определения, командой установится сообщение об ошибке FL_ER и вывод команды примет состояние ФАЛЬШ.

Если значение начального и конечнего адреса совпадает, операция выполняется только для одного двойного слова.

– **Rotation Value:**

Предел значения: $0...2^{31}$

Число с фиксированной запятой, или ссылка на регистр. Возможно задавать индексированно.

Этим параметром нужно задавать, что на сколько битов нужно повернуть образец бита, выделённый блоком.

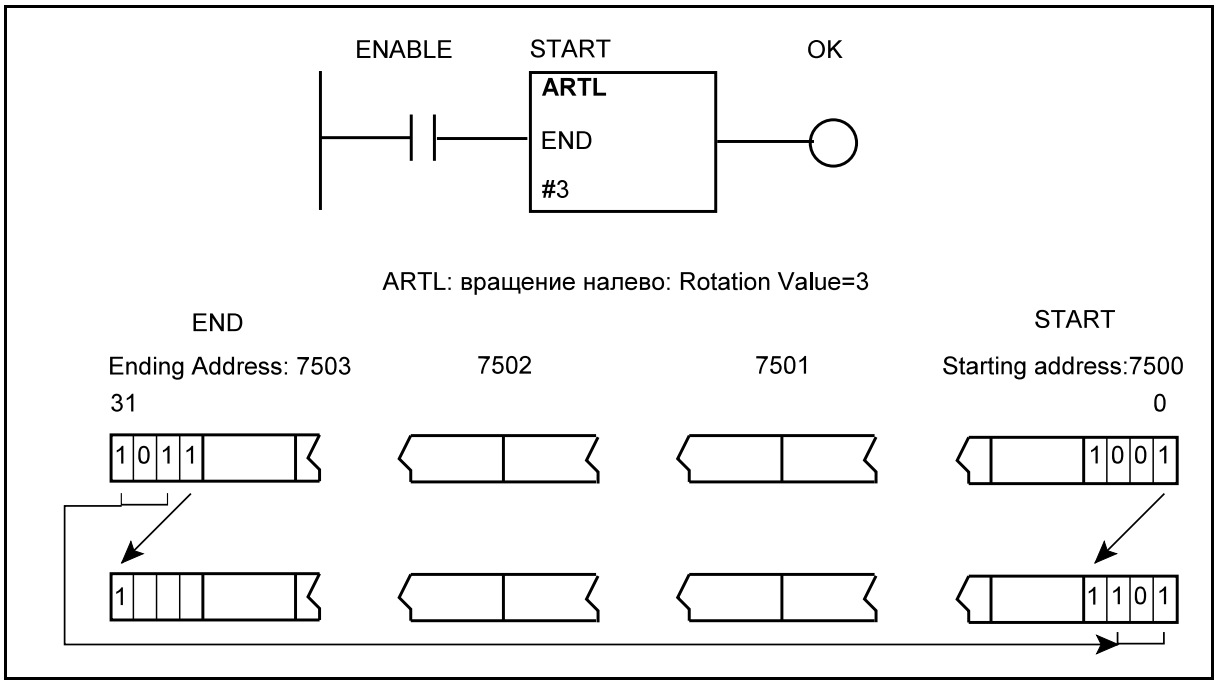
– **Output**:

В разрешенном состоянии на вывод к коробке команд можнопривязать дальнейшие команды.

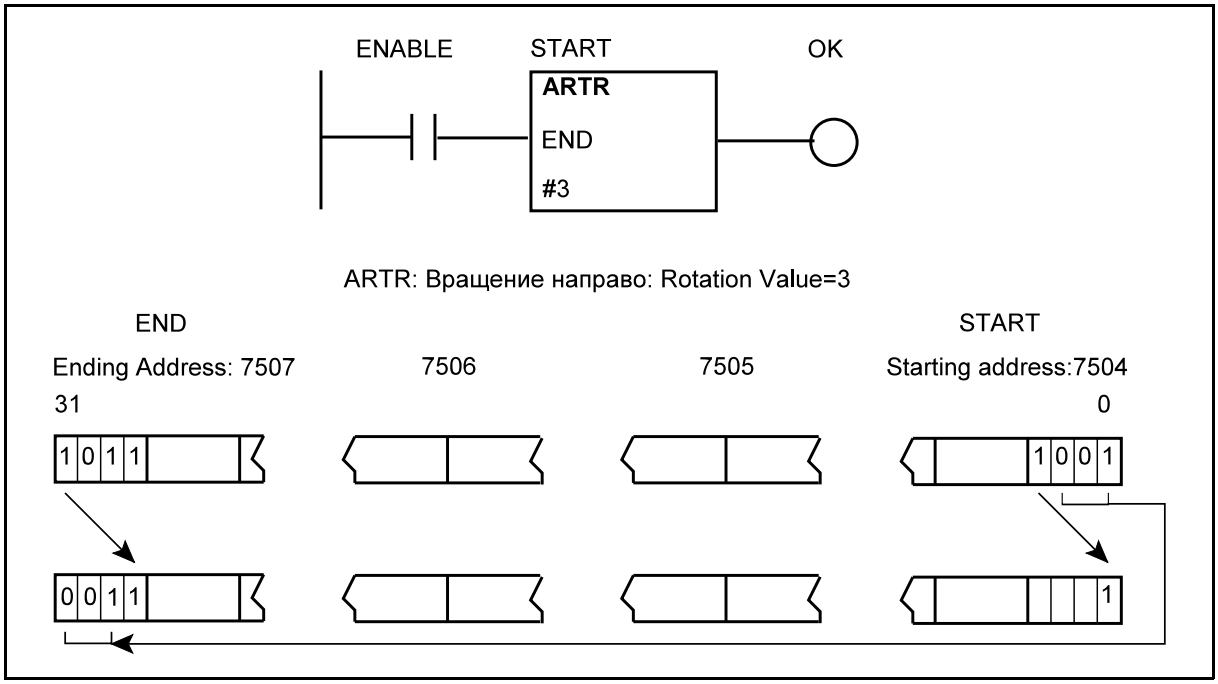
– Remark:

Звмечание.

ARTL, вращение налево на произвольное количество битов, работа команды:



ARTR, вращение направо на произвольное количество битов, работа команды:



6.8 Логические команды

Логическими командами можно выполнить логические и, или, исключительно или отвергающие операции данными из 32 битов.

6.8.1 Команда с одним операндом: NEG

Команда **NEG** выполняет отвержение регистра из 32 битов по битам. Тот бит, где 0 имеется, запишет в 1, где 1, - в 0.
Команда имеет один

статический разрешающий ввод, и один,
вывод, подающий сообщение о безошибочном выполнении.

Вывод команды будет ФАЛЬШ, если его ввод ФАЛЬШ, или если команда не выполнима, то есть если командой выводится сообщение об ошибке FL_ER!

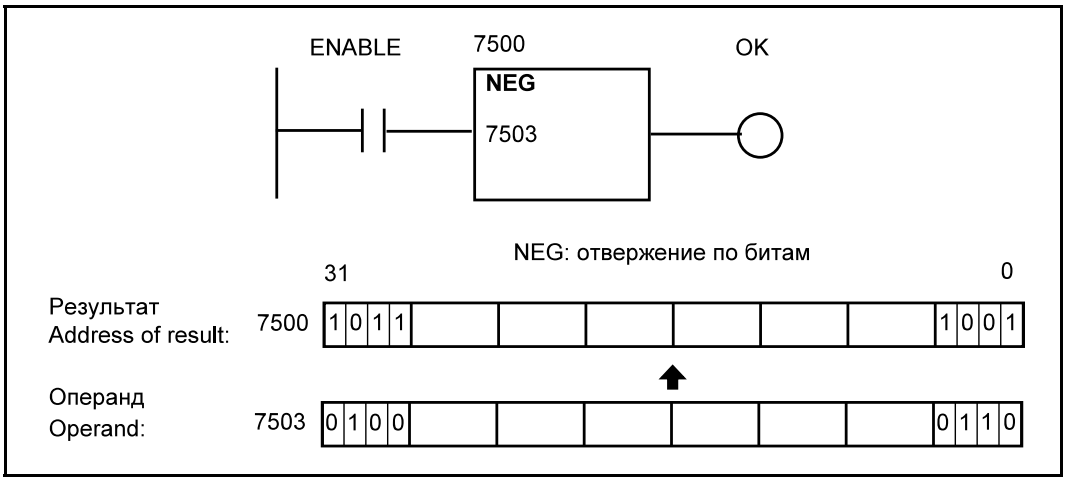
Вводные параметры команды NEG

– **Address of Result**:
Предел значения: 0...9999
Адрес результата операции задаётся численно, или символично. Возможно задавать индексированно.

– **Operand**:
Отвергаемые данные *по битам отвергает*
гексадецимальное число, заданное оператором #$, или
содержание регистра длиной DWORD
и запишет их в целевой регистр.
На операнд можно ссылаться непосредственной задачей значения, или ссылкой на регистр. Возможно задавать индексированно.

☞ ***Внимание!***
Если заданные адреса выходят за пределами области определения,вывод команды примет состояние ФАЛЬШ и выводит сообщение об ошибке FL_ER.

– **Output**:
В разрешенном состоянии на вывод коробки команд можнопривязать дальнейшие команды.



6.8.2 Команды с двумя операндами: AND, OR, XOR

Командой

AND осуществляется связь И между двумя бинарными данными из 32 битов по битам,

OR осуществляется связь ИЛИ между двумя бинарными данными из 32 битов по битам, а

XOR осуществляется связь ИСКЛЮЧИТЕЛЬНО ИЛИ между двумя бинарными данными из 32 битов по битам.

Команды имеют один

статический разрешающий ввод, и один,

вывод, подающий сообщение о безошибочном выполнении.

Вывод команды будет ФАЛЬШ, если его ввод будет ФАЛЬШ, или если команда не выполнима, то есть если командой выводится сообщение об ошибке FL_ER!

Вводные параметры команд

– **Address of Result**:

Предел значения: 0...9999

Адрес результата операции задаётся численно, или символично. Возможно задавать индексированно.

– **1st Operand**:

Он является первым операндом операции.

На первый операнд можно ссылаться *только в качестве регистра*. Возможно задавать индексированно.

– **2nd Operand**:

Он является вторым операндом операции.

На операнд можно ссылаться непосредственной задачей значения, или ссылкой на регистр. Возможно задавать индексированно.

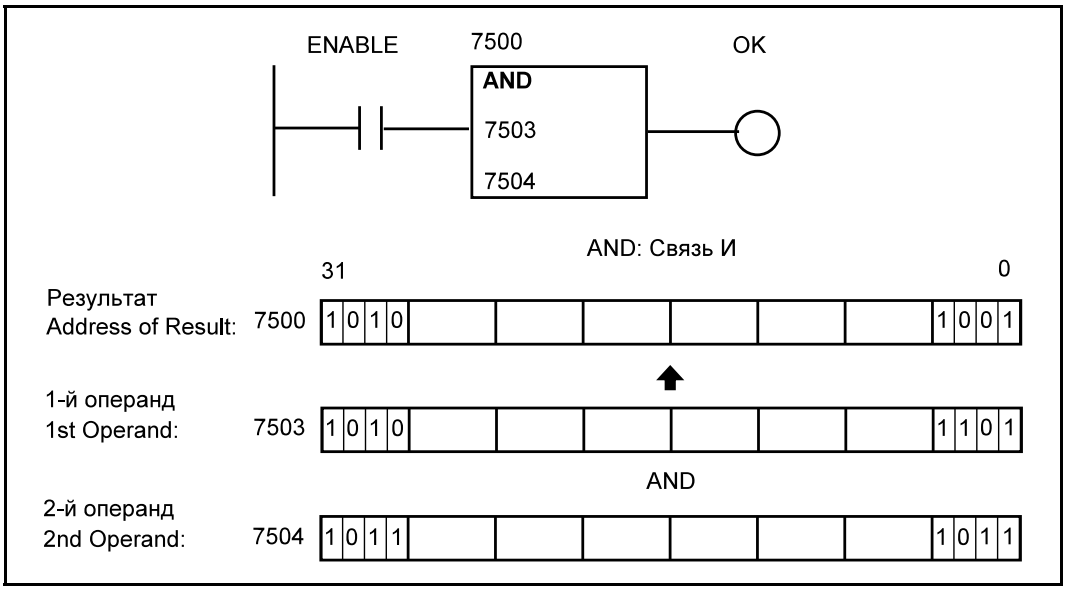
☞ ***Внимание!***

Если заданные адреса выходят за пределами области определения,вывод команды примет состояние ФАЛЬШ и выводится сообщение об ошибке FL_ER.

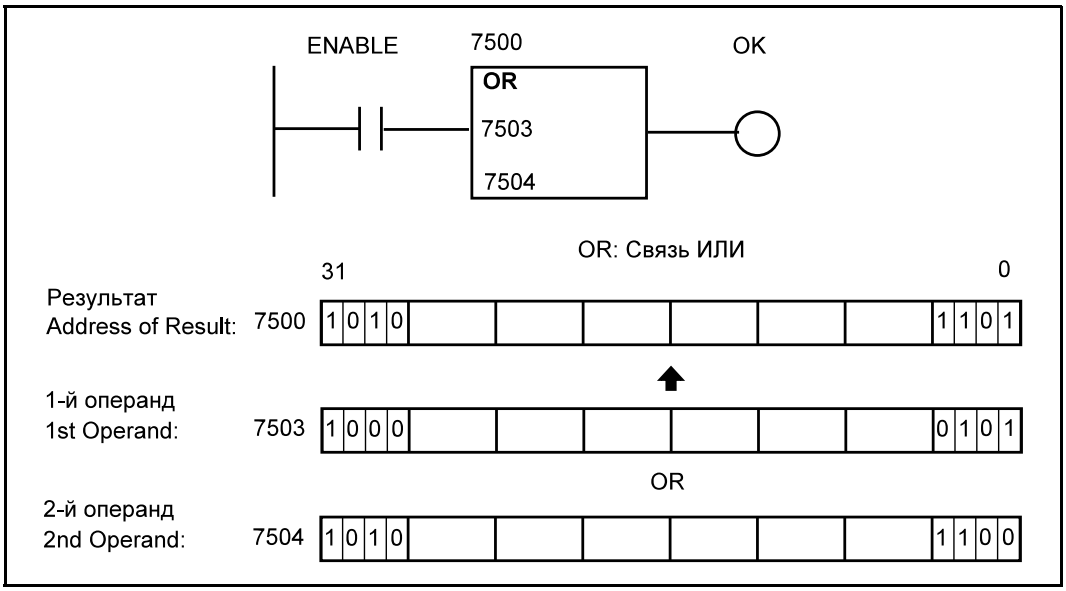
– **Output**:

В разрешенном состоянии на вывод коробки команд можнопривязать дальнейшие команды.

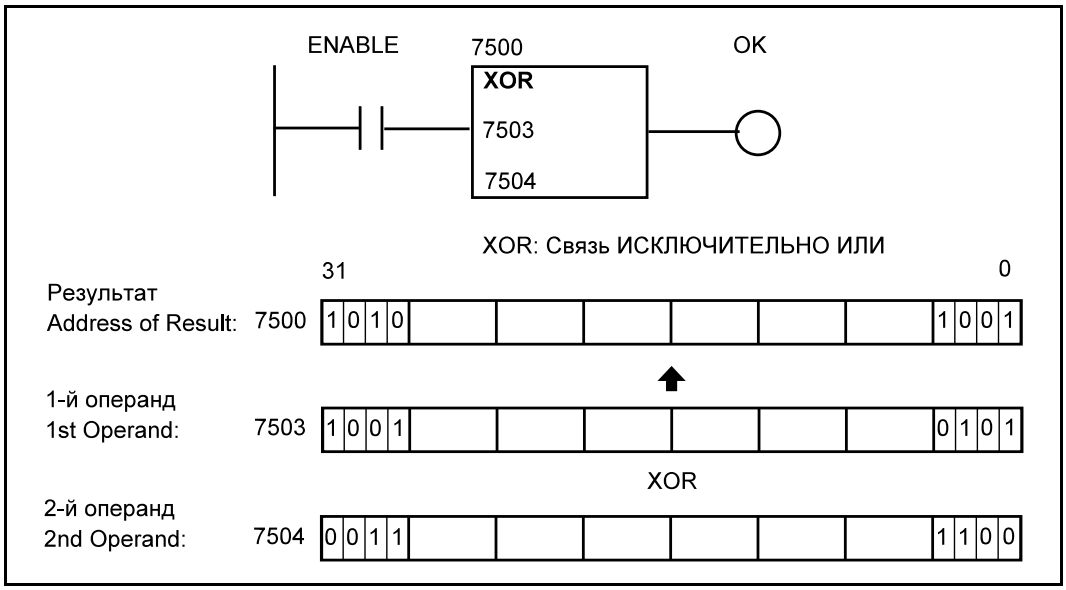
Результат по битам команды **AND** будет равно 1 тогда, если соответствующие биты обоих операндов равно 1:



Результат по битам команды **OR** будет равно 1 тогда, если соответствующий бит любого операнда равно 1:



Результат по битам команды **XOR** будет равно 1 тогда, если соответствующие биты двух операндов отличаются друг от друга.



6.9 Арифметические команды с фиксированной запятой

Арифметическими командамы являются сложение, вычитание, умножение и деление. Команды имеют один

статический разрешающий ввод, и один,

вывод, подающий сообщение об успешном выполнении.

Вывод команды будет ФАЛЬШ, если его ввод ФАЛЬШ, или если команда не выполнима, то есть если командой выводится сообщение об ошибке FL_ER!

Помимо этого, команды с фиксированной запятой обращаются следующими сигналами:

FL_UF: недополнение (underflow)

FL_OF: переполнение (overflow)

FL_CY: перенос (carry)

Вводные параметры команд

– **Address of Result**:

Предел значение: 0...9999

Адрес результата операции задаётся численно, или символично. Возможно задавать индексированно.

– **1st Operand**:

Он является первым операндом операции.

На первый операнд можно ссылаться *только в качестве регистра*. Возможно задавать индексированно.

– **2nd Operand**:

Он является вторым операндом операции.

На операнд можно ссылаться непосредственной задачей значения, или ссылкой на регистр. Возможно задавать индексированно.

– **Output**:

В разрешенном состоянии на вывод коробки команд можнопривязать дальнейшие команды.

☞ ***Внимание!***

Если заданные адреса выходят за пределами области определения,вывод команды примет состояние ФАЛЬШ и выводится сообщение об ошибке FL_ER.

6.9.1 Сложение со знаком, с фиксированной запятой, без переноса: ADD

Под понятием сложение без переноса понимается, что если перед выполнением команды ADD возник перенос в ходе предыдущей операций, то есть значениеуказателя ***FL_CY*** *равно 1, это* ***не прибавляется*** *к сумме, полученной в следствие команды ADD.*

Если

сумма попадает в регистр, заданный по адресу Address of Result,
1-ое слагаемое берётся из регистра 1st Operand, а
2 -ое слагаемое - из регистра 2nd Operand.

FL_CY записывается как указатель переноса, если в ходе сложения создаётся **перенос**, то есть сумма не вмещается на 32 битах, в прочем 0.

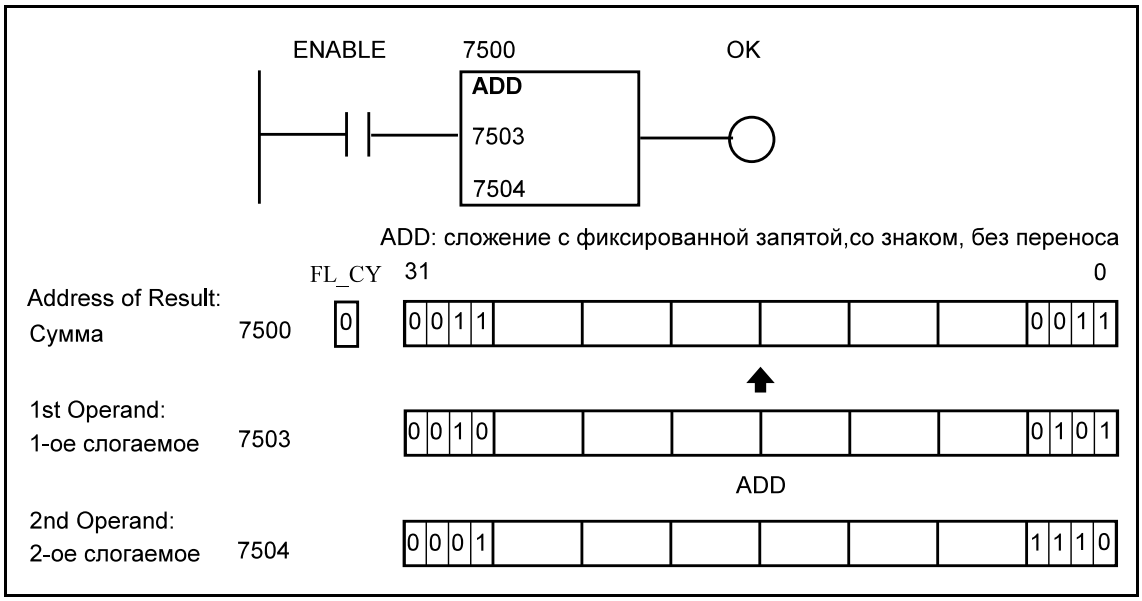
```
          7000000F
        +FC000012
FL_CY=1   6C000021
```

FL_OF записывается как указатель переполнения, если сумма, полученная после сложения двух положительных чисел (31-й бит для обоих чисел 0) попадает в область отрицательных чисел 80000000...FFFFFFFF, в прочем 0.

```
                  7FFFFFF0
                +000000FF
FL_CY=0, FL_OF=1 800000EF
```

FL_UF записывается как указатель недополнения, если сумма, полученная после сложения двух отрицательных чисел (31-й бит для обоих чисел 1) попадает в область положительных чисел 00000000...07FFFFFF, в прочем 0.

```
                  8000FC22
                +80F032A1
FL_CY=1, FL_UF=1 00F12EC3
```



6.9.2 Вычитание со знаком, с фиксированной запятой, без переноса: SUB

*Под понятием вычитание без переноса понимается, что если перед выполнением команды SUB возник перенос в ходе предыдущей операций, то есть значениеуказателя **FL_CY** равно 1, то это **не вычитается** из уменьшаемого, полученного в следствие команды SUB.*
Если

разность попадает в регистр, заданный по адресу Address of Result,
уменьшаемое берётся из регистра 1st Operand, а
вычитаемое - из регистра 2nd Operand.

FL_CY записывается как указатель переноса, если в ходе вычитания создаётся **заём**, в прочем 0:

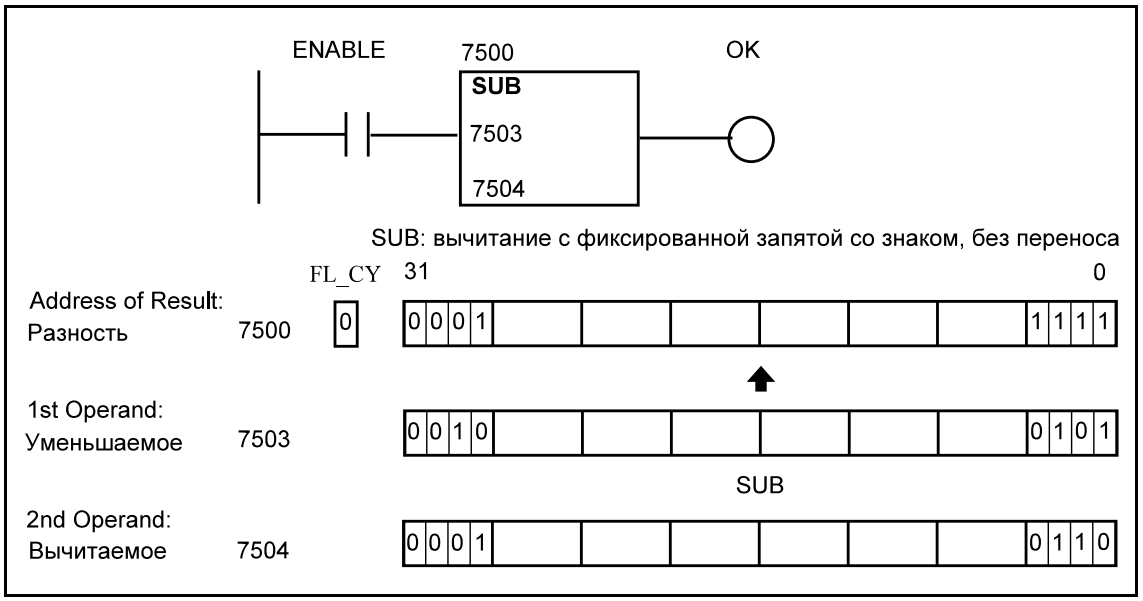
```
          2A41FB53
         -B145E681
FL_CY=1   78FC14D2
```

FL_OF записывается как указатель переполнения, если вычитав из положительного числа отрицательное число, разность попадает в область отрицательных чисел 80000000...FFFFFFFF, в прочем 0.

```
          7FFFFF00
         -FFFFF000
FL_CY=1, FL_OF=1 80000F00
```

FL_UF записывается как указатель недополнения, если вычитав из отрицательтного числа положительное число, разность попадает в область положительных чисел 00000000...07FFFFFF, в прочем 0.

```
          800000FF
         -0000A000
FL_CY=0, FL_UF=1 7FFF60FF
```



6.9.3 Умножение со знаком, с фиксированной запятой: MUL

Умножение со знаком, с фиксированной запятой MUL применяется тогда, если результат умножения (произведение)можно изображать на 32 битах (DWORD). В противом случае указатель переполнения FL_OF установится командой в 1.

Если

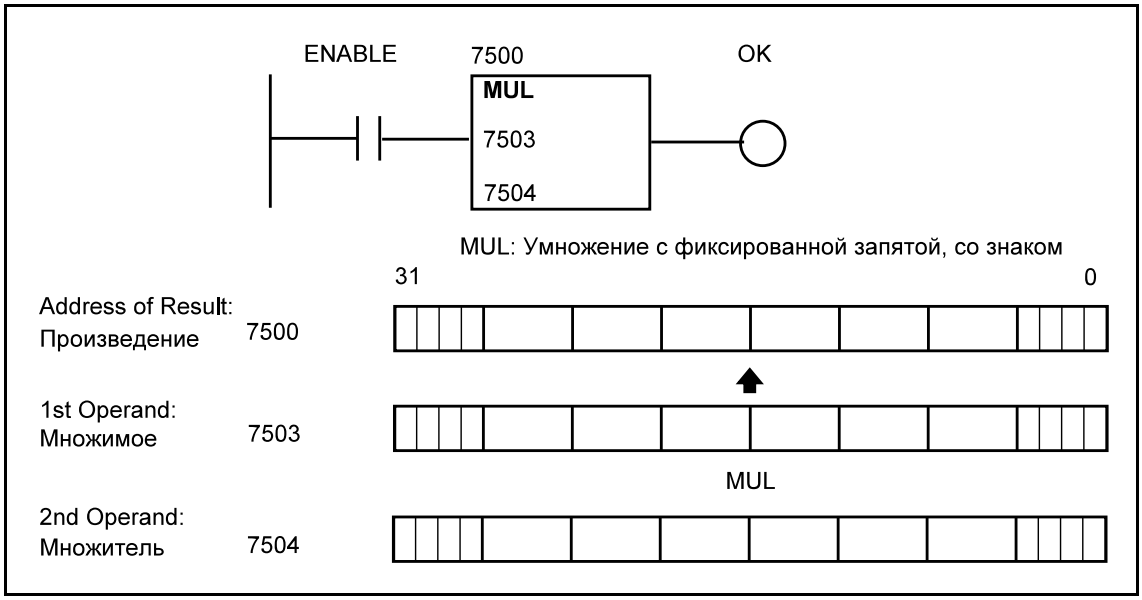
произведение попадает в регистр, заданный по адресу Address of Result,
множимое берётся из регистра 1st Operand, а
множитель - из регистра 2nd Operand.

Указатель переполнения **FL_OF** записывается, если результат нельзя изображать на 32 битах:

```
                7000000F
              ×0C000012
FL_OF=1    9400010E
```

Поскольку умножение выполняется со знаком, в случае ниже не создаётся переполнение:

```
                FFFFFFFF
              ×00000002
FL_OF=0    FFFFFFFE
```



6.9.4 Деление со знаком, с фиксированной запятой: DIV

Деление со знаком, с фиксированной запятой DIV применяется тогда, если остаток деления не нужен.

Если

частное попадает в регистр, заданный по адресу Address of Result,
делимое берётся из регистра 1st Operand, а
делитель - из регистра 2nd Operand.

Результат деления будет 0, если делитель больше делимого:

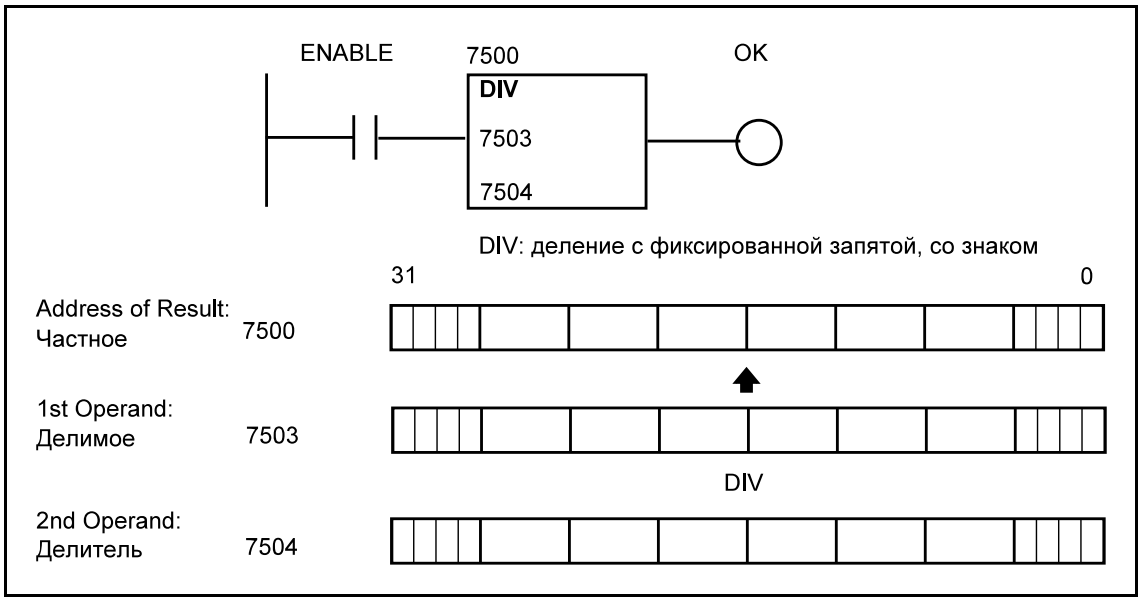
```
 0000002A
÷000000FF
 00000000
```

Результат деления будет отрицательным числом, если либо делитель, либо делимое отрицательное:

```
 00000002
÷FFFFFFFF
 FFFFFFFE
```

☞ ***Внимание!***

Если значение делителя 0, результат деления будет 0 и выводится сообщение ***FL_ER*** *в 1!*



6.10 Математические операции с плавающей запятой

Команды имеют один

статический разрешающий ввод, и один,

вывод, подающий сообщение об успешном выполнении.

Вывод команды будет ФАЛЬШ, если его ввод ФАЛЬШ, или, если команду нельзя выполнить, то есть если командой выводится сообщение об ошибке

FL_ER !

Помимо этого, команды с фиксированной запятой обращаются сообщениями ниже:

FL_UF: недополнение (underflow)

FL_OF: переполнение (overflow)

Вводные параметры команд

– **Address of Result**:

Предель значения: 0...9999

Адрес результата операции задаётся численно, или символично. Возможно задача индексированно.

– В зависимости от типа команды, возможно один или два операнда. Операцией с двумя операндами является например сложение, а с одним - извлечение вкадратного кворня.

– **Output**:

В разрешенном состоянии на вывод коробки команд можнопривязать дальнейшие команды.

☞ ***Внимание!***

Если заданные адреса выходят за пределами области определения,вывод команды примет состояние ФАЛЬШ и выводится сообщение об ошибке FL_ER.

Сообщения, которыми обращаются команды

FL_OF сообщение переполнения выводится, если абсолютное значение результата будет настолько большим, что нельзя его изображать числовым изображением двойной точности с плавающей запятой.

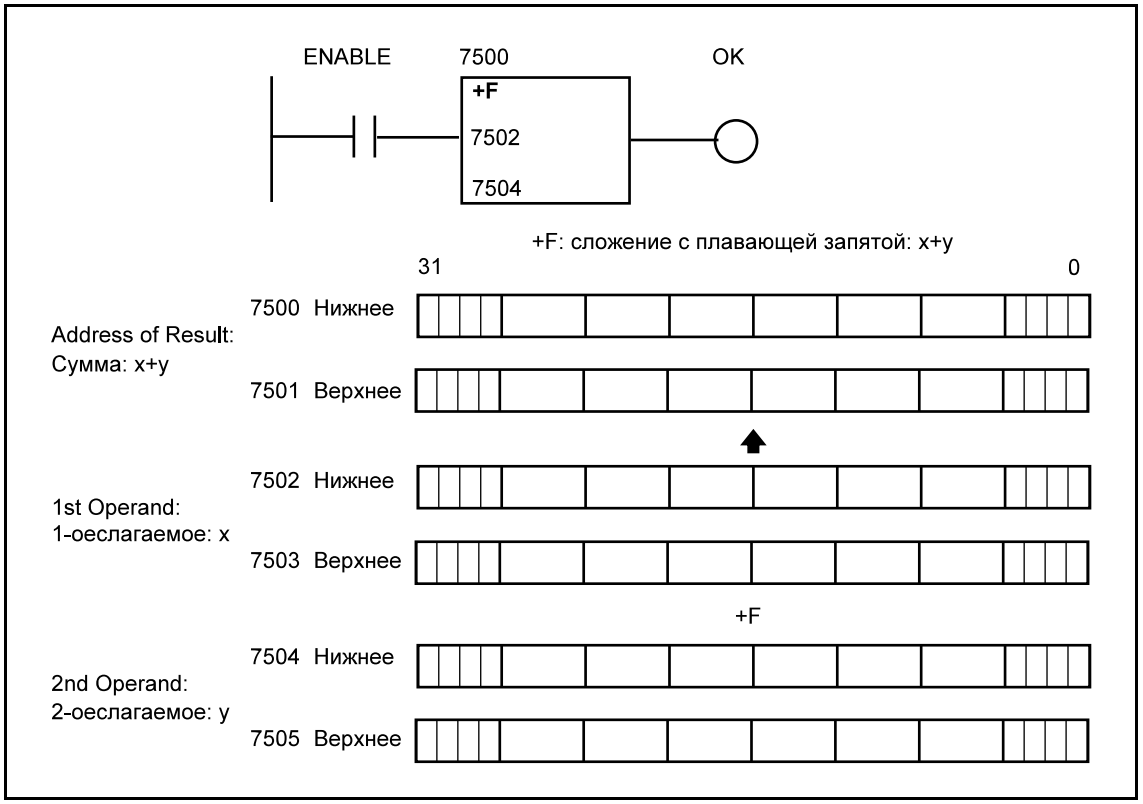
FL_UF сообщение недополнения выводится, если абсолютное значение результата будет настолько маленьким, что нельзя его изображать числовым изображением двойной точности с плавающей запятой.

6.10.1 Сложение с плавающей запятой: +F

Сумма попадает в регистр, заданный по адресу Address of Result.

– **1st Operand**
значение 1-го слагаемого

– **2nd Operand**
значение 2-го слагаемого.

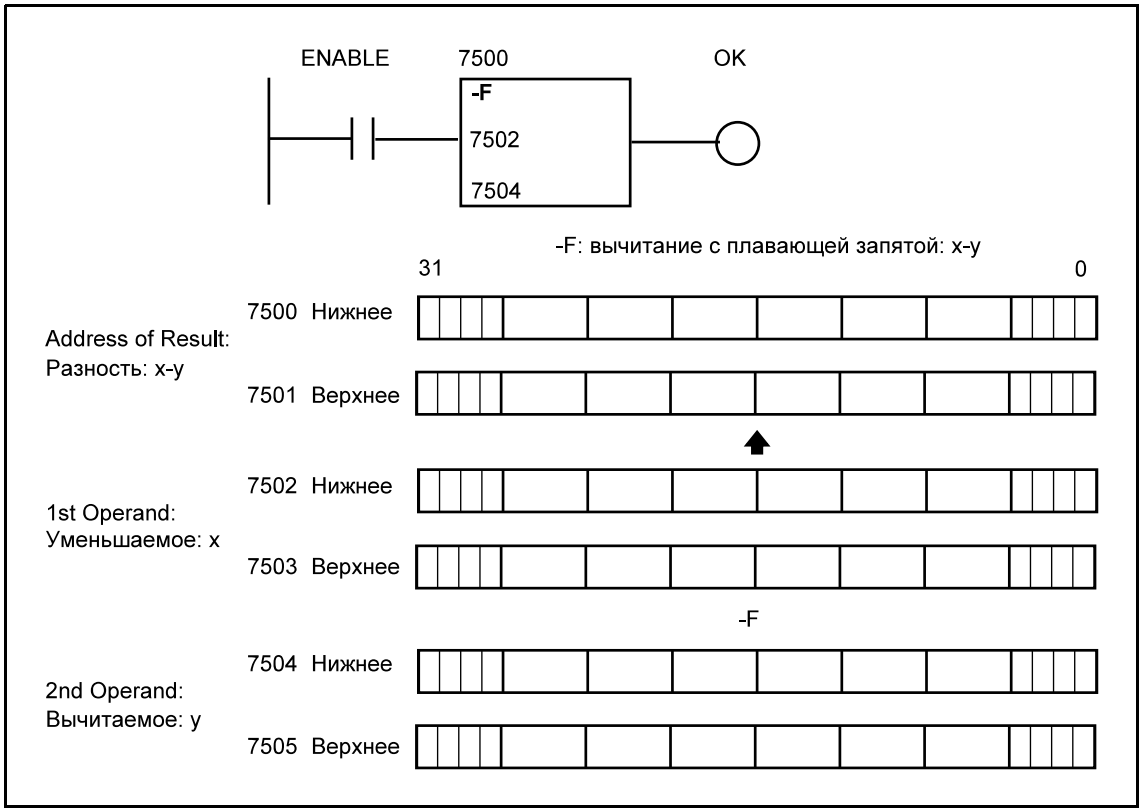


6.10.2 Вычитание с плавающей запятой: -F

Разность попадает в регистр, заданный по адресу Address of Result.

– **1st Operand**
значение уменьшаемого

– **2nd Operand**
значение вычитаемого.

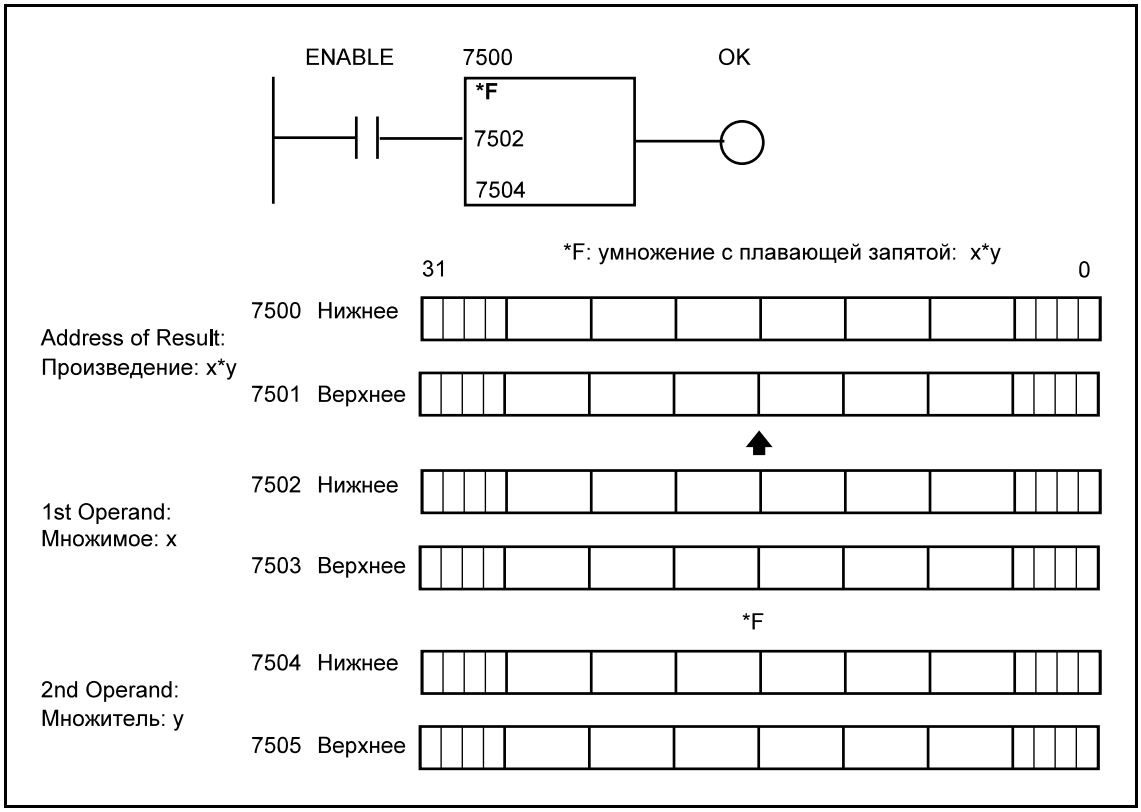


6.10.3 Умножение с плавающей запятой: *F

Произведение попадает в регистр, заданный по адресу Address of Result.

– **1st Operand**
значение множимого

– **2nd Operand**
значение множителя.

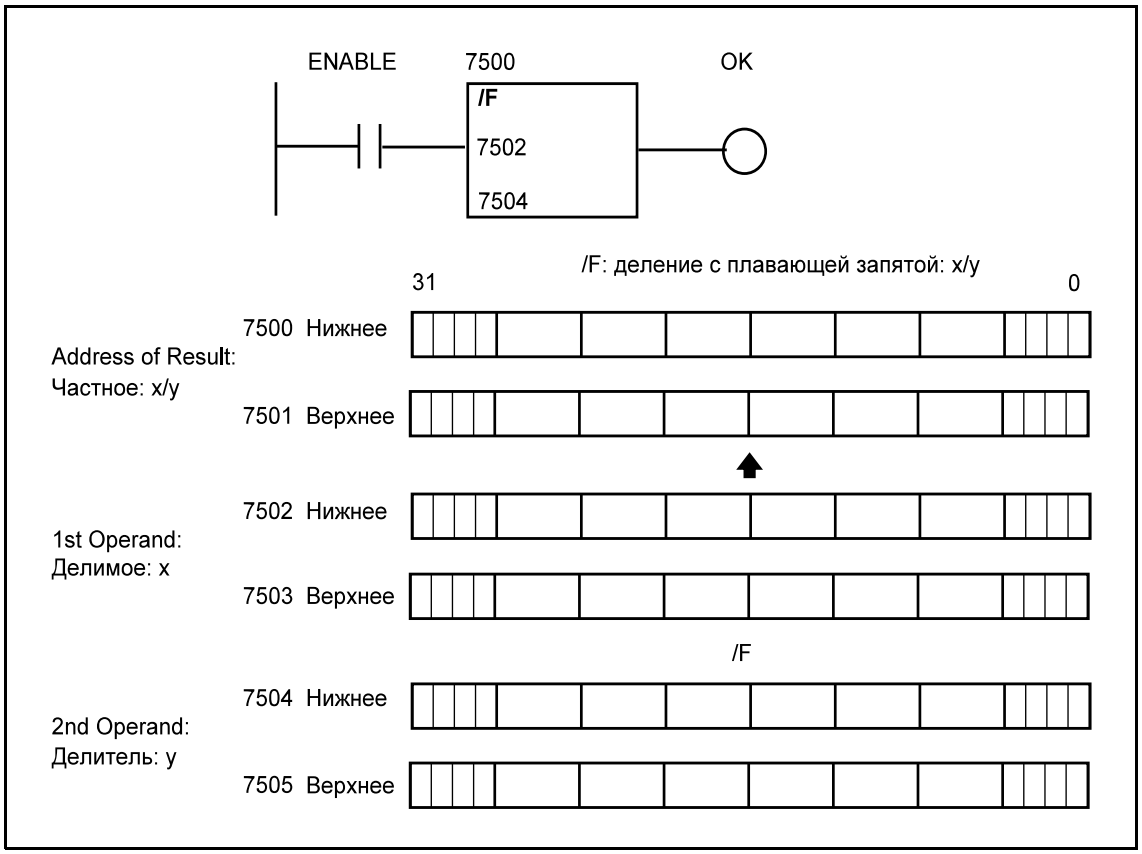


6.10.4 Деление с плавающей запятой: /F

Частное попадает в регистр, заданный по адресу Address of Result.

– **1st Operand**
значение делимого

– **2nd Operand**
значение делителя.



6.10.5 Возведение в степень: PWR

Степень попадает в регистр, заданный по адресу Address of Result.

– **Base**

значение основания степени

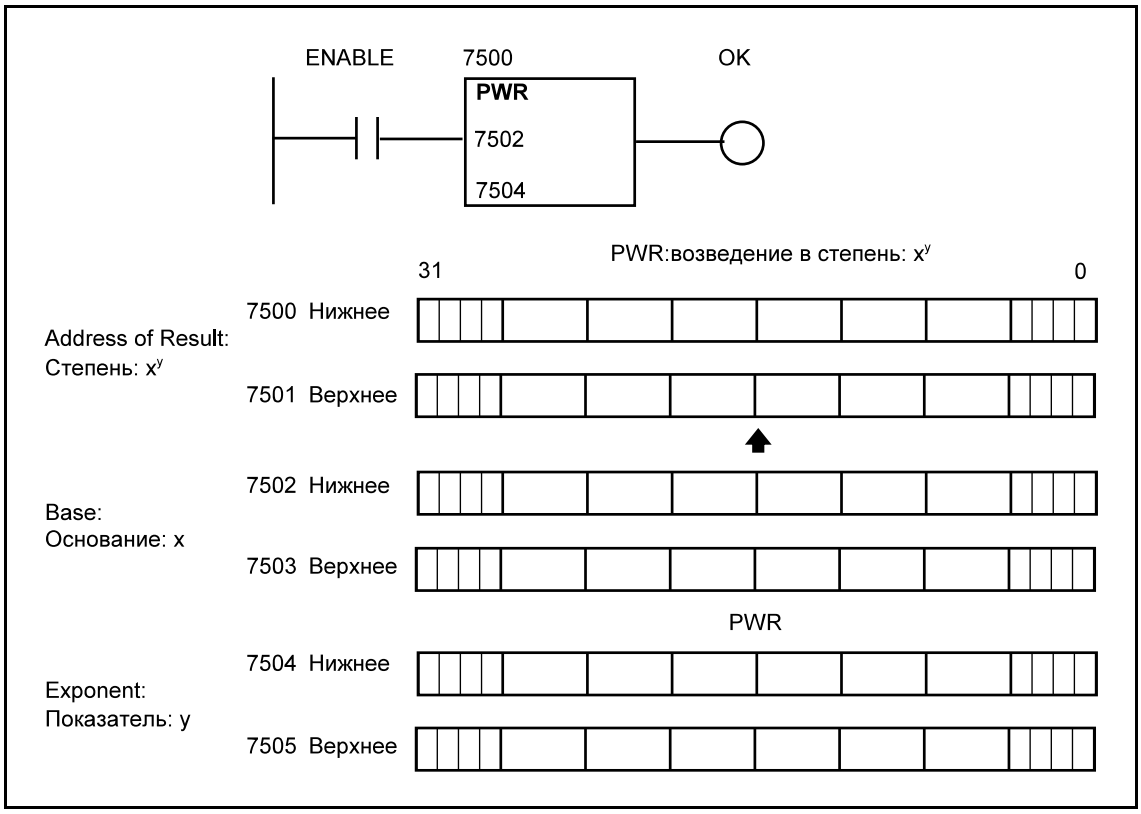
– **Exponent**

значение показателя степени.

☞ ***Внимание:*** Для значения основания степени должно быть верно следующее:

Base ≥0

Если условие выше не удовлетворяется, команда не выполняется и выводится сообщение об ошибке FL_ER=1 ВЕРНО.



6.10.6 Квадратный корень: SQRT

Квадратный кворень попадает в регистр, заданный по адресу Address of Result.

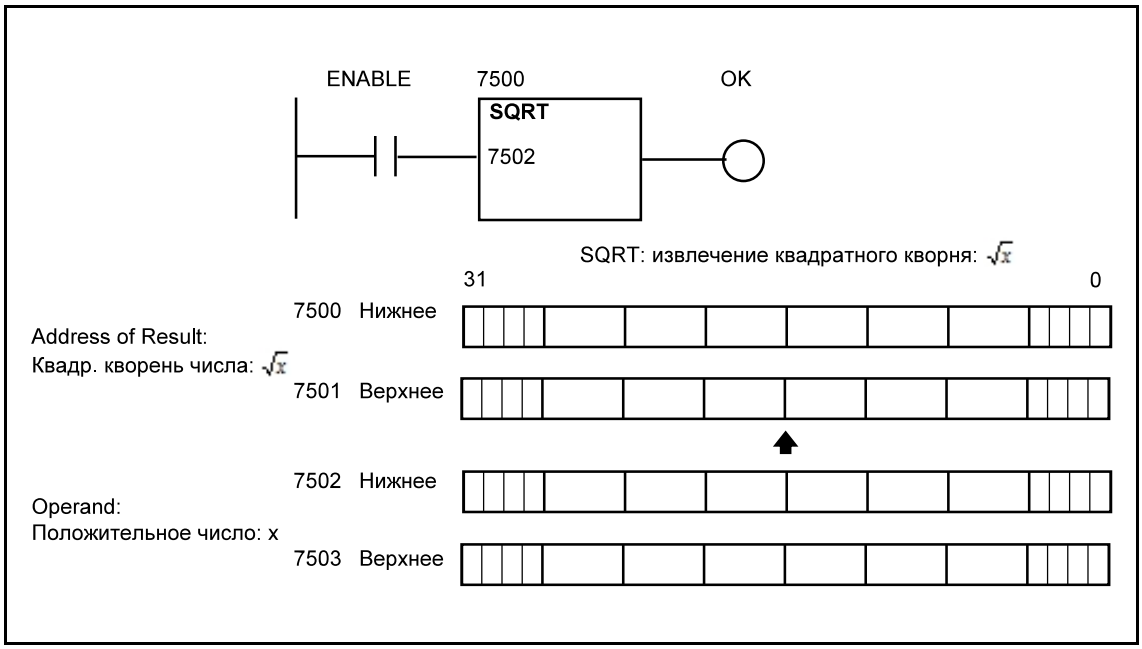
– **Operand**

подкоренное число, из которого надо извлечь квадратный кворень.

☞ ***Внимание:*** Для значения значения Операнда должно быть верно следующее:

Operand ≥0

Если условие выше не удовлетворяется, команда не выполняется и выводится сообщение об ошибке FL_ER=1 ВЕРНО.

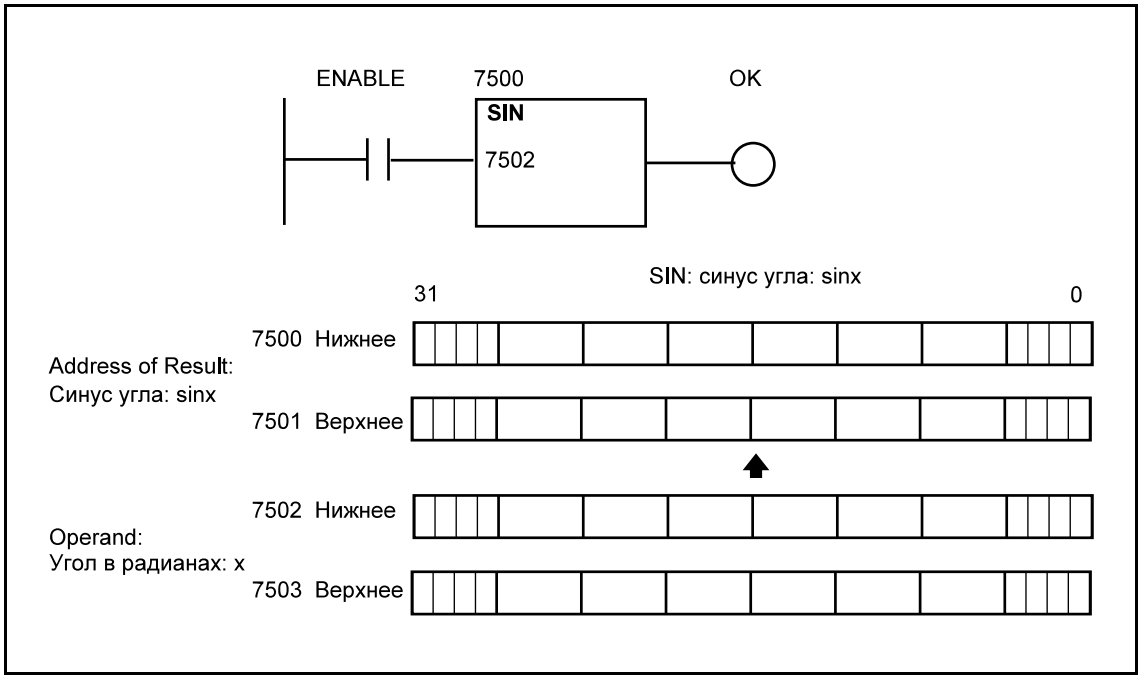


6.10.7 Синус: SIN

Синус угла попадает в регистр, заданный по адресу Address of Result.

– **Operand**

тот значение угла ***в радианах***, для которого берётся синус.

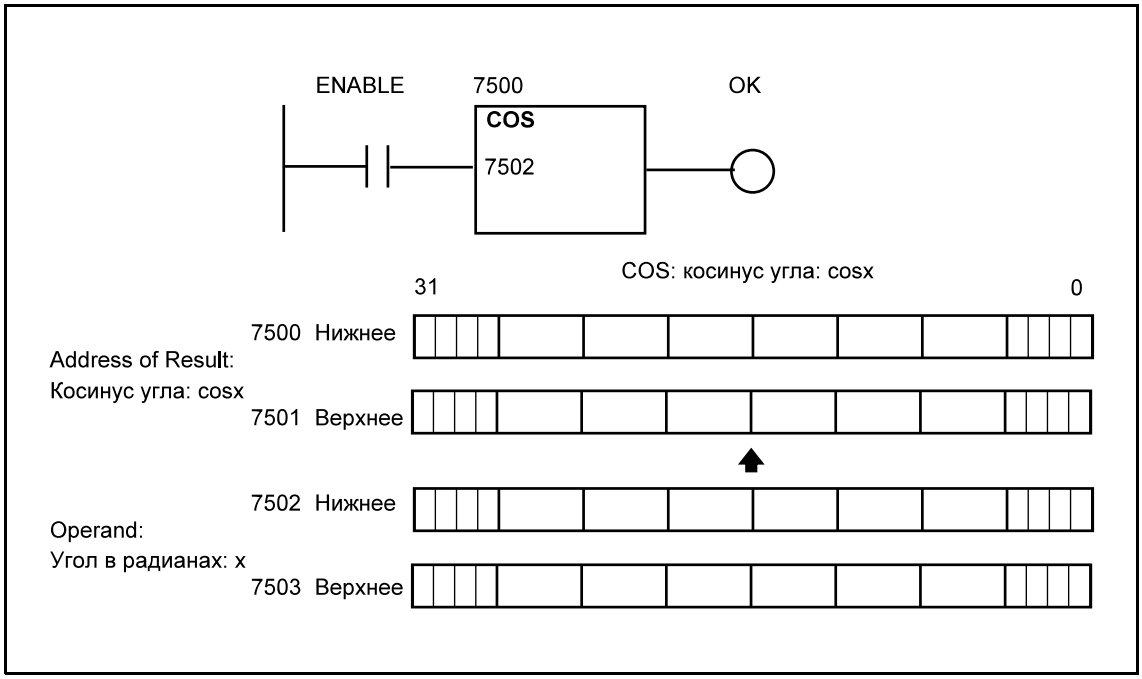


6.10.8 Косинус: COS

Косинус угла попадает в регистр, заданный по адресу Address of Result.

– **Operand**

тот значение угла ***в радианах***, для которого берётся косинус.



6.10.9 Тангенс: TAN

Тангенс угла попадает в регистр, заданный по адресу Address of Result.

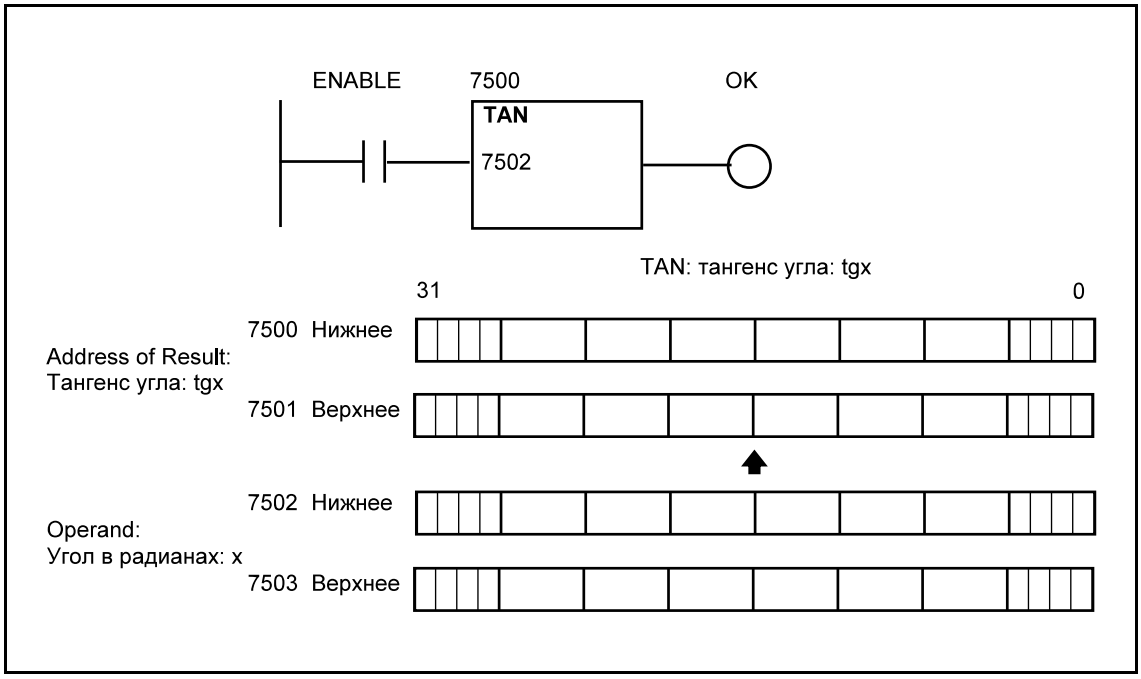
– **Operand**

тот значение угла ***в радианах***, для которого берётся тангенс.

☞ ***Внимание:*** Если значение Operand:

Operand = π/2

команда не выполняется и выводится сообщение об ошибке FL_ER=1 ВЕРНО.



6.10.10 Арксинус: ASIN

Арксинус числа попадает в регистр, заданный по адресу Address of Result ***в радианах***.

– **Operand**

это то число, для которого берётся арксинус..

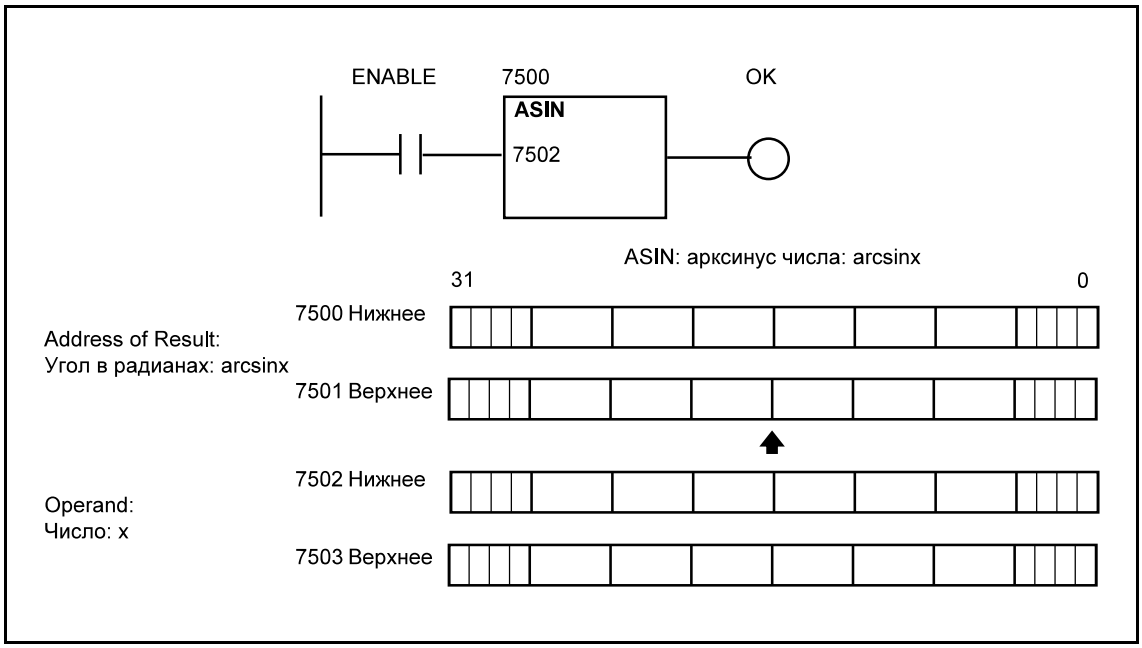
☞ ***Внимание:*** Для значения Operand должны быть верны следующие:

$-1 \leq \text{Operand} \leq 1$

Если условие выше не удовлетворяется, команда не выполняется и выводится сообщение об ошибке FL_ER=1 ПРАВДА.

Результат получится в следующей области угла:

$-\pi/2 \leq \text{Result} \leq \pi/2$



6.10.11 Арккосинус: ACOS

Арккосинус числа попадает в регистр, заданный по адресу Address of Result ***в радианах***.

– **Operand**

это то число, для которого берётся арккосинус.

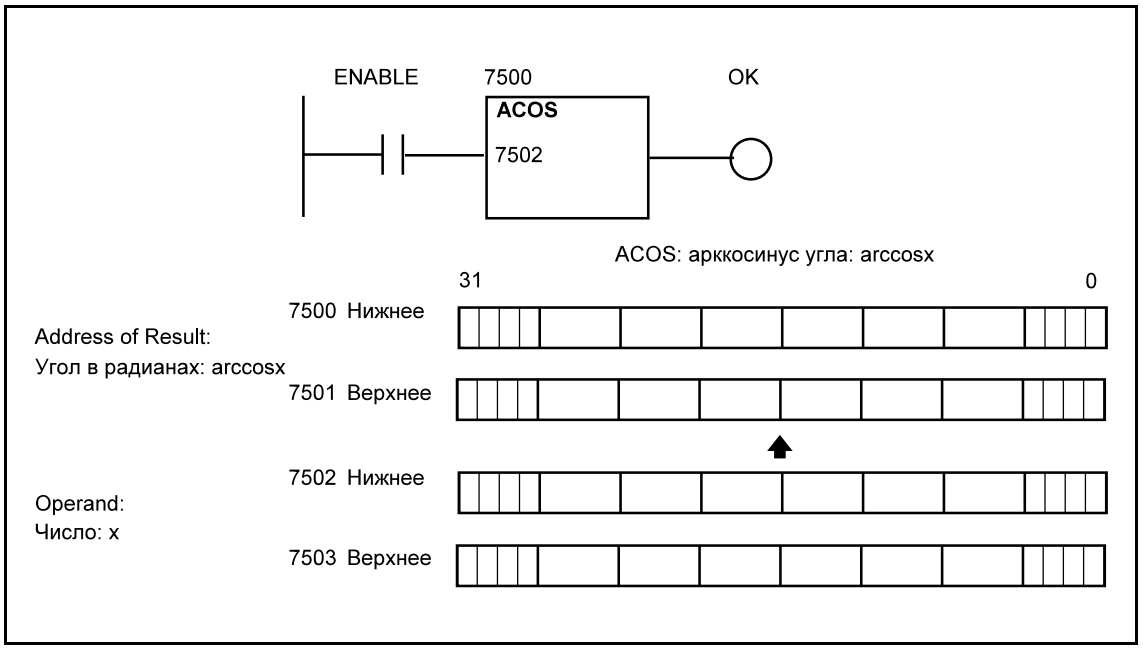
☞ ***Внимание:*** Для значения Operand должны быть верны следующие:

$-1 \leq \text{Operand} \leq 1$

Если условие выше не удовлетворяется, команда не выполняется и выводится сообщение об ошибке FL_ER=1 ВЕРНО.

Результат получится в следующей области угла:

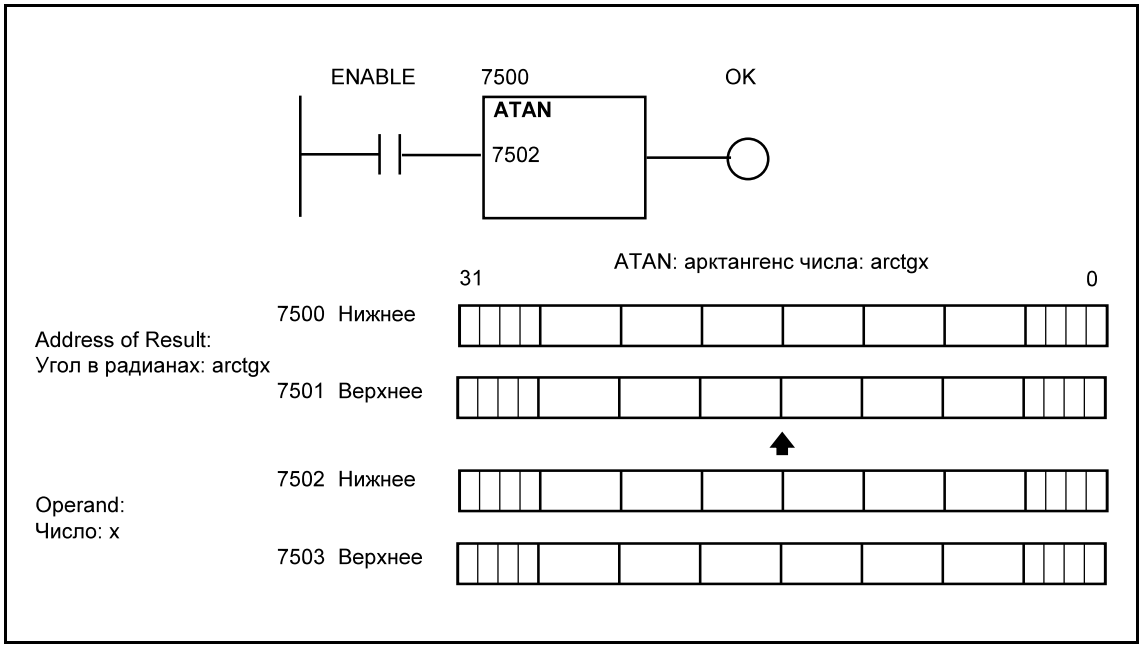
$0 \leq \text{Result} \leq \pi$



6.10.12 Арктангенс: ATAN

Арктангенс числа попадает в регистр, заданный по адресу Address of Result ***в радианах***.
– **Operand**
это то число, для которого берётся арктангенс.
Результат получится в следующей области угла:

$$-\pi/2 \leq \text{Result} \leq \pi/2$$



6.10.13 Степень с натуральным основанием (e): EXP

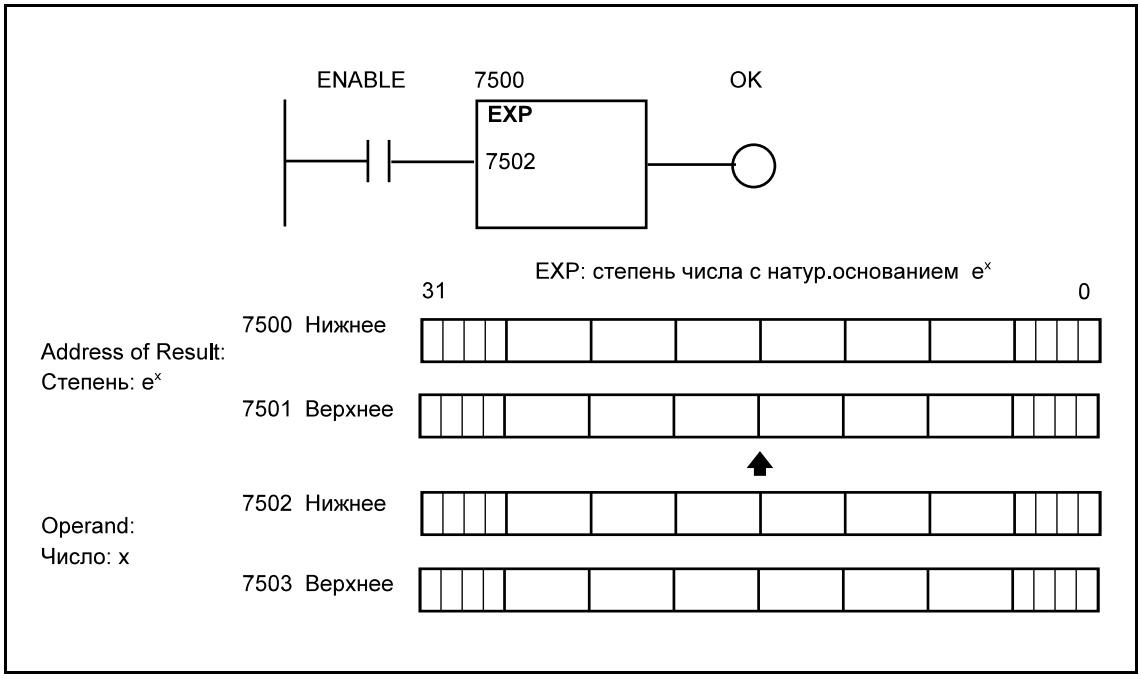
Степень числа с натуральным основанием (e=2.718282...) попадает в регистр, заданный по адресу Address of Result.

– **Operand**

это то число, для которого берётся степень с натуральным основанием.

Результат получится в следующей области:

0 < Result



6.10.14 Логарифм с натуральным основанием (e): LOG

Логарифм числа с натуральным основанием (e=2.718282...) попадает в регистр, заданный по адресу Address of Result.

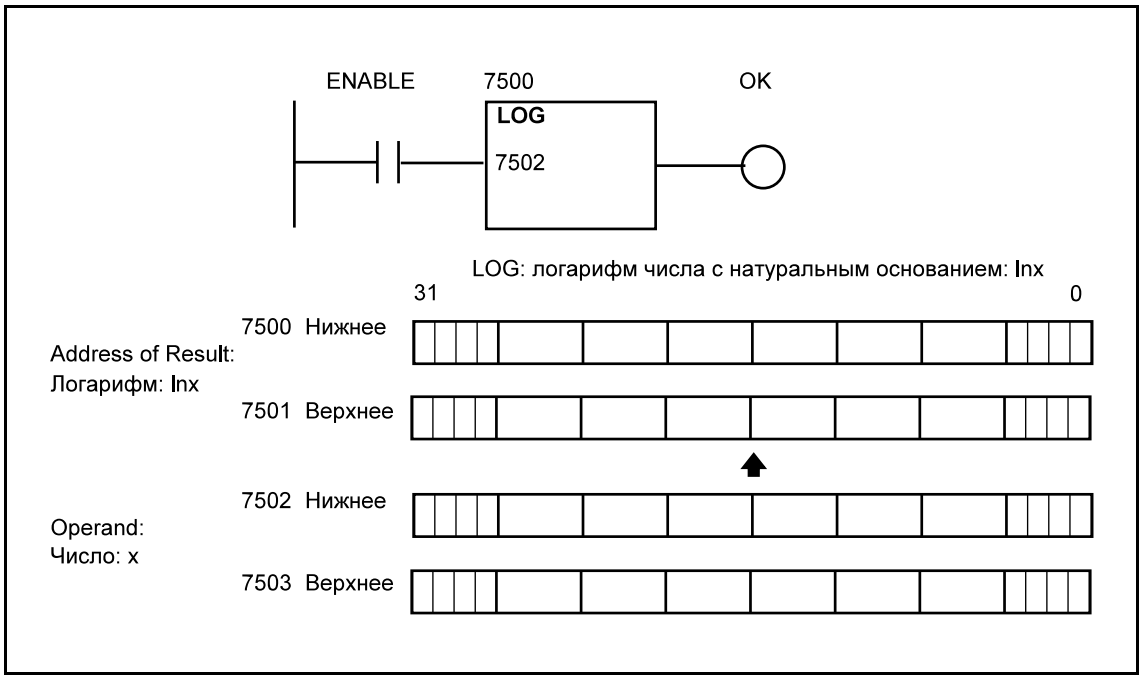
– **Operand**

это то число, для которого берётся логарифм с натуральным основанием.

☞ ***Внимание:*** Для значения Operand должны быть верны следующие:

0 < Operand

Если условие выше не удовлетворяется, команда не выполняется и выводится сообщение об ошибке FL_ER=1 ВЕРНО.



6.11 Конверсионные команды

С помощью конверсионных команд можно конвертировать данные из одного формата в другой.

Ввод и вывод конверсионных команд

Каждая конверсионная команда имеет один

разрешающий ввод.

Разрешающий ввод выполняет конверсию в состоянии верно.

Для конверсионных команд можно *конфигурировать*

вывод.

Вывод примет состояние ВЕРНО, если ввод команды ВЕРНО и выполнена конверсионная операция. Вывод примет состояние ФАЛЬШ, если его ввод ФАЛЬШ, или, если команда не выполняется, то есть выводится сообщение об ошибке FL_ER!

К выводу можно привязать даже новые дальнейшие команды.

Вводные параметры являются общими для всех конверсионных команд.

Вводные параметры конверсионных команд

– **Address of Result**:

Предел значения: n...9999 (n: где начинаются пользовательские адреса)

Адрес целевого регистра задаётся численно, или символично. Возможно задача индексированно.

Содержание конвертируемого регистра попадает на этот адрес.

– **Operand**:

Адрес конвертируемого регистра.

Адрес регистра может быть символичным, числом, или индексированным.

– **Output**:

В разрешенном состоянии на вывод коробки команд можнопривязать дальнейшие команды.

– **Remark**:

Если вывод не разрешён, записанное сюда замечание изображается в виде комментария, далее, если сюда ничего не запишем, то сюда записывается комментарий символа, заданный в поле Address of Result.

6.11.1 Преобразование числа BCD в бинарное число: BIN

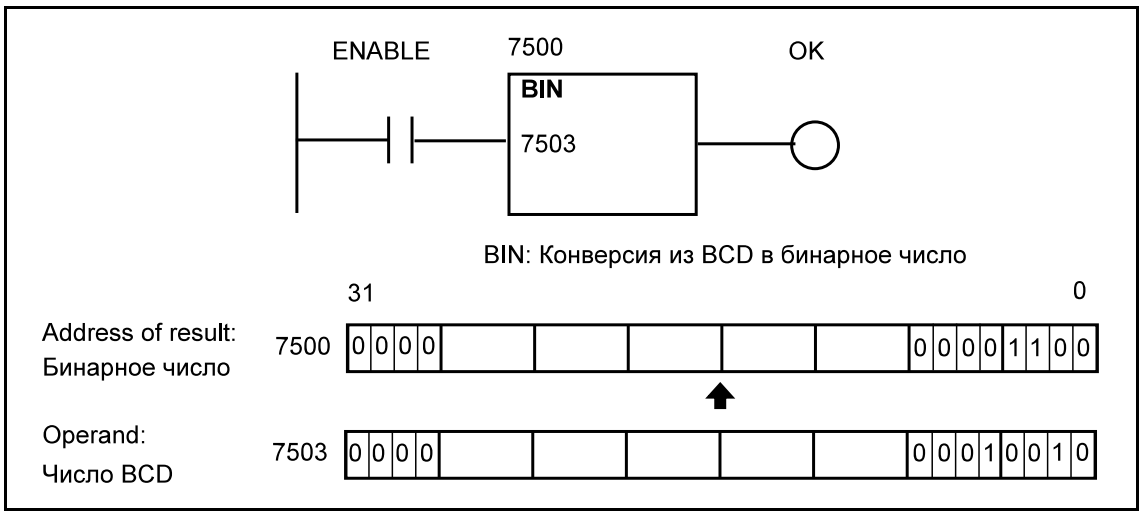
Операндом команды обязательно должно быть число BCD.
Наибольшее преобразовываемое число BCD: #$99999999, то есть его можно изображать на 8 десятичных цифрах.

☞ ***Внимание:***

Команда не выполняется и выводится сообщение об ошибке FL_ER в BEPHO, если конвертируемое число не BCD. Например число

#$000002AF

не является числом BCD!



6.11.2 Преобразование бинарного числа в BCD число: BCD

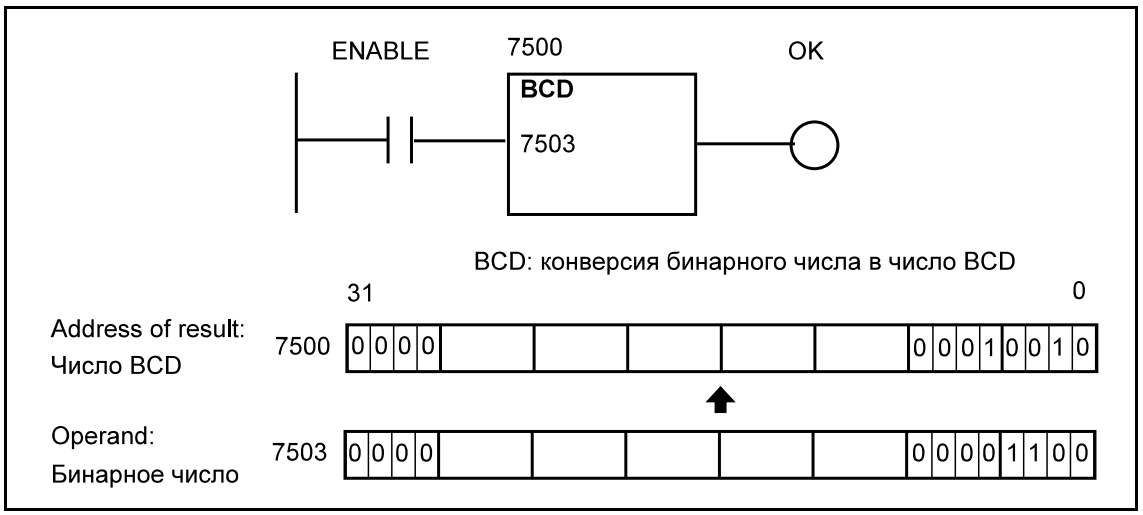
Наибольшее бинарное число, которое можно конвертировать в число BCD: #$05F5E0FF. Это число соответствует числу на 8 десятичных цифрах #$99999999.

☞ ***Внимание:***

Команда не выполняется и выводится сообщение об ошибке FL_ER в ПРАВДА, если конвертируемое бинарное число нельзя преобразовать в число BCD на 8 десятичных цифрах. Например число

#$2FFFFFFF

нельзя преобразовать!



6.11.3 Преобразование числа с фиксированной запятой в число с плавающей запятой: FLT

☞ ***Внимание!***

Результат сохраняется по адресам

Address of Result

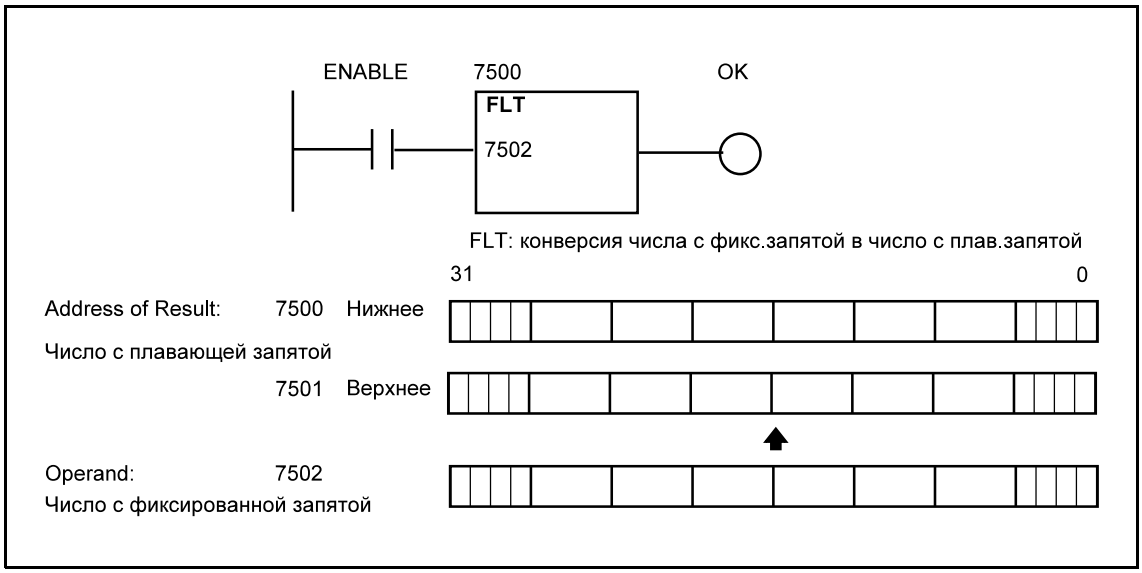
Address of Result+1!

Данными, заданными вводным параметром Operand, обращается как целым числом со знаком, поэтому результат получим после конверсии тоже согласно правилу знаков. Например, число

#$FFFFFFFF

после преобразования в число с плавающей запятой даст результат:

-1.000.



6.11.4 Преобразование числа с плавающей запятой в число с фиксированной запятой: FIX

☞ ***Внимание!***

Конвертируемое число с плавающей запятой берётся командой из адресов

Operand

Operand+1!

Совокупность конвертируемых чисел с плавающей запятой следующие:

-2147483648.000... ≤ Operand ≤ 2147483647.000...

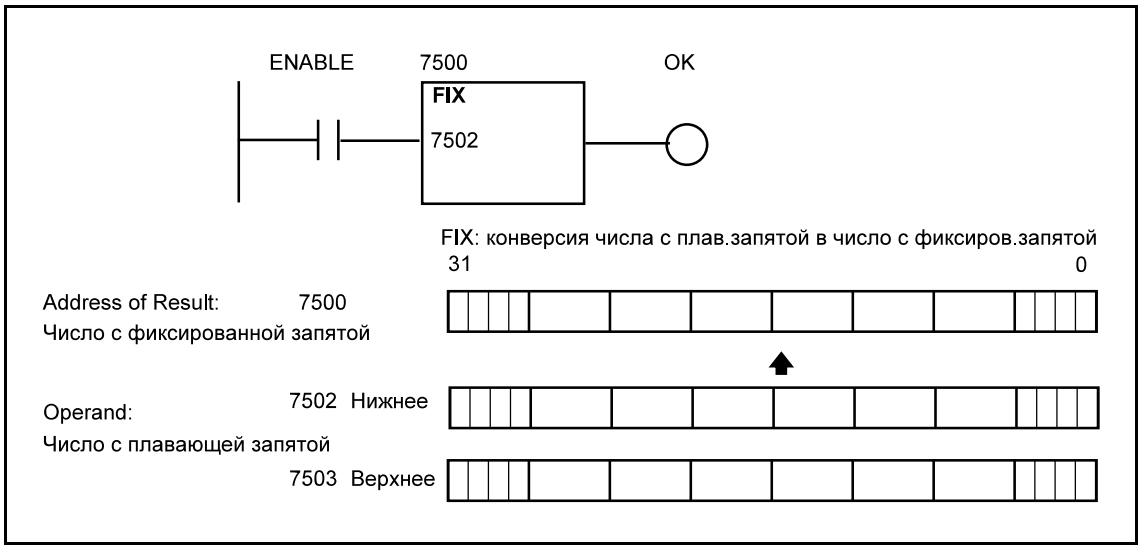
Если конвертируемое число с плавающей запятой выходит за пределами заданной выше области, команда не выполняется и выводится сообщение об ошибке FL_ER в BEPHO.

Число с фиксированной запятой, полученное по адресу Address of result получится согласно правилу знаков. Например, число с плавающей запятой

-1.000

после преобразование в число с фиксированной запятой, даст результат:

#$FFFFFFFF.



6.11.5 Конверсия угла, заданного в радианах, в градусы: DEG

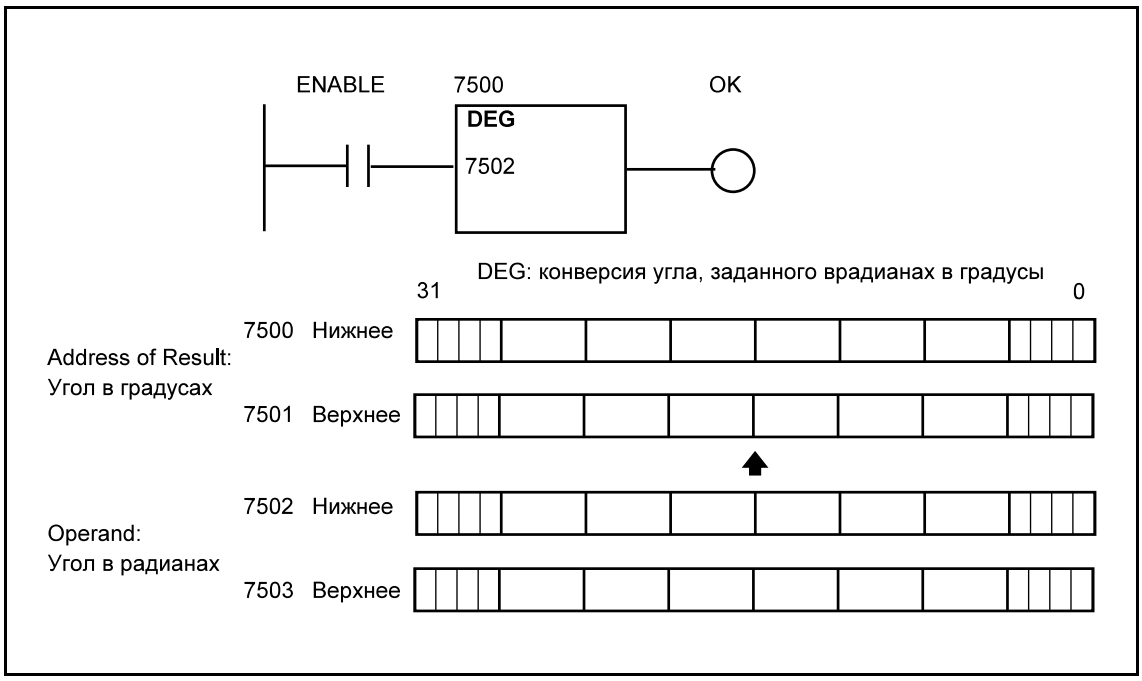
☞ ***Внимание!***

Угол, заданный в радианах, берётся командой в качестве числа с плавающей запятой из адресов

Operand
Operand+1!

Результат, полученный в градусах, сохраняется по адресам

Address of Result
Address of Result+1!



6.11.6 Конверсия угла, заданного в градусах, в радианы: RAD

☞ ***Внимание!***

Угол, заданный в градусах, берётся командой в качестве числа с плавающей запятой из адресов

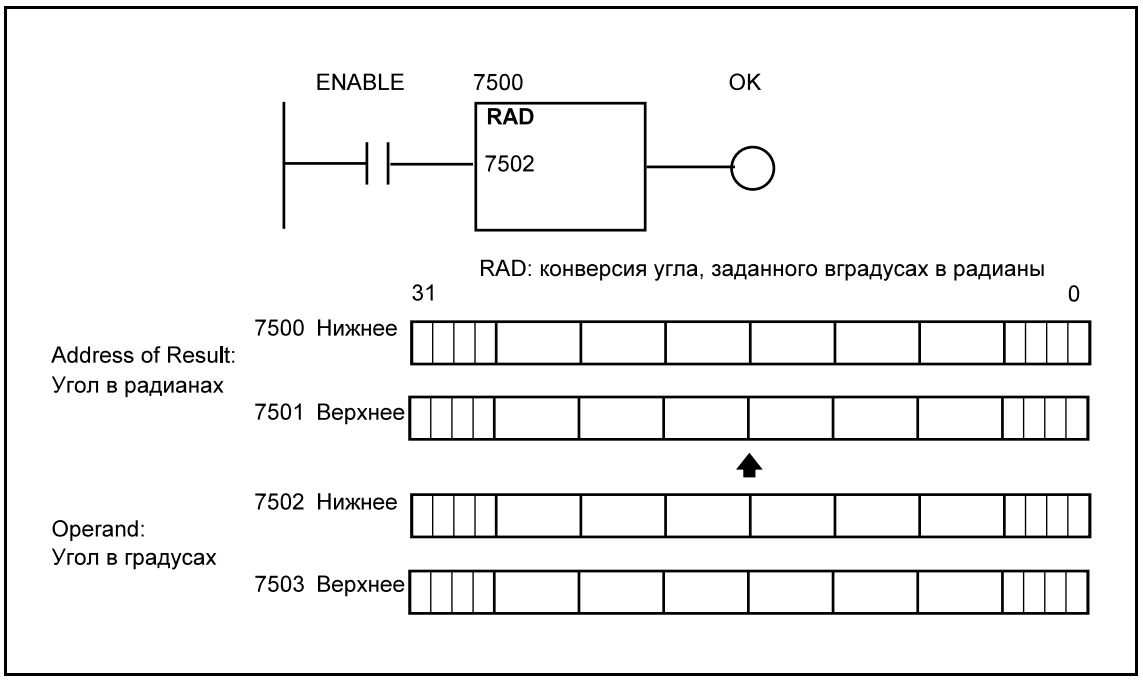
Operand

Operand+1!

Результат, полученный в радианах, сохраняется по адресам

Address of Result

Address of Result+1!



6.12 Команды для сравнивания

С помощью команд для сравнения можно сравнивать два значения на 32 битах, со знаком, с фиксированной запятой, или два значения на 64 битах с плавающей запятой.

6.12.1 Команды CMP и FCMP

Команда

CMP выполняет сравнение чисел на 32 битах, **со знаком**, **с фиксированной запятой**, а

FCMP выполняет сравнение чисел на 64 битах **с плавающей запятой.**

Обе команды имеют один

статический разрешающий ввод.

Ни одна из команд **не имеет вывода**.

Результат сравнения определяется оценкой сообщений

FL_GT больше, чем (>),
FL_EQ равно (=), и
FL_LT меньше, чем (<).

Вводные параметры команд CMP и FCMP

– **Address of Value**:

Предел знаяения: 0...9999

Адрес значения сравниваемой левой стороны можно задавать численно, или символично. Возможно задача индексированно.

– **Data**:

Значение сравниваемой правой стороны.

В случае команд CMP является числом со знаком, с фиксированной запятой,

В случае команд FCMP является числом с плавающей запятой.

В обоих случаях можно ссылаться на дааные с ссылкой на регистр. Возможно задача индексированно.

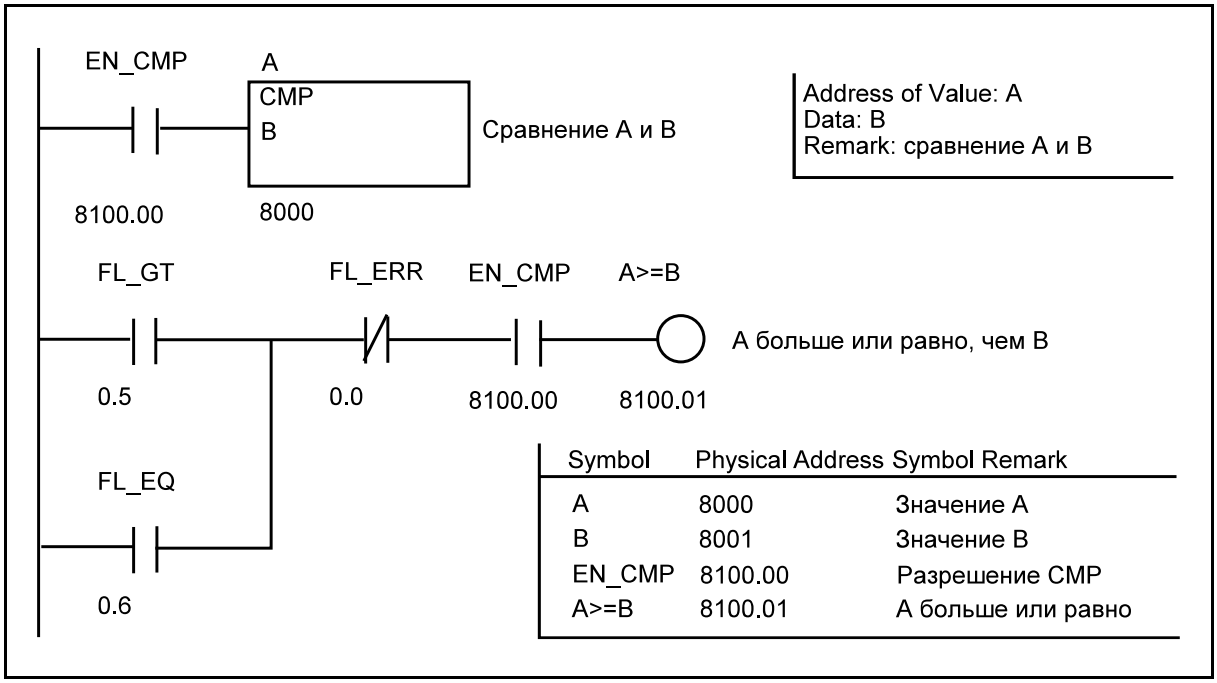
☞ ***Внимание:***

В случае команд FCMP данные с плавающей запятой берутся из адреса Address of Value и Address of Value+1. То же самое относится к параметру Data, если в ней запишем ссылку на регистр.

☞ ***Внимание!***

Если заданные адреса выходят за пределами области определения, командой выводится сообщение об ошибке FL_ER.

На рисунке ниже показано использование команды CMP при оценке условий A≥B.



6.12.2 Команды LT, FLT, LE, FLE, EQ, FEQ, NE, FNE, GE, FGE, GT, FGT

Команды для сравнения, выполняемые для чисел на 32 битах, **со знаком**, **с фиксированной запятой**:

CLT (<) меньше, чем
CLE (<=) меньше, или равно
CEQ (=) равно
CNE (<>) не равно
CGE (>=) больше, или равно
CGT (>) больше, чем

Команды для сравнения, выполняемые для чисел на 64 битах **с плавающей запятой**:

FLLT (<) меньше, чем
FLLE (<=) меньше, или равно
FLEQ (=) равно
FLNE (<>) не равно
FLGE (>=) больше, или равно
FLGT (>) больше, чем

Каждая команда имеет один

статический разрешающий ввод, и один,
вывод, подающий сообщение об успешном выполнении (состояние ВЕРНО).

Вывод команды будет ФАЛЬШ и тогда, если его ввод будет ФАЛЬШ, или, если команду нельзя выполнить, то есть если командой выодится сообщение об ошибке FL_ER!

Вводные параметры команд

– **Address of Value**:

Предел значения: 0...9999

Адрес значения сравниваемой левой стороны можно задавать численно, или символично. Возможно задача индексированно.

– **Data**:

Значение сравниваемой правой стороны.

В случае команд CLT, CLE, CEQ, CNE, CGE, CGT является числом со знаком, с фиксированной запятой,

В случае команд FLLT, FLLE, FLEQ, FLNE, FLGE, FLGT является числом с плавающей запятой.

В обоих случаях можно ссылаться на данные с ссылкой на регистр. Возможно задача индексированно.

☞ ***Внимание:***

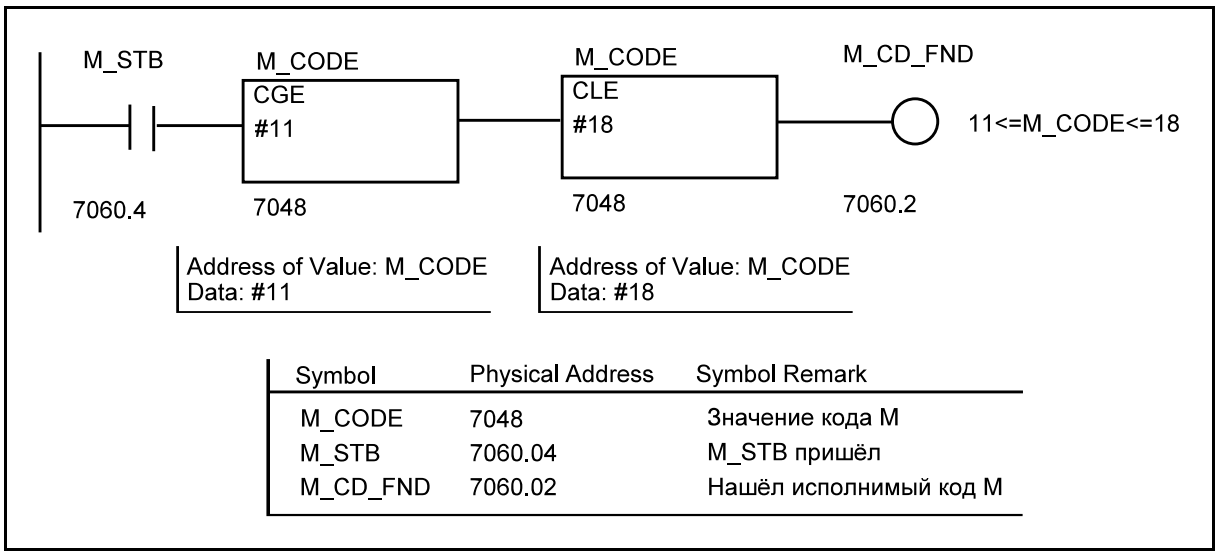
В случае команд FLLT, FLLE, FLEQ, FLNE, FLGE, FLGT данные с плавающей запятой берутся из адреса Address of Value и Address of Value+1. То же самое относится к параметру Data, если в ней запишем ссылку на регистр.

☞ ***Внимание!***

Если заданные адреса выходят за пределами области определения, вывод команды будет ФАЛЬШ и командой выводится сообщение об ошибке FL_ER.

На приведенной ниже примере видно,что обмотка M_CD_FND включается тогда, если для переменной M_CODE выполняется следующее условие:

$11 \leq M\_CODE \leq 18$



Symbol	Physical Address	Symbol Remark
M_CODE	7048	Значение кода M
M_STB	7060.04	M_STB пришёл
M_CD_FND	7060.02	Нашёл исполнимый код M

6.13 Сообщения, посылаемые из программы PLC

Из программы PLC можно посылать **сообщения** на **экран** и в **файл-журнал**. Имеются такие команды, которые посылают сообщения только на экран, имеются такие команды, которые посылают только в файл-журнал, и имеются такие, которые посылают сообщения как на экран, так и в файл-журнал.
Выводимые на экран сообщения накапливаются в буффере для сообщений и ожидают, чтобы вследствие какого-то вмешательства удалились оттуда. Таким вмешательством может быть напримериспользование клавиши удаления, клавиши старта, или просто лишь прекращение причины сообщения.
Важнейшие события и ошибки NC собирает в так называемом файле-фурнале. Сообщения и записи о событиях сохраняются в файле-фурнале, подискываемо обратно в течение примерно 1 месяца. В этот файл-фурнал может записать сообщения и программа PLC.
Сообщения PLC имеют **код** и **текст**. Из программы PLC можно посылать в итоге 999 сообщений, которые внутри кода текста идентифицируется и **номером сообщения**. Сообщения PLC являются частью глобальных сообщений, посылаемых от NC.

Порядок появления сообщений на экране

На экране первым появится всегда сообщение, выводимое последним по времени: Last In First Out. Если все сообщения выводить, то последнее сообщение будет наверху списка, а первое - внизу списка.

Код сообщений

Каждая глобальное сообщение идентифицируется 8-ми значным, десятичным числом. Это число называется кодом сообщения.
Код сообщения формируется следующим образом:

AABCCCDD

где:

AA: **индекс канала**. Его значение при сообщений, независимых от каналов будет 0, а для тех, которые зависят от кагналов, указывается номер того канала, из которого пришло сообщение.
При сообщении PLC програмист PLC решает, что сообщение не зависит ли от каналов, или сообщение назначает к данному каналу. Например, сообщение “Аварийное состояние” независимое от каналов, значит, его индекс AA=0, а сообщение “Прошу вращение шпинделя” придёт от данного канала, значит, его индекс AA>0. Это важно потому, что тот же текст сообщения можно посылать из многих каналов, однако кодом сообщения однозначно надо идентифицировать отправителя сообщения.

B: **индекс модуля**: это является индексом модуля той системы (измерительной системы, интерполятора, подготовителя кадра, и т.д.), которая посылает сообщение.
Его значение при **сообщении PLC** B=0.

CCC: **номер сообщения**. Номер сообщения может быть от 1 до 999.

DD: **произвольный индекс**: при сообщений NC это может быть например индексом оси, шпинделя и т.д.
При сообщении PLC програмист решает: это будет индексом например шпинделя, или револверным индеком, то есть он кодирует, что какому по номеру шпинделю, или какому по номеру револверной головке относится сообщение. Это важно потому, что тот же текст сообщения может относиться к многим шпинделям, к многим револверным головкам, однако кодом сообщения однозначно надо идентифици-

ровать отправителя сообщения.

Классификация команд программы PLC, посылающих сообщения

На уровне команд PLC можно классифицировать сообщения следующим образом:

– Выводятся на экран, или нет:

выводятся на экран: **MSG, MSGF, ALR, ALRF**

не выводятся на экран: **REM, REMF**

– Попадают в файл-журнал, или нет:

попадают в файл-журнал: **REM, REMF, ALR, ALRF**

не попадают в файл-журнал: **MSG, MSGF**

При составлении программы PLC необходимо следить за тем, что какое сообщение какой командой выводим.

Например,сообщение “Дверь открыта” надо выводить на экран, чтобы информировать оператора, почему не идёт обработка. Поскольку сообщение может много раз появится в течение дня, но оно указывает не на ошибку, или проблему по работе, просто предупреждает оператора, было бы лишно вводить его в файл-журнал. Значит, в таком случае обосновано пользоваться командой MSG.

Однако, если сообщение выводится из-за ошибки, например, включатель защиты двигателя находится в отключенном состоянии, целесообразно пользоваться командой ALR, так как это сообщение выводится на экран и записывается и в файл-журнал.

В том случае, если ничего не желаем выводить на экран, не требуется никакого вмешательства оператора, но что-нибудь желаем записать в файл-журнал, надо пользоваться командой REM.

В **сообщение PLC**, на произвольное место внутри текста сообщения, можно опционально записать и **числовое значение**. Число может быть либо с фиксированной либо с плавающей запятой. Сообщения можно классифицировать и так, что желаем ли помещать в сообщении данные с фиксированной либо с плавающей запятой:

– Сообщения**,** содержащие данные с **фиксированной запятой**: **MSG**, **ALR**, **REM**,

– Сообщения**,** содержащие данные с п**лавающей запятой**: **MSGF**, **ALRF**, **REMF**.

Текст сообщений PLC

Программа редактора PLC поддерживает редактирование текстов сообщения, они сохраня.тся в файле *.plc вместе с программой PLC.

Сообщения можно сохранить и отдельно под названием *.mes с помощью редактора PLC, клавишей Экспорт сообщений. Так же можно импортировать уже существующий файл -сообщения *.mes в программу PLC.

Кодирование текстового файла *.mes выполняется по UTF8. Это значит, что в нём можно задавать все известные в данный момент знаки препинания.

Вывод текстового содержания переменной PLC внутри сообщения

В тексте сообщения в произвольном месте можно выводить содержания регистра. То обстоятельство, что выводимое число является с фиксированной запятой или с плавающей запятой, решается командой PLC. В редакторе можно задавать, что с точностью до скольких десятичных знаков должно быть написано число, если выводимое число является числом с плавающей запятой.

Например, можно выводить текст, что “ Вставьте n-й инструмент в шпиндель”, где “n” является значением одного регистра PLC.

При экспорте сообщения получим следующий формат:

Вставьте {#0:F0}. инструмент в шпиндель

где в тексте ряд характеров {#0:F0} означает, что бинарные данные с фиксированной запятой, заданные в команде MSG, ALR, или REM, конвертировать в десятичное число и записать в текст обработчика сообщения. значение #: число с фиксированной запятой и значение F0: не использовать десятичную запятую при выводе.
При экспорте сообщения, содержащего число с плавающей запятой получим текст:

Температура шпинделя {*0:F1} °C

где в тексте ряд характеров {*0:F1} означает, что бинарные данные с плавающей запятой, заданные в команде MSGF, ALRF, или REMF конвертировать в десятичное число и записать в текст обработчика сообщения с точностью до 1 десятичного знака. * означает изображение с плавающей запятой, а F1 означает, что число надо выводить с точностью до 1 десятичного знака.
Правила формирования:

{ (начало формирования)
(данные с фиксированной запятой), или
***** (данные с плавающей запятой)
0 (обязательно)
: (разделить)
F
i (количество выводимых десятичных знаков при изображении с плавающей запятой, для числа с фиксированной запятой 0)
} (конец формирования)

6.13.1 Команды для посылки сообщений: MSG, MSGF, ALR, ALRF, REM, REMF

Сообщения PLC, выводимые командами ***MSG, MSGF, ALR, ALRF*** **выводятся на экран**, в поле сообщения. Задачей програмиста PLC является решать, что при каких вмешательств оператора удаляются эти сообщения с экрана.
Сообщения команд ***MSG, MSGF*** **не попадают в файл-журнал**.
Сообщения команд ***ALR, ALRF*** **всегда попадают в файл-журнал**.
Сообщения PLC, высланные командами ***REM, REMF*** **не выводятся на экран**, в поле сообщения, однако всегда **попадают в файл-журнал**.
Опциональные вводные переменные команд ***MSG, ALR, REM*** являются **с фиксированной запятой**, а
опциональные вводные переменные команд ***MSGF, ALRF, REMF*** **с плавающей запятой**.

Вводы и выводы команд

Все команды, высылающие сообщения, **на своём разрешающем вводе**
– запишет данные сообщения на набегающий фронт в буффере сообщения, относящемся к экрану и в файл-журнал,
– а удаляет сообщение на задний фронт из буффера сообщения, относящегося к экрану.
Эти команды **имеют выводы**: вывод примет значение ВЕРНО, если сообщение выводится на экран, или попадает в журнал при команде REM, REMF.

Вводные параметры команд MSG, MSGF, ALR, ALRF, REM, REMF:

– **Message Number:**

Предел значения: 1,...999
номер сообщения
Число с фиксированной запятой, или адрес регистра. Возможно задача индексированно.

– **Channel Index:**

Предел значения: 0...99
индекс канала сообщения.
Число с фиксированной запятой, или адрес регистра. Возможно задача индексированно.

– **Arbitrary Index:**

Предел значения: 0...99
произвольный индекс сообщения.
Число с фиксированной запятой, или адрес регистра. Возможно задача индексированно. Можно задавать например номер оси, шпинделя, револьверной головки.

– **Optional Variable:**

Предел значения: в случае команд MSG, ALR, REM число на 32 битах, или адрес регистра,
в случае команд MSGF, ALRF, REMF число с плавающей запятой, или адрес регистра
опциональная переменная сообщения
Возможно задача индексированно.

– **Remark:**

Замечание

Работа и вывод команд MSG, MSGF, ALR, ALRF,:

Текст сообщения вставится в буффер сообщения командой, если она на своём вводе детектирует набегающий фронт.
Текст сообщения удаляется из буффера сообщения командой, если она на своём вводе детектирует задний фронт.
В отношение набегающего фронта на вводе, записывается код и текст сообщения командой ALR, ALRF и в файл-журнал.
Вывод команд будет ВЕРНО тогда, если код сообщения, имеющийся в команде, выводится на экран:

AA0CCCDD

где:
AA: значение параметра команды Channel Index,
0: сообщение выслано от PLC,
CCC: значение параметра команды Message Number,
DD: значение параметра команды Arbitrary Index.

Работа и вывод команд REM, REMF:

В отношение набегающего фронта на вводе, записывается код и текст сообщения командой REM, REMF в файл-журнал.
Вывод команд будет ВЕРНО тогда, если код сообщения, имеющийся в команде, практически сразу попадает в файл-журнал.

Примеры:

Сообщение об ошибке останова, относящееся к шпинделю с индексом S1:

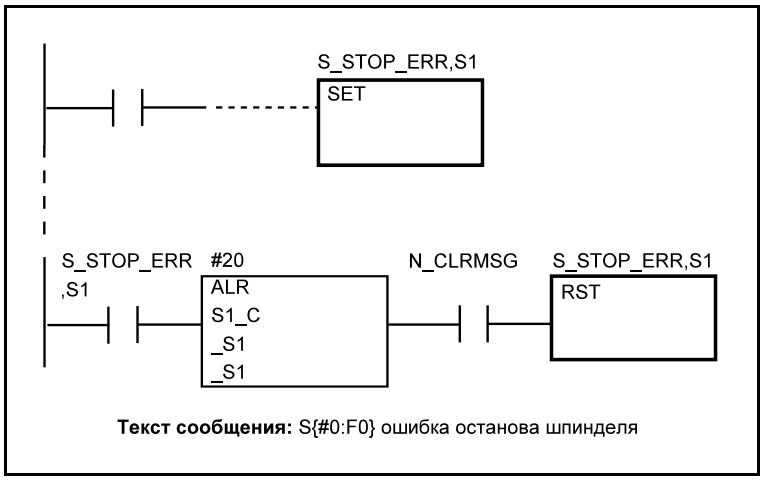
S_STOP_ERR

номер сообщения: #20

Сообщение приходит с канала S1_C, где регистр S1_C скажет, что шпиндель S1 как раз в каком канале находится.

Сообщение относится к шпинделю с номером _S1, где _S1 номер шпинделя S1.

(В нашем случае индекс шпинделя S1=0, номер шпинделя _S1=1)



В текст сообщения также запишем номер шпинделя: значение опциональной переменной _S1.

Сообщение вполностью раскодировано: канал и шпиндель с номером n. В интересах лучшей обозримоти в текст сообщения также включили номер шпинделя.

Текст сообщения отождествляется числом 20. Тот же текст сообщения можно использовать и для 2-го, 3-го и т.д. шпинделей., значит, не нужно вводить новый текст снова по шпинделям, всего лишь номер шпинделя надо менять.

Если на воде коробки сообщения появится состояние ВЕРНО, сообщение попадает в файл-журнал и в буффер сообщения.

Если сообщение выводится на экран, вывод команды ALR попадает в состояние ВЕРНО. При этом с помощью указателя N_CLRMSG, работающего под действием клавиши CANCEL, с удалением указателя S_STOP_ERR,S1 удаляется сообщение.

В канале с индексом CH1 запросим нажать клавиш старта на указателе

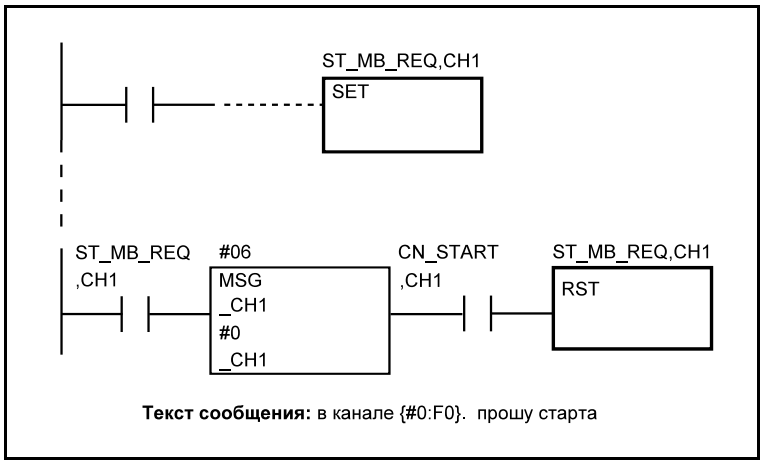
ST_MB_REQ.

Номер сообщения: #06

Сообщение запросим из канала с номером _CH1.

(В нашем случае индекс канала CH1=0, номер канала _CH1=1.)

Значение произвольного индекса #0.



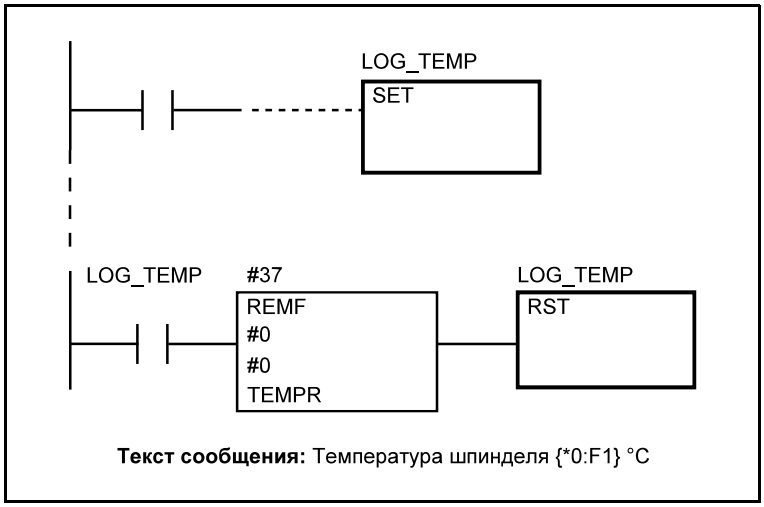
В текст сообщения также запишем номер канала с использованием опциональной переменной: _CH1.

Сообщение вполностьб раскодировано с номером канала. В интересах лучшей обозримоти в текст сообщения также включили номер канала.

Текст сообщения отождествляется числом 06. Тот же текст сообщения можно использовать и в других каналах, значит, не нужно вводить снова новый текст, всего лишь номер канала надо менять.

Если на воде коробки сообщения появится состояние ВЕРНО, сообщение попадает в файл-журнал, но в буффер сообщения нет.
Если сообщение выводится на экран, вывод команды MSG попадает в состояние ВЕРНО. При этом под действием клавиши START сообщение удаляется, одновременным удалением указателя ST_MB_REQ,CH1.

После истечения заранее определённого времени, PLC вводит в журнал температуру шпинделя: выводит сообщение
LOG_TEMP.
Номер сообщения #37.
На станке имеется только один канал и один шпиндель, поэтому индекс канала и шпинделя также #0.
В регистре с значком TEMPR хранится температура шпинделя в виде числа с плавающей запятой.



В текст сообщение запишем, чтобы выводились данные с плавающей запятой с точностью до одного десятичного знака {*0:F1}.
Если вод коробки сообщения переключается на состояние ВЕРНО, содержание файла попадает в файл-журнал, а вывод коробки сообщения одновременно примет состояние ВЕРНО. Этим удаляется прежний указатель запроса ввода в журнал.

6.14 Команды для управления программой

6.14.1 Команда конец модуля: END

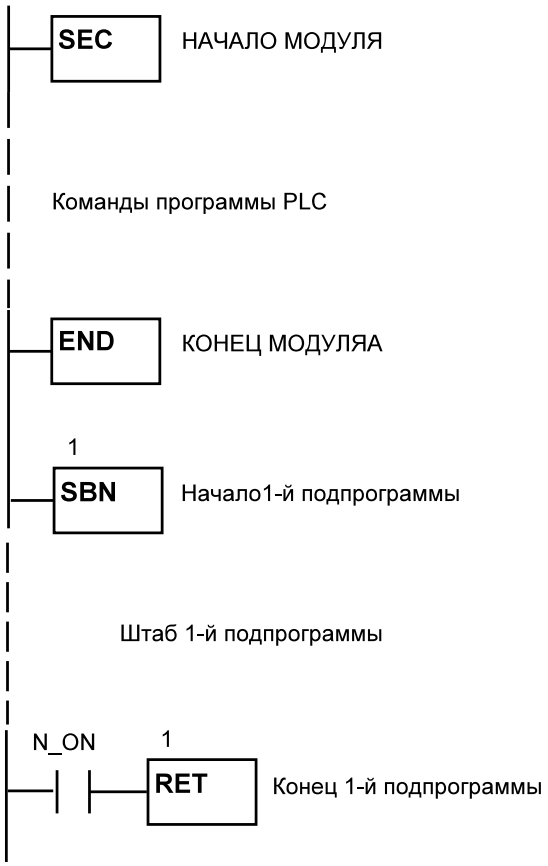
Команда, указывающая конец цикличного модуля PLC. Ей можно пользоваться как в Главной программе, так и в модуле Int0.
Модуль программы PLC выполняет команды программы с самого начала ступенчатой программы, затем, когда выполнение программы доходит до команды END, закончится прогон модуля программы PLC, и запускается снова только после истечения времени цикла (в случае Главной программы - T_{PLC}, в случае Int0 - TimeSlice).
Команда END имеет разделительную роль. Штаб подпрограмм, использованных в модуле программ PLC, необходимо записать также в формате ступенчатой диаграмме после команды END. Если в модуле не используем подпрограмы, пользоваться командой не обязательно.

☞ ***Внимание!*** *Хотя команда END является конечным элементом, её можно записать исключительно в 1-й столбец!*

Вводный параметр команды END:

– Remark:

Замечание: текст, записанный в Remark, будет комментарием логического блока.



6.14.2 Скачок в модуле программ PLC: команды JMP и JME

С помощью команд JMP и JME можно перескачить данный отрезок модули программы PLC.
Задействие команды на скачок JMPобусловлено: если условие выполняется на вводе, тогда выполняет скачок. Скачок выполняется на значок с идентификацией, заданной в команде JMP.
Командой JME можно выделить тот значок, куда команда JMP выполняет скачок.

☞***Внимание!*** *Значки, использованные в командах JMPи JME могут распостраняться от 0 до 99. Надо следить за тем, чтобы каждый значок имел индивидуальный вид и не было перекрытия значками команд SBS, SBN и RET подпрограмм!*

☞***Внимание!*** *Командой JMP можно выполнить скачок только внутри модуля, в другой модуль нельзя!*

Ввод команды JMP

Имеется один

разрешающий ввод.

Разрешающий ввод выполняет скачок только в его состоянии ВЕРНО.

Вводные параметры команды JMP

– **Jump Number**:
Предел значения: 0...99
Идентификация значка, куда выполняется скачок.

– **Remark**:
Замечание: текст, записанный в Remark, будет комментарием логического блока.

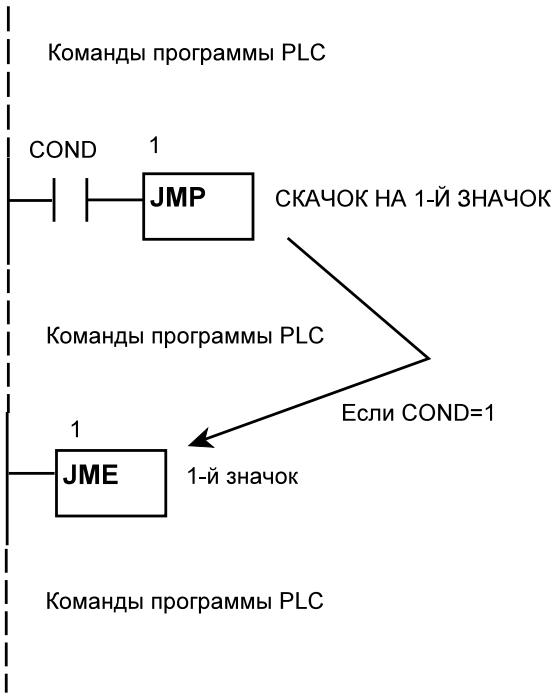
Ввод команды JME

☞ ***Внимание!*** *Хотя команда JME является конечным элементом, её можно записать исключительно в 1-й столбец, она не имеет ввода!*

Вводные параметры команды JME

– **Jump Number**:
Предел значения: 0...99
Идентификация значка, команда JMP сюда выполняется скачок.

– **Remark**:
Замечание: текст, записанный в Remark, будет комментарием логического блока.



6.14.3 Команды вызова подпрограммы: команды SBS, SBN и RET

Из модуля программы PLC командой SBS можно проявить инициативу для вызова подпрограммы. На вводе команды надо задавать условие, что когда вызвать подпрограмму. В команде надо задавать идентификацию подпрограммы, то есть значок, куда передаётся управление.
Подпрограмма начинается командой SBN, снабжённой соответствующим значком.
Конец подпрограммы можно задавать командой RET. На вводе команды надо задавать условие, что когда она может возвращаться.
Штаб подпрограмм, значит, программную часть, охваченной командами SBN и RET, надо написать всегда после команды END модуля!
Возможно выполнить многоратные вызовы, закладыванные друг в друга. Глубина вызова ограничена только количеством значков, имеющихся на распоряжении.

☞***Внимание!*** *Значки, использованные в командах SBS, SBN и RET внутри одного модуля, могут распостраняться от 0 до 99. Надо следить за тем, чтобы каждый значок имел индивидуальный вид и не было перекрытия значками команд JMP, JME!*

☞***Внимание!*** *Подпрограмму, описанную в одном модуле, нельзя вызывать из другого модуля!*

Ввод команды **SBS**

Имеется один

разрешающий ввод.

Разрешающий ввод выполняет вызов подпрограммы только в его состоянии ВЕРНО.

Вводные параметры команды **SBS**

– **Subroutine Number**:
Предел значения: 0...99
Идентификация, значок подпрограммы, которая подлежит вызову.
– **Remark**:
Замечание: текст, записанный в Remark, будет комментарием логического блока.

Ввод команды **SBN**

☞ ***Внимание!*** *Хотя команда SBN является конечным элементом, её можно записать исключительно в 1-й столбец, она не имеет ввода!*

Вводные параметры команды **SBN**

– **Subroutine Number**:
Предел значения: 0...99
Идентификация значка, командой SBS это вызывается.
– **Remark**:
Замечание: текст, записанный в Remark, будет комментарием логического блока.

Ввод команды **RET**

Имеется один

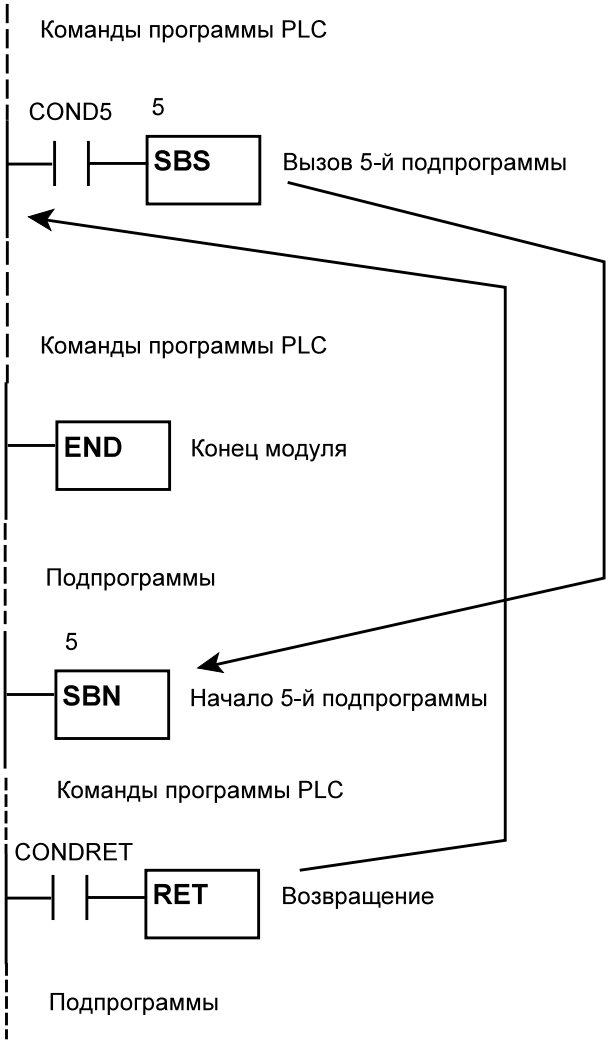
разрешающий ввод.

Возвращается из подпрограммы только в состоянии ВЕРНО разрешающего ввода.

Вводные параметры команды **RET**

– **Remark**:

Замечание: текст, записанный в Remark, будет комментарием логического блока.



6.15 Команда для перемещения оси: MOVCMD

Перемещение осей является в основном задачей NC.
Однако, может потребоваться, что **PLC** иногда **попросит** одну или несколько **осей у NC**, чтобы перемещать их своими командами, затем после выполнения операции вернуть ось или оси для NC. Таким случаем может быть например, когда во время смены инструмента или паллета приходится послать некоторые оси в позицию смены. Просьба, возврат, пуск и т.д. осей происходит через указатели, служащие для этой цели.

☞ ***Внимание!*** *Перемещениями такого характера нужно обращаться строго в функции разгрузки (сборки) буфера в PLC, эти функции можно выделить параметрами. Причиной этого является то, что после выполнения функции NC должен зачитать позиции осей, перемещённых из PLC и исходя из этой новой позиции должен продолжать обработку.*

Другой областью применения перемещения осей является, когда перемещение данной оси входит исключительно в круг действия PLC. Таким может быть например вращение револьверной головки или магазина с помощью серводвигателя. При этом PLC попросит ось у NC уже в момент включения управления и даже не должен вернуть её.

Команда для перемещения оси напишет команду непосредственно не в интерполятор, а в **буфер одного кадра**. Интерполятор забирает команду из буфера только после выполнения предыдущей команды. Нерперывным писанием буфера можно нерперывно перемещать буфер.

Ввод и вывод команды для перемещения оси

Команда для перемещения оси имеет один

разрешающий ввод.

Разрешающий ввод в состоянии верно дождётся, чтобы разгруился буфер кадра и запишет команду в буфер.
Для команды для перемещения оси можно *конфигурировать*

вывод.

Вывод примет состояние ВЕРНО после того, что удалось записать команду в буфер. Вывод примет состояние ФАЛЬШ, или, если команду не выполнима, выводит сообщение об ошибке FL_ER!
К выводу можно присоединить и новые команды.

Вводные параметры команды для перемещения оси

– **Axis Address**:
  Индекс перемещаемой оси численно, или символично. Возможно индексированная задача. Целое число, предел значения 0...31.
– **Command Code**:
  Код команды может быть символичный, численный, или индексированный. Целое число. Истолкования и возможные значения кода команды включены в таблицу ниже.
– **Speed**:
  Скорость команды движения. Может быть символичная, численная, или индексированная. Данные истолкуются в выходной системе измерения на основании параметра IND: в мм/мин, градус/мин, или дюйм/мин. Число с плавающей запятой.

– **Distance**:
Длина пути движения. Может быть символичная, численная, или индексированная. Данные истолкуются в выходной системе измерения на основании параметра IND: мм, градус, дюйм. Число с плавающей запятой.
– **Output**:
В разрешённом состоянии к выводу коробки команд можно присоединить и дальнейшие команды.
– **Remark**:
Если вывод не разрешён, записанное сюда замечание выводится как комментарий, далее, если ничего не пишем сюда, то сюда запишется комментарий символа, заданного в поле Axis Address.

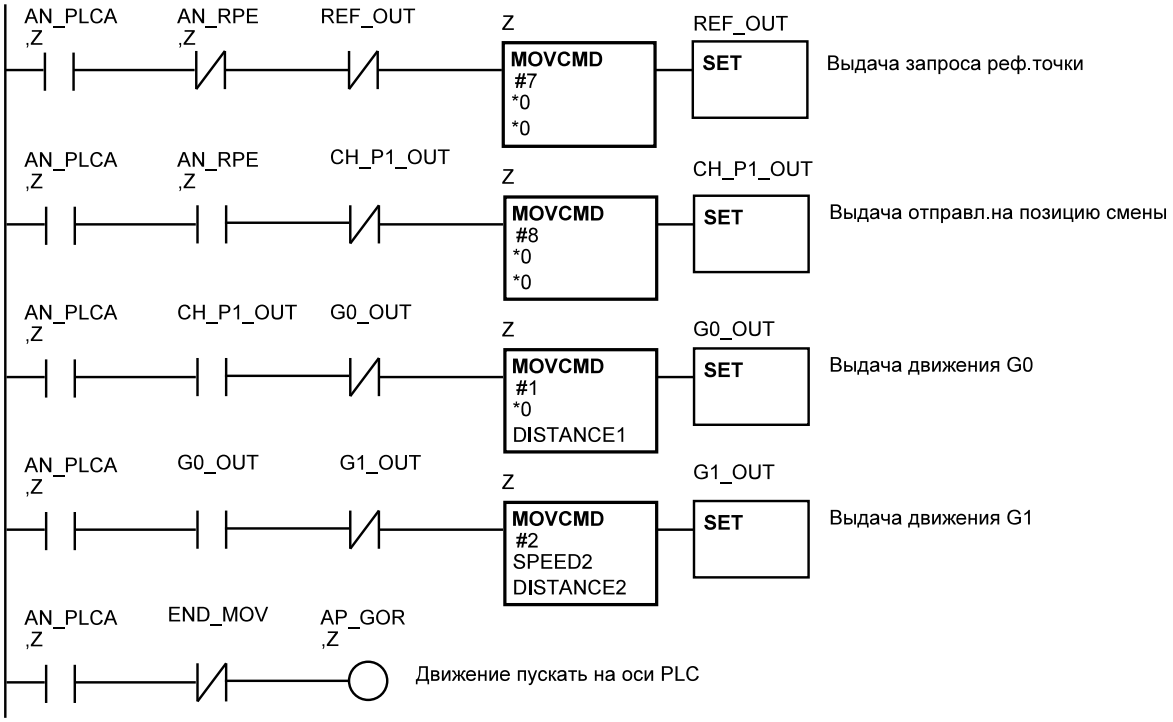
Выводится сообщение об ошибке FL_ER в следующих случаях:
перемещаемая ось не переключена в ось PLC,
код команды не правильный (правильные коды см. в таблице ниже),
для тех движений, где нужен параметр Speed и значение параметра 0.

Истолкование **Command Code**:

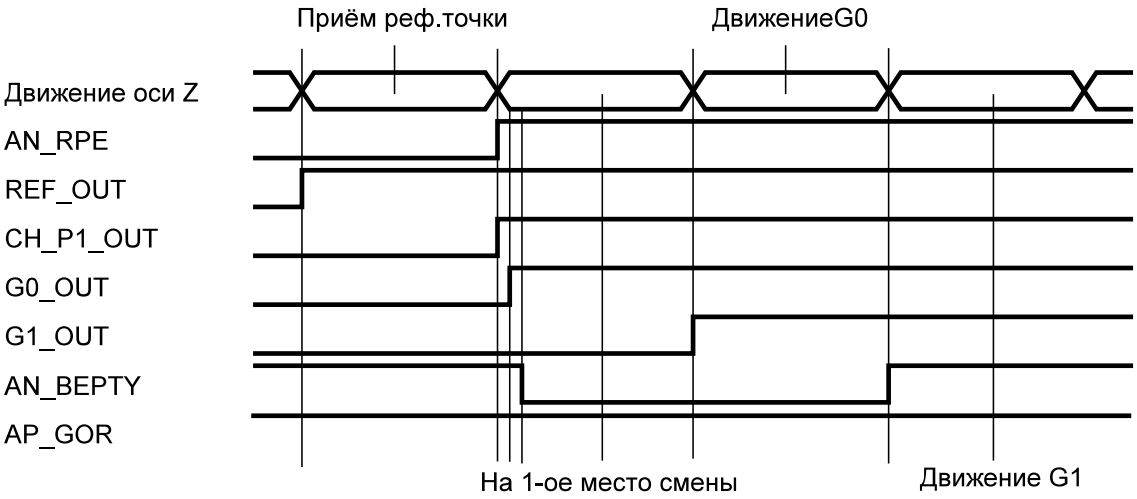
Com. Code	Эквивалентный код NC	Описание	Speed	Distance
00	G00 G90	Движение быстрого хода	не считается	абсолютная позиция, заданная в актуальной системе кординат
01	G00 G91	Движение быстрого хода	не считается	Приращение перемещения
02	G01 G90 G94	Прямая интерполяция	минутная подача	абсолютная позиция, заданная в актуальной системе кординат
03	G01 G91 G94	Прямая интерполяция	минутная подача	Приращение перемещения
04	G01 G90 G95	Прямая интерполяция	подача за оборот	абсолютная позиция, заданная в актуальной системе кординат
05	G01 G91 G95	Прямая интерполяция	подача за оборот	Приращение перемещения
06	G53	Настройка в позицию в станочной системе кординат	не считается	абсолютная позиция, заданная в станочной системе кординат

Com. Code	Эквивалентный код NC	Описание	Speed	Distance
07	G28	Приём референтной точки	не считается	не считается
08	G30 P2	Настройка в позицию 2-й референтной точки	не считается	не считается
09	G30 P3	Настройка в позицию 3-й референтной точки	не считается	не считается
10	G30 P4	Настройка в позицию 4-й референтной точки	не считается	не считается
11	G4 G94	Ожидание	не считается	Время ожидания в секундах
12	G4 G95	Ожидание	не считается	Время ожидания во-боротах шпинделя
13	Перемещение Jog. Команда удаляется новой командой, или AP_RES	AP_JOGP=1: перемещается в положительное направление, AP_JOGN=1: в отрицательное направление	минутная подача: если её значение=0: перемещается на основании параметра 0316 Jog F Contr, если её значение >0 при заданной подаче.	не считается
14	Перемещение маховичка. Команда удаляется новой командой, или AP_RES	Если любой из регистров P_HnAS выделён к оси PLC, движется согласно маховичку	не считается	не считается

На рисунке ниже приведён образцовый пример для перемещения оси PLC. Команды MOVCMD следуют друг за другом. Ряд команд сперва проверит, имеется ли референтная точка на оси Z. Если нет, выдаётся команда на приём референтной точки, если есть, сразу отправляет ось на 2-ю референтную точку (1-ое место для смены). Затем следует движение с быстрым ходом, с приращением, со смещением DISTANCE1, потом линейная интерполяция на абсолютную позицию DISTANCE2, со скоростью SPEED2.



Команда на приём референтной точки отбирается интерполятором сразу, поэтому указатель AN_BEPTY (буффер пустой) не меняется, то есть сообщает, что буфер пустой.
Следующая команда (движение на 1-ое место смены) выдаётся после появления указателя, что имеется референтная точка (AN_RPE=1). Это отбирается интерполятором сразу (AN_BEPTY ещё 1), затем в следующем цикле PLC запишется команда G0 в буфер. С задержкой на один цикл PLC сообщается состояние AN_BEPTY=0, что буфер не пустой: интерполятор перемещается на 1-ое место смены, и команда движения G0 имеется в буфере. Как только ось поступила на место смены, команда G0 отбирается интерполятором из буфера, но команда G1 сразу запишется, значит, указатель AN_BEPTY остаётся 0. Выполнив движение G0, команда G1 отбирается интерполятором из буфера, и поскольку нет больше команд движения, принимается состояние AN_BEPTY=1.
Если требуется непрерывное движение, программа PLC не должна следить за указателем AN_BEPTY, так как команда MOVCMD выполняет это.


Приём реф.точки
ДвижениеG0
Движение оси Z
AN_RPE
REF_OUT
CH_P1_OUT
G0_OUT
G1_OUT
AN_BEPTY
AP_GOR
На 1-ое место смены
Движение G1

6.16 Писание и чтение глобальных макропеременных

Значение глобальных макропеременных #100...#499 и #500...#599 можно писать и читать из программы PLC.
Поскольку эти макропеременные существуют по каналам, при использовании команды всегда надо задавать и индекс того канала, из которого читать, или в который писать желаем.

6.16.1 Чтение глобальных макропеременных: команда MACR

Командой MACR можно запросить глобальные макропеременные (#100...#499 и #500...#599) одного канала. Эти переменные являются всегда с плавающей запятой, значит, в программе PLC для них всегда нужно занимать место 2-х двойного слова.

Ввод и вывод команды MACR

Команда MACR имеет один

разрешающий ввод.

Разрешающий ввод выполняет чтение в состоянии верно.
Для команды MACR *можно конфигурировать*

вывод.

Вывод примет состояние ВЕРНО, если ввод команды ВЕРНО и выполнил операцию чтения. Вывод будет ФАЛЬШ, далее указатель FL_ER запишется в 1, если команда не выполнима.
К выводу можно присоединить и новые команды.

Вводные параметры команды MACR

– **Address of Result**:
  Передал значения: n...9998 (n: где начинаются пользовательские адреса).
  Адрес целевого регистра численно, или символично. Возможно задача индексированно.
  Запрашиванное значение попадает на этот адрес. *Потребность в месте заданного здесь адреса всегда должно быть double, так как команда вернёт всегда значение с плавающей запятой!*
– **Number of macro variable**:
  При задаче численного значения нужно использовать предварительный знак # (десятичный ввод).
  Передал значения численно: #100...#999.
  Можно задавать и символично. Возможно ссылка индексированно.
– **Channel**:
  Индекс того канала численно, или символично, из которого желаем прочитать макропеременное..
– **Output**:
  В разрешённом состоянии к выводу коробки команды можно присоединить и дальнейшие команды.
– **Remark**:
  Если вывод не разрешён, записанное сюда замечание выводится как комментарий, далее, если ничего не пишем сюда, то напишется сюда комментарий символа, заданного в поле Address of Result.

Если заданное макропеременное выходит за пределами #100...#999, или индекс заданного

канала указывает на не существующий канал, вывод будет ФАЛЬШ, далее указатель FL_ER запишется в 1.

6.16.2 Писание глобальных макропеременных: команда MACW

Командой MACW можно писать глобальные макропеременные (#100...#499 és a #500...#599) одного канала. Эти переменные являются всегда с плавающей запятой, значит, из памяти PLC запишет всегда 2 двойного слова в макропеременное.

Ввод и вывод команды MACW

Команда MACW имеет один

разрешающий ввод.

Писание выполняется в состоянии разрешающего ввода верно.

Для команды MACW *можно конфигурировать*

вывод.

Вывод примет состояние ВЕРНО, если ввод команды ВЕРНО и выполнил операцию писания. Вывод будет ФАЛЬШ, далее указатель FL_ER запишется в 1, если команда не выполнима.

К выводу можно присоединить и новые команды.

Вводные параметры команды MACW

– **Number of macro variable**:

При задаче численного значения нужно использовать предварительный знак # (десятичный ввод).

Передал значения численно: #100...#999.

Можно задавать и символично. Возможно ссылка индексированно.

– **Operand**:

Передал значения: n...9998 (n: где начинаются пользовательские адреса)

Адрес регистра источника PLC численно, или символично. Возможно задача индексированно.

Потребность в месте заданного здесь адреса всегда должно быть double, так как командапишет всегда значение с плавающей запятой!

– **Channel**:

Индекс того канала численно, или символично, из которого желаем прочитать макропеременное..

– **Output**:

В разрешённом состоянии к выводу коробки команды можно присоединить и дальнейшие команды.

– **Remark**:

Если вывод не разрешён, записанное сюда замечание выводится как комментарий, далее, если ничего не пишем сюда, то напишется сюда комментарий символа, заданного в поле Address of Result.

Если заданное макропеременное выходит за пределами #100...#999, или индекс заданного канала указывает на не существующий канал, вывод будет ФАЛЬШ, далее указатель FL_ER запишется в 1.

6.17 Запрос внутренних переменных NC: команда SCP

Командой SCP можно запросить те внутренние регистры NC, которые можно выводить на осциллоскоп, встроенный в управление. Эти переменные могут быть различного типа: бинарные, состоящие из двойного слова, или с плавающей запятой. Однако, полученный результат всегда с плавающей запятой!

Ввод и вывод команды SCP

Команда SCP имеет один

разрешающий ввод.

Чтение выполняется в состоянии разрешающего ввода верно.

Для команды SCP *можно конфигурировать*

вывод.

Вывод примет состояние ВЕРНО, если ввод команды ВЕРНО и выполнил операцию чтения.

К выводу можно присоединить и новые команды.

Вводные параметры команды SCP

– **Address of Result**:

Предел значения: n...9998 (n: где начинаются пользовательские адреса)

Адрес целевого регистра численно, или символично. Возможно задача индексированно.

Запрашиванное значение попадает на этот адрес. *Потребность в месте заданного здесь адреса всегда должно быть double, так как команда вернёт всегда значение с плавающей запятой!*

– **Scope Symbol**:

Символ данных, выбираемых и запрашиваемых из развёртывающийся меню. Значение символов приведено в таблицу ниже.

– **Id**:

Сюда попадает номер идентификации символа, выбранного из развёртывающегося меню. Записать его не нужно.

– **Param1**:

Значение 1-го параметра, принадлежащего к выбранному символу: #<число>. Его задачу см. в таблице ниже.

– **Param2**:

Значение 2-го параметра, принадлежащего к выбранному символу: #<число>. Его задачу см. в таблице ниже.

– **Output**:

В разрешённом состоянии к выводу коробки команды можно присоединить и дальнейшие команды.

– **Remark**:

Если вывод не разрешён, записанное сюда замечание выводится как комментарий, далее, если ничего не пишем сюда, то напишется сюда комментарий символа, заданного в поле Address of Result.

Переменные NC, заданные приведёнными ниже символами, можно запрашивать с использованием команды SCP:

Символ	Описание	Param1	Param2
PlcBit	Запрос бинарного переменного PLC из памяти PLC. Данные с плавающей запятой.	Адрес двойного слова: численно, перед ним карактер # Значение: #0...#9999	Адрес бита: численно, перед ним карактер # Значение: #0...#31
PlcInt	Запрос целого переменного PLC из памяти PLC. Данные с плавающей запятой.	Адрес двойного слова: численно, перед ним карактер # Значение: #0...#9999	#0
PlcDouble	Запрос переменного с плавающей запятой PLC из памяти PLC. Данные с плавающей запятой.	Адрес переменного с плавающей запятой: численно, перед ним карактер # Значение: #0...#9998	#0
DirectPlcBit	Запрос бинарного переменного PLC из памяти Time Slice. Данные с плавающей запятой.	Адрес двойного слова: численно, перед ним карактер # Значение: #0... #<PLCNVRAM-1>	Адрес бита: численно, перед ним карактер # Значение: #0...#31
DirectPlcInt	Запрос целого переменного PLC из памяти Time Slice. Данные с плавающей запятой.	Адрес двойного слова: численно, перед ним карактер # Значение: #0... #<PLCNVRAM-1>	#0
DirectPlcDouble	Запрос переменного с плавающей запятой PLC из памяти Time Slice. Данные с плавающей запятой.	Адрес переменного с плавающей запятой: численно, перед ним карактер # Значение: #0... #<PLCNVRAM-2>	#0

Символ	Описание	Param1	Param2
ComPosAx	Позиция команды, выданной для измерительной системы одной оси. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0
ComVelAx	Команда скорости, выданная для измерительной системы одной оси. Данные с плавающей запятой в выходной измерительной системе (мм/сек, градус/сек, дюйм/сек).	Номер оси. Значение: #1...#32	#0
FolErrAx	Значение меры ошибки слежения (отставания) на одной оси. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0
CommandAx	Знак команды, выходящей в сторону привода для одной оси. Данные с плавающей запятой в выходной измерительной системе (мм/сек, градус/сек, дюйм/сек).	Номер оси. Значение: #1...#32	#0
ActPosAx	Актуальная позиция одной оси, измеренная от датчика. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0

Символ	Описание	Param1	Param2
PosErrAx	Разность между актуальной позицией, предположенной из позиции команды одной оси и позицией, измеренной с датчика. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0
TachAx	Скорость одной оси, измеренной с датчика. Данные с плавающей запятой в выходной измерительной системе (мм/сек, градус/сек, дюйм/сек).	Номер оси. Значение: #1...#32	#0
TachRealAx	Скорость одной оси, измеренной с датчика. Число импульсов, поступающих с датчика за время Time Slice как данные с плавающей запятой.	Номер оси. Значение: #1...#32	#0
EGBvTarAx	Скорость мастер-оси EGB. Данные с плавающей запятой в выходной измерительной системе (мм/сек, градус/сек, дюйм/сек).	Нужно задавать номер оси аппликата! Значение: #1...#32	#0
EGBcCurrAx	Скорость оси апплеката EGB. Данные с плавающей запятой в выходной измерительной системе (мм/сек, градус/сек, дюйм/сек).	Номер оси. Значение: #1...#32	#0
SyncErrAx	Отклонение позиции оси апплеката от мастер-оси, в случае EGB, или синхрона gantry. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0

Символ	Описание	Param1	Param2
PitchAx	Значение компенсации шага резьбы, действующего мгновенно на данной оси. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0
StraightnessAx	Значение компенсации прямолинейности, действующей мгновенно на данной оси. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0
CompenValAx	Сумма всех компенсаций, действующих мгновенно на данной оси. Данные с плавающей запятой в выходной измерительной системе (мм, градус, дюйм).	Номер оси. Значение: #1...#32	#0
ComPosSp	Позиция команды, выданной для измерительной системы одного шпинделя в случае замкнутого петля позиции. Данные с плавающей запятой, заданные в оборотах шпинделя.	Номер шпинделя. Значение: #1...#16	#0
ComVelSp	Команда скорости, выданной для измерительной системы одного шпинделя в случае замкнутого петля позиции. Данные с плавающей запятой, заданные в оборот/сек шпинделя.	Номер шпинделя. Значение: #1...#16	#0

Символ	Описание	Param1	Param2
FolErrSp	Мера ошибки слежения (отставания) на одном шпинделе в случае замкнутого петля позиции. Данные с плавающей запятой, заданные в оборотах шпинделя.	Номер шпинделя. Значение: #1...#16	#0
CommandSp	Знак команды, идущий в сторону привода шпинделя. Данные с плавающей запятой, заданные в оборот/сек шпинделя.	Номер шпинделя. Значение: #1...#16	#0
ActPosSp	Актуальная позиция шпинделя. Данные с плавающей запятой, заданные в оборотах шпинделя.	Номер шпинделя. Значение: #1...#16	#0
PosErrSp	Разность между актуальной позицией, предположенной из позиции команды одного шпинделя и позицией, измеренной с датчика в случае замкнутого петля позиции. Данные с плавающей запятой, заданные в оборотах шпинделя.	Номер шпинделя. Значение: #1...#16	#0
TachSp	Скорость одного шпинделя, измеренная с датчика. Данные с плавающей запятой, заданные в оборот/сек шпинделя.	Номер шпинделя. Значение: #1...#16	#0
NActSp	Скорость одного шпинделя, измеренная с датчика. Данные с плавающей запятой, заданные в оборот/мин шпинделя.	Номер шпинделя. Значение: #1...#16	#0

Символ	Описание	Param1	Param2
NSetSp	Команда чисел оборотов, изменённая на вводе оператора шпинделя с override-ом. Данные с плавающей запятой, заданные в оборот/мин шпинделя.	Номер шпинделя. Значение: #1...#16	#0
NCommandSp	Знак команды, идущий в сторону привода шпинделя, изменённая оператором. Данные с плавающей запятой, заданные в оборот/мин шпинделя.	Номер шпинделя. Значение: #1...#16	#0
SyncVTargetSp	Скорость мастер-оси в состоянии синхрона шпинделя. Данные с плавающей запятой, заданные в оборот/сек шпинделя.	Нужно задавать номер оси аппликата! Значение: #1...#16	#0
SyncVSlaveSp	Скорость оси аппликата в состоянии синхрона шпинделя. Данные с плавающей запятой, заданные в оборот/сек шпинделя.	Номер шпинделя. Значение: #1...#16	#0
SyncErrSp	Значение отклонения позиции оси аппликата от мастер-оси в состоянии синхрона шпинделя. Данные с плавающей запятой, заданные в оборотах шпинделя.	Номер шпинделя. Значение: #1...#16	#0
Measured	Для внутреннего использования.		
RealTime	Потребность времени всех задач реального времени в µсек-ах. Данные с плавающей запятой.	#0	#0

Символ	Описание	Param1	Param2
HardwareTime	Время обновления I/O в µсек-ах. Данные с плавающей запятой.	#0	#0
ChannelsTime	Потребность времени оператора каналов в µсек-ах. Данные с плавающей запятой.	#0	#0
AxesTime	Потребность времени оператора оси в µсек-ах. Данные с плавающей запятой.	#0	#0
PlcTime	Потребность времени модуля Int0 PLC в µсек-ах. Данные с плавающей запятой.	#0	#0
PlcCycle	Потребность времени PLC Main в µсек-ах. Данные с плавающей запятой. Содержит прерывания real time.	#0	#0
CANErr	Для внутреннего использования.		
TSliceErr	Для внутреннего использования.		
CNCBufferCount	Число кадров в буфере. Данные с плавающей запятой.	Номер канала. Значение: #1...#8	#0
RotaryAx	Значение цикличных позиций в случае вращения осей. Данные позиции с плавающей запятой, заданные в градусах.	Номер оси. Значение: #1...#32	#0
TachRealSp	Скорость одной оси, измеренная с датчика. Число импульсов, поступающих с датчика в качестве данных с плавающей запятой за время Time Slice.	Номер шпинделя. Значение: #1...#16	#0

Символ	Описание	Param1	Param2
NCTDriveMess	Данные с плавающей запятой, полученные от привода NCT.	Номер вспомогательного привода или привода шпинделя: #1...#48	#0: число оборотов двигателя [об/мин]
			#1: ток двигателя [A],
			#2: относительный ток двигателя I/I_n [%]
			#3: напряжение шин [в]
			#4: температура двигателя [°C]
			#5: мощность двигателя [Кватт]

6.18 Блочное писание и чтение внутренних переменных NC

Командой блочного чтения можно зачитать один блок памяти NC в область памяти PLC. Командой блочного писания можно записать один блок памяти PLC в данную область памяти NC.

6.18.1 Чтение блока памяти NC: команда MR

Командой можно вычитать различные данные, сохранённые в NC.

Ввод и вывод команды MR

Команда MR имеет один

разрешающий ввод.

Вычитание выполняется в состоянии разрешающего ввода верно.

Для команды MR *можно конфигурировать*

вывод.

Вывод примет состояние ВЕРНО, если ввод команды ВЕРНО и выполнил операцию чтения.

К выводу можно присоединить и новые команды.

☞ ***Внимание!*** *Выполнение команды может потребовать несколько циклов PLC, поэтому его разрешающий ввод нужно держать в состоянии верно до тех пор, пока на выводе команды не появится состояние верно. После этого разрешающий ввод надо выключить.*

Вводные параметры команды MR

– **Data Address**:

Предел значения: 0...9999

Начальный адрес блока памяти PLC, выделённый для команды численно, или символично. Возможно задача индексированно. В этот блок записываются сообщения о выполнения команды и сюда же зачитываются данные из NC.

– **Function Code**:

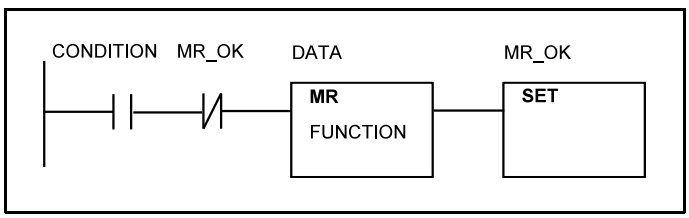
Определяет то, что с какой области памяти NC выполнилось чтение. Значение различных кодов функции рассматриваются в следующих главах.

– **Output**:

В разрешённом состоянии к коробке команд можно присоединить дальнейшие команды.

– Remark:

Замечание.



В памяти PLC в зависимости от длины зачитаемых данных нужно занимать область для их сохранения. На адрес, выделённый в команде переменной Data Address и на последующий адрес командой записываются данные о выполнении.

– **Адрес**: начинается с адреса, заданного параметром Data Address. Код выволнения записывается всегда в этот адрес.
– **Код выполнения**: выводные данные, создаются в ходе выполнения команды.
– **Длина данных**: вводные данные. Определяют *длину специфической по функциям области данных* для команды в двойном слове (DWORD).
– **Специфическая по функциям область**: От этого адреса начинаются специальные коды, определённые кодом Функции. Эти данные могут быть:
  – вводные данные, конфигурирующие чтение дальше, и
  – выводные данные чтения.

Адрес	Данные
0	Код вывполнения
1	Длина данных (вводные данные)
2	Специфическая по функциям область
3	
...	...
n	...

☞***Внимание!*** *Длиной данных считается всегда суммарное число конфигурирующих вводных данных и зачитаемых данных в единицах DWORD!*

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных
..	Специфические по функциям ошибки
n	

6.18.2 Писание блока памяти NC: команда MW

Командой можно изменить различные данные, сохранённые в NC.

Ввод и вывод команды MW

Команда MW имеет один

разрешающий ввод.

Писание выполняется в состоянии разрешающего ввода верно.

Для команды MW *можно конфигурировать*

вывод.

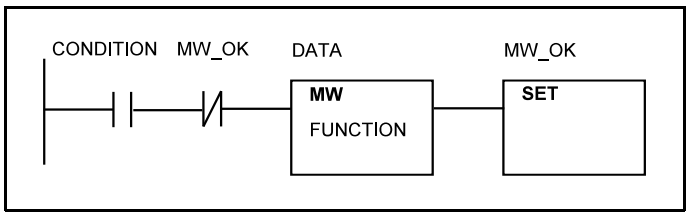
Вывод примет состояние ВЕРНО, если ввод команды ВЕРНО и выполнил операцию писания.

К выводу можно присоединить и новые команды.

☞***Внимание!*** *Выполнение команды может потребовать несколько циклов PLC, поэтому его разрешающий ввод нужно держать в состоянии верно до тех пор, пока на выводе команды не появится состояние верно. После этого разрешающий ввод надо выключить.*

Вводные параметры команды MW

– **Data Address**:
  Предел значения: 0...9999
  Начальный адрес блока памяти PLC, выделённый для команды численно, или символично. Возможно задача индексированно. В этот блок записываются сообщения о выполнения команды и отсюда же записываются данные в NC.
– **Function Code**:
  Определяет то, что в какую область памяти NC выполнилось писание. Значение различных кодов функции рассматриваются в следующих главах.
– **Output**:
  В разрешённом состоянии к коробке команд можно присоединить дальнейшие команды.
– Remark:
  Замечание.



В памяти PLC в зависимости длины выписываемых данных нужно занимать область для их сохранения. На адрес, выделённый в команде переменной Data Address и на последующий адрес командой записываются данные о выполнении.

– **Адрес**: начинается с адреса, заданного параметром Data Address. Код выволнения записывается всегда в этот адрес.
– **Код выполнения**: выводные данные, создаются в ходе выполнения команды.
– **Длина данных**: вводные данные. Определяют *длину специфической по функциям области данных* для команды в двойном слове (DWORD).
– **Специфическая по функциям область**: От этого адреса начинаются специальные коды, определённые кодом Функции. Эти данные могут быть:
  – вводные данные, конфигурирующие писание дальше, и
  – выводимые данные.

Адрес	**Данные**
0	Код вывполнения
1	Длина данных (вводные данные)
2	Специфическая по функциям область
3	
...	...
n	...

☞***Внимание!*** *Длиной данных считается всегда суммарное число конфигурирующих вводных данных и выписываемых данных в единицах DWORD!*

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных
3	Защищённое от записи
...	Специфические по функциям ошибки
n	

6.19 Перемещение данных между незабывающим магазином и памятью PLC

Может возникнуть необходимость сохранить какой-то блок переменных PLC в незабывающей памяти, а также прочитать его из неё.
Некоторые типы управления автоматически сохраняют блок данных, начиная с адреса PLCNVRAM с длиной 1024 двойного слова, с их сохранением программисту PLC не приходится заниматься.
Другие типы управления *не поддерживают автоматическое сохранение* блока данных, начиная с адреса PLCNVRAM с длиной 1024 двойного слова *после выключения*, а нужные данные надо сразу сохранить из программы PLC, с помощью соответствующего блочного шрифта, при каждом их изменений. Таким же образом, в соответствующий момент, например, после включения надо зачитать сохранённый блок памяти. Для желаемых сохранить переменных PLC и в этом случае обеспечивается управлением объём памяти с длиной 1024 двойного слова.

Опознавательный знак первой переменной в незабывающем магазине: 0,
а опознавательный знак последней 1023.

Выполнение команд MW, MR, сохраняющие и зачитающие переменные PLC, занимает время на несколько циклов PLC больше, поэтому всегда надо дождаться, чтобы их вывод попал в состояние ПРАВДА.

6.19.1 Чтение данных переменных PLC из незабывающего магазина

Команда: MR
Код функции: 10

Зачитание переменных блока, заданного начальным опознавательным числом и количеством штук переменных из незабывающего магазина к переменным PLC.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 3
2	Опознавательный знак первой зачитаемой переменной в незабывающем магазине: 0...1023
3	Начальный адрес первой зачитаемой переменной в памяти PLC
4	Число зачитаемых переменных: 1...1024

*: эти данные не надо заполнять
К длине данных надо записать 3. В случае зачитания нескольких переменных, память PLC заполняется непрерывно от начального адреса.

Выводные параметры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных: 3
2	Опознавательный знак первой зачитаемой переменной в незабывающем магазине 0...1023
3	Начальный адрес первой зачитаемой переменной в памяти PLC
4	Число зачитаемых переменных: 1...1024

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных: не 3
4	Ошибочная контрольная сумма магазина
10	Опознавательный знак переменных не попадает среди 0...1023
11	Начальный адрес первой переменной не болшее или равно PLCNVRAM
12	Ошибка числа данных: (опознавательный знак первой зачитаемой переменной)+ (число зачитаемых переменных)>1024

6.19.2 Запись данных переменных PLC в незабывающий магазин

Команда: MW
Код функции: 11

Запись переменных блока, заданного начальным адресом переменной и количеством штук переменных из памяти PLC в незабывающий магазин. Ввыход команды возвращается кодом выполнения 0 только после того, когда после выписания вычисляется и контрольная сумма магазина.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 3
2	Опознавательный знак первой выписываемой переменной в незабывающем магазине: 0...#1023
3	Начальный адрес первой выписываемой переменной в памяти PLC
4	Число выписываемых переменных: 1...1024

*: эти данные не надо заполнять
К длине данных надо записать 3. В случае выписывания нескольких переменных, незабывающий магазин заполняется непрерывно, начиная от опознавательного знака первой переменной.

Выводные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 3
2	Опознавательный знак первой выписываемой переменной в незабывающем магазине: 0...#1023
3	Начальный адрес первой выписываемой переменной в памяти PLC
4	Число выписываемых переменных: 1...1024

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных: не 3
10	Опознавательный знак переменной не попадает среди 0...1023
11	Начальный адрес первой переменной не болшее или равно PLCNVRAM
12	Ошибка числа данных: (опознавательный знак первой выписываемой переменной)+ (число выписываемых переменных)>1024

6.20 Чтение и писание параметров из программы PLC

Программа PLC с помощью команд MR и MW может достигать параметры NC для чтения, или писания.
В ходе чтения или писания необходимо принимать во внимание следующие точки зрения:
изображение данных параметром,
является ли параметр глобальным, или индексированным.
С точки зрения изображения данных можно различать параметры:
– **битные**,
– ***DWORD: с фиксированной запятой, двойного слова***,
– ***double: с плавающей запятой*** по изображению.
Потребность в площади битных параметров из PLC будет DWORD. Различные битные параметры будут изображены на битах DWORD 0...7.
Параметр можно индексировать:
– **по группам станков**,
– **по каналам**,
– **по осям**, или
– **по шпинделям**.
Индексированные параметры не имеют отдельного опознавательного числа.

На какой-то параметр можно ссылаться его опознавательным числом и индексом. В случае ссылки на глобальный параметр значение индекса 0. Писание, или чтение блочного параметра (одновременное писание, или чтение нескольких параметров) возможно только для не глобальных параметров. Например, чтение различных параметров по шпинделям может осуществляться одновременно для всех шпинделей.
Описание параметров содержит опознавательное число (Nnnnn), формат (бит, DWORD, double) отдельных параметров, далее, что на основании чего индексируется.

6.20.1 Чтение параметров типа DWORD

Команда: MR
Код функции: 31

Чтение параметров формата DWORD, заданных опознавательным числом и начальным идексом среди переменных PLC. Число прочитаемых данных задаётся параметром длины данных команды.

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 3, не более n+2
2	Опознавательное число прочитаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го прочитаемого параметра
4	значение 1-го параметра*

Адрес	Данные
5	значение 2-го параметра*
...	...
n+3	значение n-го параметра*

*: данные не надо заполнять
На длину данных надо написать не менее 3, при этом прочитается только 1 двойное слово.
В случае прочтения больше данных PLC непрерывно наполняет память.
Если например, желаем прочитать параметр, индексированный по осям, и желаем узнать параметр 2-й и 3-й оси, длина данных=4 и начальный индекс=2.

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 3, не более n+2
2	Опознавательное число прочитаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го прочитаемого параметра
4	значение 1-го параметра
5	значение 2-го параметра
...	...
n+3	значение n-го параметра

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных: < 3, или > (индекс последней переменной блока параметров - заданное число начального индекса) + 2
4	Контрольная сумма накопителя ошибочная
30	Ссылка на несуществующий параметр
31	Параметр не глобальный (индекс параметра = 0)

Код	Значение
32	Параметр глобальный (индекс параметра> 0)
33	Непригодный код чтения: формат параметра не DWORD

6.20.2 Чтение параметров типа Double

Команда: MR
Код функции: 32

Чтение параметров формата double, заданных опознавательным числом и начальным идексом среди переменных PLC. Число прочитаемых данных задаётся параметром длины данных команды.

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 4, не более 2n+2
2	Опознавательное число прочитаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го прочитаемого параметра
4	значение 1-го параметра*
5	
6	значение 2-го параметра*
7	
...	...
2n+2	значение n-го параметра*
2n+3	

*: данные не надо заполнять
На длину данных надо написать не менее 4, при этом прочитается только 1 данные с плавающей запятой. В случае прочтения больше данных PLC непрерывно наполняет память.
Если например, желаем прочитать параметр, индексированный по осям, и желаем узнать параметр 2-й и 3-й оси, длина данных=6 и начальный индекс=2.

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 4, не более 2n+2
2	Опознавательное число прочитаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го прочитаемого параметра
4	значение 1-го параметра
5	
6	значение 2-го параметра
7	
...	...
2n+2	значение n-го параметра
2n+3	

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных: < 3, или > (индекс последней переменной блока параметров - заданное число начального индекса)×2 + 2
4	Контрольная сумма накопителя ошибочная
30	Ссылка на несуществующий параметр
31	Параметр не глобальный (индекс параметра= 0)
32	Параметр глобальный (индекс параметра> 0)
33	Непригодный код чтения: формат параметра не double

6.20.3 Писание параметров типа DWORD

Команда: MW
Код функции: 34

Писание параметров формата DWORD, заданных опознавательным числом и начальным идексом среди переменных PLC. Число писаемых данных задаётся параметром длины данных команды.

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 3, не более n+2
2	Опознавательное число писаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го писаемого параметра
4	значение 1-го параметра
5	значение 2-го параметра
...	...
n+3	значение n-го параметра

*: данные не надо заполнять
На длину данных надо написать не менее 3, при этом пишется только 1 двойное слово. В случае писания больше данных PLC непрерывно наполняет память.
Если например, желаем писать параметр, индексированный по осям, и желаем узнать параметр 2-й и 3-й оси, длина данных=4 и начальный индекс=2.

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 3, не более n+2
2	Опознавательное число писаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го писаемого параметра
4	значение 1-го параметра
5	значение 2-го параметра
...	...

Адрес	Данные
n+3	значение n-го параметра

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных: < 3, или > (индекс последней переменной блока параметров - заданное число начального индекса) + 2
30	Ссылка на несуществующий параметр
31	Параметр не глобальный (индекс параметра= 0)
32	Параметр глобальный (индекс параметра > 0)
33	Непригодный код писания: формат параметра не DWORD

6.20.4 Писание параметров типа Double

Команда: MR
Код функции: 35

Писание параметров формата double, заданных опознавательным числом и начальным идексом среди переменных PLC. Число писаемых данных задаётся параметром длины данных команды.

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 4, не более 2n+2
2	Опознавательное число писаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го писаемого параметра
4	значение 1-го параметра
5	
6	значение 2-го параметра
7	
...	...
2n+2	значение n-го параметра
2n+3	

*: данные не надо заполнять
На длину данных надо написать не менее 4, при этом пишется только 1 данные с плавающей запятой. В случае писания больше данных PLC непрерывно наполняет память.
Если например, желаем писать параметр, индексированный по осям, и желаем узнать параметр 2-й и 3-й оси, длина данных=6 и начальный индекс=2 .

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 4, не более 2n+2
2	Опознавательное число писаемого параметра :<число>
3	Начальный индекс параметра: =0: глобальный параметр >0 индекс 1-го писаемого параметра
4	значение 1-го параметра
5	
6	значение 2-го параметра
7	
...	...
2n+2	значение n-го параметра
2n+3	

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных: < 3, или > (индекс последней переменной блока параметров - заданное число начального индекса)×2 + 2
4	Контрольная сумма накопителя ошибочная
30	Ссылка на несуществующий параметр
31	Параметр не глобальный (индекс параметра= 0)
32	Параметр глобальный (индекс параметра> 0)
33	Непригодный код писания: формат параметра не double

6.21 Выделение программы к выполнению

Из программы PLC можно выделить к выполнению произвольную программу деталей, имеющуюся в памяти управления.

6.21.1 Выделение программы, заданной номером программы к автоматическому выполнению

Команда: MW
Код функции: 40

Выделяется к выполнению в автоматическом режиме, в заданном канале программа деталей, заданная со своим четырёх-, или восьмизначным номером программы (Onnnn, или Onnnnnnnn). Операция писания выполняется только тогда, то есть выход команды будет верной только тогда, когда в автоматическом режиме не будет прогон программы.

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 2
2	Номер программы: <число>
3	Номер канала: 1...8

*: данные не надо заполнять
На длину данных надо написать всегда 2.

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 2
2	Номер программы: <число>
3	Номер канала: 1...8

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных: не 1
40	Несуществующая программа: заданной программы нет в накопителе

6.22 Команды сетевой коммуникации

PLC может принимать данные программы, а также может посылать через сеть Ethernet. Коммуникация осуществляется через протокол UDP. Построение одной рамы Ethernet UDP/IP:

Заголовок Ethernet UDP/IP	**Посланные, или принятые данные**			**CRC**
	...	...	...	

PLC работает посланными, или принятыми данными.

6.22.1 Открыть сетевую связь

Команда: MW
Код функции: 60

Входные параметры:

Адрес	**Данные**
0	Код выполнения *
1	Длина данных: 5
2	Опознавательное число связи: всегда 1
3	Код ошибки Widows *
4	Протокол: 1: UDP 2: TCP Сервер 3: TCP клиент
5	IP адрес целевого средства. Pl: 192.168.1.12 = #$C0A8010C Его значение не должно быть: 0.0.0.0 и 255.255.255.255
6	Номер порта, по которому желаем общаться. Рекомендуемые значения: 49202...49205

*: данные не надо заполнять
На длину данных надо написать всегда 5.

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 5
2	Опознавательное число связи: всегда 1
3	Код ошибки Widows: если код выполнения 61, сюда попадает сообщение об ошибке посланное от Windows декодировано дальше (см. Windows Sockets Error Codes) *
4	Протокол: 1: UDP 2: TCP Сервер 3: TCP клиент
5	IP адрес целевого средства. Pl: 192.168.1.12 = #$C0A8010C Его значение не должно быть: 0.0.0.0 и 255.255.255.255
6	Номер порта, по которому желаем общаться. Рекомендуемые значения: 49202...49205

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных
60	Недействительное опознавательное число связи
61	Создание связи безуспешное
66	Входящие параметры ошибочные

6.22.2 Закрыть сетевую связь

Команда: MW
Код функции: 61

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 1
2	Опознавательное число связи: всегда 1

*: данные не надо заполнять
На длину данных надо написать всегда 2.

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 1
2	Опознавательное число связи: всегда 1

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных
60	Недействительное опознавательное число связи

6.22.3 Принимать сетевой пакет данных

Команда: MR
Код функции: 62

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 3-5
2	Опознавательное число связи: всегда 1
3	Начальный адрес блока данных в памяти PLC
4	Длина блока данных: столько регистров PLC нужно будет для приёма данных 0 < значение< = 350
5	Код ошибки Widows **
6	IP адрес средства источника. Например: 192.168.1.12 = #$C0A8010C **
7	Номер порта средства источника **

*: данные не надо заполнять
**: Даты опции, данные не надо заполнять

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 3-5
2	Опознавательное число связи: всегда 1
3	Начальный адрес блока данных в памяти PLC
4	Длина блока данных: столько регистров PLC нужно будет для приёма данных 0 < значение< = 350
5	Код ошибки Widows: если код выполнения 62, сюда попадает сообщение об ошибке, посланное от Windows декодировано дальше (см. Windows Sockets Error Codes)
6	IP адрес средства источника. Например: 192.168.1.12 = #$C0A8010C

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных
60	Недействительное опознавательное число связи
62	В ходе приёма данных произошла ошибка
63	Заданная связь не открыта
64	Длина блока данных ошибочная: =0, или >350
65	Не поступили данные

6.22.4 Посылать сетевой пакет данных

Команда: MW
Код функции: 63

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 3-5
2	Опознавательное число связи: всегда 1
3	Начальный адрес блока данных в памяти PLC
4	Длина блока данных: столько регистров PLC данных будет послано 0 < значение < = 350
5	Код ошибки Widows **
6	IP адрес целевого средства. Например: 192.168.1.12 = #$C0A8010C **
7	Номер порта целевого средства: Например: 49202 **

*: данные не надо заполнять
**: Даты опции, данные надо заполнять только в случае использования UDP

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 3-5
2	Опознавательное число связи: всегда 1
3	Начальный адрес блока данных в памяти PLC
4	Длина блока данных: столько регистров PLC данных будет принято 0 < значение <= 350
5	Код ошибки Widows: если код выполнения 62, сюда попадает сообщение об ошибке, посланное от Windows декодировано дальше (см. Windows Sockets Error Codes)
6	IP адрес целевого средства. Например: 192.168.1.12 = #$C0A8010C

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных
60	Недействительное опознавательное число связи
62	В ходе посылки данных произошла ошибка
63	Заданная связь не открыта
64	Длина блока данных ошибочная: =0, или >350

6.23 Открыть окно на дисплее управления

Окна, изображаемые из системы меню управления вмешательством оператора, можно выбирать из программы PLC и командой MW.

Команда: MW
Код функции: 70

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 2-4
2	Опознавательное число окна: см. таблицу
3	Номер канала: =0: независимое от канала окно =1...8: изображать окно с данными, действительными в канале
4	Параметр1: см. таблицу
5	Параметр2: см. таблицу

*: Данные не надо заполнять

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 2-4
2	Опознавательное число окна: см. таблицу
3	Номер канала: =0: независимое от канала окно =1...8: изображать окно с данными, действительными в канале
4	Параметр1: см. таблицу
5	Параметр2: см. таблицу

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции

Код	Значение
2	Недействительная длина данных
6	Недействительный номер канала
70	Недействительный номер окна
71	Нет связи с индикатором (индикатор не готов к приёму)

Опознание окон, изображаемые из программы PLC

Имя окна	Опознав. число	Параметр 1	Параметр 2
Алфавитно-цифровая клавиатура	100	-	-
Клавиши PLC	110	-	-
Клавиши выбора окон	120	-	-
Станочная панель оператора	130	-	-
Активные коды G с видом, установленным последним		-1	
Активные коды G	1000	0	-
Активные коды G с текстом	1000	1	-
Все коды G	1000	2	-
Все коды G с текстом	1000	3	-
Активные коды M с видом, установленным последним		-1	
Активные коды M	1010	0	-
Активные коды M с текстом	1010	1	-
Все коды M	1010	2	-
Все коды M с текстом	1010	3	-
FST	1020	-	-
Все шпиндели	1030	-	-
Индивидуальный ввод кадра	1040	-	-
Список программ	1050	-	-
Окно каталогов левой стороны	1090	-	-
Окно каталогов правой стороны	1091	-	-

Имя окна	Опознав. число	Параметр 1	Параметр 2
Список текущих программ (главных и подпрограмм)	1110	-	-
Все сообщения	1130	-	-
Счётчик времени/заготовок	1150	-	-
Позиция	3000	-	-
Замер заготовок	3011	-	-
Замер инструментов	3012	-	-
Таблица коррекций фрезерных инструментов	3020	-	-
Таблица коррекций токарных инструментов	3031	-	-
Таблица коррекций токарных инструментов 2	3032	-	-
Окно смещения нулевой точки	3040	-	-
Таблица обращения инструментами	4010	-	-
Таблица мест инструментов	4020	-	-
Окно индивидуального редактирования инструментов	4030	Номер магазина	Номер кармана
Таблица формы инструмента	4040	-	-
Обобщённая таблица стойкости инструментов	4050	-	-
Локальные макропеременные #1...#33	5010	-	-
Глобальные макропеременные #100...#499	5020	-	-
Глобальные макропеременные #500...#999	5030	-	-
Калькулятор	6000	-	-
Расчёт скорости резания	6010	-	-
Счётчик шагов	6020	-	-
Конвертер DXF	6030	-	-

Имя окна	Опознав. число	Параметр 1	Параметр 2
Установка позиции ориентировки шпинделя	6040	-	-
Установка сдвига по фазе шпинделя	6041	-	-
Калибровка замера заготовки	6060	-	-
Замер поверхности заготовки	6061	-	-
Замер углов заготовки	6062	-	-
Замер кармана/плеча, отверстия/вала	6063	-	-

6.24 Выписка данных привода

Значения потребляемого двигателями тока можно выводить в индикаторе позиции и в окне FST. Число оборотов двигателей шпинделей выводятся в окне FST. Данные приводов производства NCT автоматически принимаются управлением от приводов.
В случае приводов иного производства, если программа PLC может прочесть из привода различные данные, или например, через преобразователя A-D может прочесть данные тока, с помощью команды 81 может их передать для NC для индикации.

Команда: MW
Код функции: 81

Входные параметры:

Адрес	Данные
0	Код выполнения*
1	Длина данных: 3
2	Адрес привода: значение, установленное параметром N0501 Axis Input Address, или N0601 Spindle Input Address
3	Тип переданных данных: 0 – Число оборотов двигателя [1/мин] 1 – Мгновенный ток двигателя [10мА] (125 = 1,25А) 2 – Нагрузка двигателя (относительный ток) [%] 3 – Напряжение шины DC [в] 4 – Температура двигателя [°C] 5 – Мгновенная мощность двигателя [кВт]
4	Значение переданных данных (DWORD)

*: данные не надо заполнять

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 3
2	Адрес привода: значение, установленное параметром N0501 Axis Input Address, или N0601 Spindle Input Address

Адрес	Данные
3	Тип переданных данных: 0 – Число оборотов двигателя [1/мин] 1 – Мгновенный ток двигателя [10мА] (125 = 1,25А) 2 – Нагрузка двигателя (относительный ток) [%] 3 – Напряжение шины DC [в] 4 – Температура двигателя [°C] 5 – Мгновенная мощность двигателя [кВт]
4	Значение переданных данных (DWORD)

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных
80	Недействительный адрес привода
81	Недействительный тип данных
82	Недействительное значение переданных данных

6.25 Запись данных позиции

Программа PLC может изменить позицию, регистрированную в обращении осями, если
– на оси имеется открытый контур регулирования позиции (AN_OPNA=1), и
– запрещён приём позиции от средства измерителя хода (AP_EFD=1, или бит #4 EFD=1 параметра N0514 Servo Control).

Под действием команды оператор осями ***зарегистрирует для данной оси*** состояние ***референтная точка принета*** (AN_RPE=1).

Командой можно пользовать например тогда, когда посадим вращающуюся ось на диск Хирта (Hirth) и учёт позиций, сохранение выполняется программой PLC. Таким образом можно избежать приём референтной точки после включения. Особенно полезно это применять, когда один датчик выполняет измерение позции нескольких осей.

Командаs: MW
Код функции: 90

Входные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 3
2	Индекс оси (0, ..., 31)
3	Записаемая позиция в станочной системе координат (double)
4	

*: данные не надо заполнять

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 3
2	Индекс оси (0, ..., 31)
3	Записаемая позиция в станочной системе координат (double)
4	

Код выполнения может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных
30	Недействительный индекс оси: <0, или >31
90	Внутреняя ошибка, командой пользоваться нельзя
91	Индекс оси указывает на несуществующую ось, или контур регулирования позиции не открыт (AN_OPNA=0), или приём позиции от средсва измерителя хода не запрещён (AP_EFD=0)

6.26 Писание и чтение данных Таблицы оператора инструментов

Функцию оператора инструментов нужно задействовать в управлении в следующих случаях:

– для инструментов желаем применить проверку стойкости,
– на инструменты, хранившихся в магазине, желаем ссылаться из программы деталей не по коду места, а по коду инструмента,
– если смена инструментов, применённых на станке, требует рандомизацию в обращении магазином.

Данные оператора инструментов сохранены в таблицах в NC. Программа PLC с помощью команд MR и MW имеет доступ к данным таблицы, их может читать и писать.

6.26.1 Таблица оператора инструментов

В таблицу оператора инструментов можно записать код T инструментов, использованных в управлении, то есть их номер типа, на который ссылаемся в программе деталей.
В этой таблице можно задавать характеристику отдельных инструментов, то есть, что они имеют нормальный размер, или особый размер, участвуют ли в обращении стойкости лезвии инструмента, или нет, далее, что какой элемент магазина коррекции прикрепляем к ним. Возможно прикрепить и дальнейшие технологические данные к данному инструменту, такие, как число оборотов, подача и т.д.
Таблица оператора инструментов является глобальной,то есть она общая для всех каналов.

Номер данных	Номер типа (T)	Информ. инструм.	Номер окна	Статус стойкости	Счётчик стойкости	Стойкость
1	10002001	UENCV		Просрочен	150	150
2	10002001	UENCV		Использ.	131	200
3	10002001	UENCV		Не использ.	0	170
4	150	SDLCV	2	Нет		
5	3210	UENTV		Сломан	1ч42м26с	2ч30м00с
6	3210	UENTV		Использ.	1ч34м14с	1ч50м30с
...						

Элементы таблицы оператора инструментов изложены ниже. Рядом с отдельными элементами указаны их потребности места:

1 Номер данных (DWORD)

Порядковый номер таблицы. Не редактируется, число строк таблицы определяется параметром. В таблице места инструмента ведётся учёт на основании порядкового номера таблицы оператора инструментов, то есть на основании номера данных.

2 Номер типа (DWORD)

Из программы деталей по адресу T ссылаемся на инструмент с Номером типа. Если одна группа инструментов участвует в обращении стойкости, каждый инструмент группы обозначим тем же Номером типа. Номер типа можно задавать не более 8-ми цифрами:

T: от 1 до 99 999 999.

Функция поиска выбирает всегда тот инструмент с номером данных, значение счётчика стойкости которого максимальное, но ещё не просроченное. На основании таблицы выше на ссылку

T10002001

выбирает инструмент с номером данных 2. Если для инстументов с одинаковым Номером типа значение счётчика стойкости одинаково, выбирает инструмент с меньшим номером данных.

3 Информация об инструменте (DWORD)

Информация об инструменте содержит следующие бинарные информации:

#0 **I** (=0, Invalid): целая строка таблицы оператора инструментов недействительна, можно её удалить

V (=1, Valid): целая строка таблицы оператора инструментов действительна.

#1 **C** (=0, Count): проверка стойкости производится на использование

T (=1, Time): проверка стойкости производится на время.

☞ *Внимание:*

Для инструментов того же Номера типа надо установить того же информацию об инструменте C, или T.

#2 **N** (=0, Normal): инструмент имеет нормальный размер (занимает место одного кармана)

L (=1, Large): инструмент имеет большой размер. занимает место несколько карманов.

#3 **E** (=0, Enabled): целая строка таблицы оператора инструментов редактируемая

D (=1, Disabled): целая строка таблицы оператора инструментов нередактируемая.

#4 Если статус стойкости говорит о том, что данный инструмент не участвует в обращении стойкости и этот бит

U (=0, Unsearchable): для этого инструмента NC не ищет

S (=1, Searchable): для этого инструмента NC ищет.

☞ ***Замечание:*** *три столбца выше Таблицы оператора инструментов всегда существует, если пользуемся Таблицой оператора инструментов. Все остальные столбцы попа-*

дают или не попадают в таблицу параметром. путём альтернативного отбора. Если на экране горизонтально прокручивать таблицу,эти три столбца выше будут видны всегда на левой стороне.

4 Номер окна (DWORD)

Если в Информации об инструменте определяем особый размер по ширине инструмента с отметкой “L”, необходимо определить форму инструмента. Форму особенно широкого инструмента (занятие кармана) изложены в Таблицу формы инструмента. Номер формы указывает на соответствующую строкуТаблицы формы инструмента. Если значение номера формы 0, инструмент занимает место 1 кармана. Если в Информации об инструменте определяем номальный инструмент со знаком “N”, значение номера формы недействительное.

5 Статус стойкости (DWORD)

Статус стойкости содержит следующие коды:

=0: Нет (Not performed)

Инструмент не участвует в обращении стойкости, однако, если в таблице Информации об инструменте обозначен со знаком “S”, алгорифм поиска по Номеру типа T учитывает инструмент, иначе нет.

=1: Не использован (Not used)

Инструмент участвует в обращении стойкости, но ещё не был в употреблении, то есть значение счётчика стойкости =0, этот статус будет выведён. Алгорифм поиска принимает во внимание.

=2: Использованный (Used)

Инструмент участвует в обращении стойкости и значение счётчика стойкости меньше, чем заданная стойкость, этот статус будет выведён. Алгорифм поиска принимает во внимание.

=3: Просрочен (Expired)

Инструмент участвует в обращении стойкости и значение счётчика стойкости достиг заданную стойкость, этот статус будет выведён. На этот инструмент алгорифм в будущем поиск не ведёт.

=4: Сломан (Broken)

Если PLC для данного инструмента отмечает поломку, этот статус будет выведён. Для алгорифма поиска это означает то же самое, как статус “Просрочен”. Это относится и для тех инструментов, которые не участвуют в обращении стойкости.

6 Счётчик стойкости (DWORD)

Если в таблице Информации об инструменте включена проверка стойкости и установлена на C, производится учёт выполнения смены инструмента. Каждый раз, когда после выполнения кода M06, или T появится значок FIN, значение счётчика увеличивается на 1. Как только значение счётчика стойкости достигает значение, определённое в Стойкости, статус инструмента примет состояние “Просрочен”. То обстоятельство, что учёт ведётся по M06, или по T, решается параметром.

Если в таблице Информации об инструменте включена проверка стойкости и установлена на T, наблюдение за стойкостью происходит по времени. Время измеряется тогда, когда

Находится в состоянии Старт,
значение override не 0,
шпиндель вращается,
и выполняется движение подачи.

Если значение времени, измеренное счётчиком стойкости, достигает значение стойкости, статус инструмента примет состояние “Просрочен”.

7 Стойкость (DWORD)

Стойкость, действительную для данного инсрумента, можно установить на число, или на время. Когда счётчик Стойкости одного инструмента достигает определённое здесь значение, статус Стойкости примет состояние “Просрочен”.

Дальнейшие элементы таблицы

Номер данных	**Номер типа (T)**	**Информ. инструм.**	**Предупр. стойкость**	**H**	**D**	**S**
1	10002001	UENCV	20	1	1	12500
2	10002001	UENCV	12	2	3	11400
3	10002001	UENCV	15	31	31	13000
4	150	SDLCV		12	13	5400
5	3210	UENTV	5м30с	326	326	2500
6	3210	UENTV	4м10с	63	63	2700
...						

8 Предупреждающая сойкость (DWORD)

Если разность между установленной для инструмента Стойкостью и показанием счётчика. Стойкости достигает установленное здесь значение, управлением выводится предупреждение для PLC.

9 H: номер ячейки коррекции длины (DWORD)

В канале фрезера, если инструмент участвует в наблюдении за стойкостью, это номер ячейки коррекции длины, относящийся к инструменту (код H).

10 D: номер ячейки коррекции радиуса (DWORD)

В канале фрезера, если инструмент участвует в наблюдении за стойкостью, это номер ячейки коррекции радиуса, относящийся к инструменту (код D).

11 G: номеря чейки коррекции геометрии (DWORD)

В канале токарного станка, если инструмент участвует в наблюдении за стойкостью, это номер ячейки коррекции геометрии, относящийся к инструменту.

12 W: номер ячейки коррекции износа (DWORD)

В канале токарного станка, если инструмент участвует в наблюдении за стойкостью, это номер ячейки коррекции износа, относящийся к инструменту.

13 Число оборотов шпинделя (DWORD)

Число оборотов шпинделя, относящееся к инструменту в 1/мин-ах.

Дальнейшие элементы таблицы

Номер данных	**Номер типа (T)**	**Информ. инструм.**	**F**	**Пользо-ватель-ские, би-нарный**	**Пользо-ватель-ские 1**	**...**
1	10002001	UENCV	3000			
2	10002001	UENCV	2800			
3	10002001	UENCV	3300			
4	150	SDLCV	650			
5	3210	UENTV	1800			
6	3210	UENTV	1700			
...						

14 Подача (double)

Подача,относящаяся к инструменту в размерности мм/мин, дюйм/мин, или мм/об, дюйм/об.

15 Пользовательские, бинарные данные (DWORD)

По типам инструментов можно задавать 8 шт. бинарных данных, для произвольного назначения.

16 Пользовательские данные (double)

Столбец, пригодный для хранения данных с плавающей запятой, число которых задаётся параметром.

6.26.2 Таблица места инструмента

В Таблицу места инструмента записываются Номер данных инструментов, находящихся в карманах отдельных магазинов, то есть номер соответствующей строки Таблицы оператора инструментов.
Таблица места инструмента является тоже глобальной, общей таблицей для всех каналов.

Порядк. номер	Номер магазина	Номер кармана	Номер данных
1	1	1	0
2	1	2	4
3	1	3	3
...	...	...	...
24	1	24	1
25	2	1	12
26	2	2	28
27	2	3	0
...	...	...	...
40	2	16	62
...	...	...	...
41	10	1	2
42	11	1	0
43	20	1	5
44	21	1	6
...	...	...	...

Таблица места инструмента делится следующим образом:

1 Порядковый номер

Показывает порядковый номер полной Таблицы места инструмента. Совпадает с номером всех имеющихся на станке карманов для инструментов.

2 Номер магазина (DWORD)

Параметром можно принять не более 4-х различных магазинов. Их порядковый номер меняется от 1-го до 4-х. То обстоятельство, что к отдельным магазинам для какие рычаги смены имеют доступ, далее, что рычаги смены какие из шпинделей могут обслуживать, это зависит от механического построения станка.

Параметром можно задавать, что расположение отдельных магазинов имело вид цепи или матрицы, сколько они содержали карманов, далее, что при расположении в виде матрицы в скольких рядах были разложены карманы.

Специальные магазины

Помимо этого, каждый шпиндель, участвующий в обращении инструментами, параметром закрепим к таблице магазинов, далее прикрепим к каждому шпинделю один магазин готовности. Магазины шпинделя и магазины готовности являются магазинами с одним карманом.

Номер магазина 1-го шпинделя 10, а магазина 2-го шпинделя 20 и так далее.

Магазины готовности, закреплённых к шпинделю (например, рычаги инструментов) имеют номер магазина 11, 21 и так далее.

☞ *Замечание.*

Бывает, что инструмент попадает в качестве позиции обработки не в шпиндель, а только в держатель инструмента. Возьмём для примера такой карусельный токарный станок с осями 2x2, у которого в держателях инструмента левой и правой стороны нельзя вращать инструмент. Даже при этом надо определить в обеих каналах по одному виртуальному шпинделю (например: S1, S2), так как указатели коммуникации PLC-NC и вызванные коррекции находятся на учёте управлением на основании магазинов шпинделя и раставляются индексы по номеру шпинделя.

3 Номер кармана (DWORD)

Внутри магазинов номера карманов пастут по одному. Начальный номер отдельных карманов надо задавать параметром.

4 Номер данных (DWORD)

Сюда записывается Номер данных Таблицы оператора инструментов, то есть порядковый номер, указывающий на инструмент, находящийся в кармане.

Если значение Номера данных 0, в кармане нет инструмента.

Если в Номере данных имеется знак “*”, это означает, что карман пустой, но инструмент особого размера занимает место.

6.26.3 Таблица формы инструмента

В Таблице формы инструмента можно задавать форму тех инструментов, которые занимают место в магазине больше одного кармана.

Если инструмент является сверхразмерным, в Таблице оператора инструментов, то

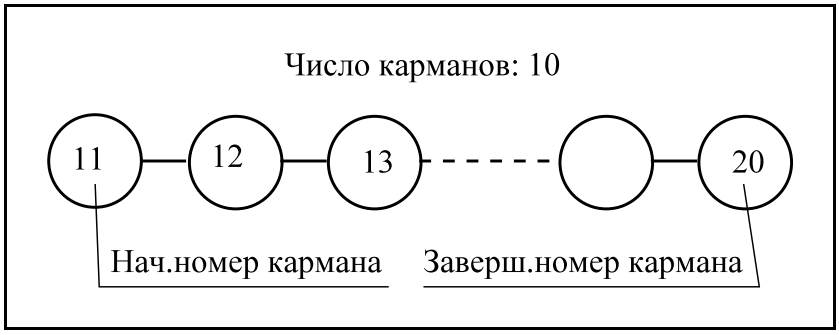
– #2-й бит столбца **Информация об инструменте** надо установить на “L”, далее

– в столбец **Номер формы** следует записать, что в какой строке Таблицы формы инструмента задавали форму инструмента.

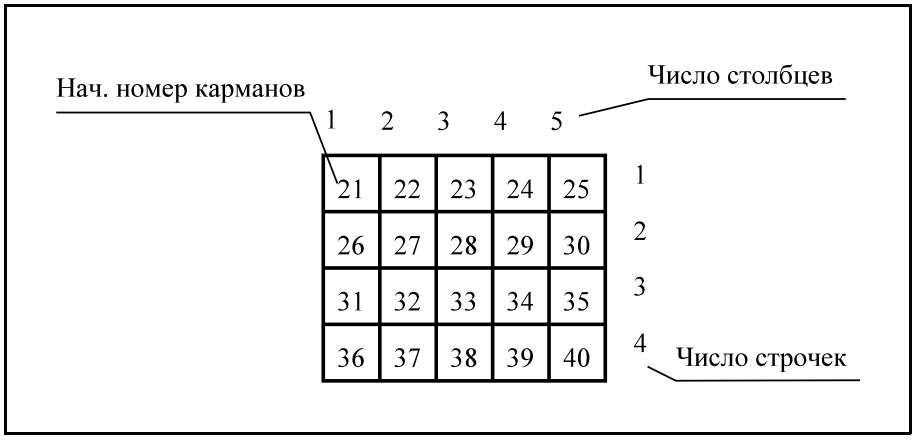
Перед пользованием Таблицы формы инструмента надо задавать параметром тип отдельных магазинов, то есть, что их расположение имеет вид цепи или матрицы.

Помимо этого, параметром следует задавать

– в случае **цепного магазна** начальный номер карманов отдельных магазинов (не обязательно начинать с 1-го), далее число карманов в магазине,



– в случае распложения магазинов в **матрицы** следует задавать начальный номер карманов (не обязательно дать 1), далее число строчек и столбцев, имеющихся в матрице.



Внутри магазина, имеющего распложение в матрицы, управлением истолкуется нумерация карманов, строчек и столбцев таким образом, как это видно на рисунке выше, значит, нумерация начинается с левого верхнего угла и заканчивается в правом нижнем углу.

Таблица формы инструмента является глобальной, действительной для каждого канала. Число элементов Таблицы формы инструмента не более 20. Это означает, что система может обращаться не более 20 инструментами различной формы.

Номер формы	Налево	Направо	Верх	Вниз	Геометрия
1					
...					

1 Номер формы DWORD

Это есть порядковый номер Таблицы формы инструмента. Число, заданное в столбце Номера формы Таблицы оператора инструмента указывает на соответствующую строку Таблицы формы инструмента.

2 Налево, Направо, Верх, Вниз

Столбцы Налево, Направо используем так в случае цепного магазина, как и магазина в матрицы. Заполнение столбцев Верх, Вниз имеет значение только при расположении магазинов инструментов в матрицы. Число, записанное в столбцы показывает, что сколько карманов занимает инструмент в соответствующее направление в размерности

½ карман.

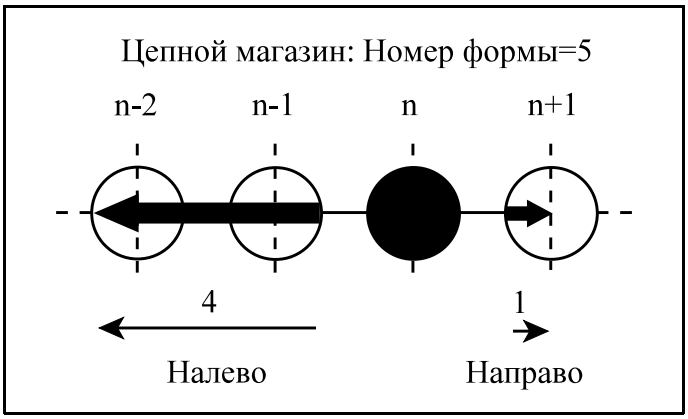
В те карманы, которые занимает какой-то инсрумент только на половину, естественно, уже нельзя разместить больше инструментов. Однако, рядом с этими карманами ещё можно разместить такие инструменты, которые в противоположенное направление занимают место также по ½, 1 ½, 2 ½ карманов.
Истолкование различных направлений

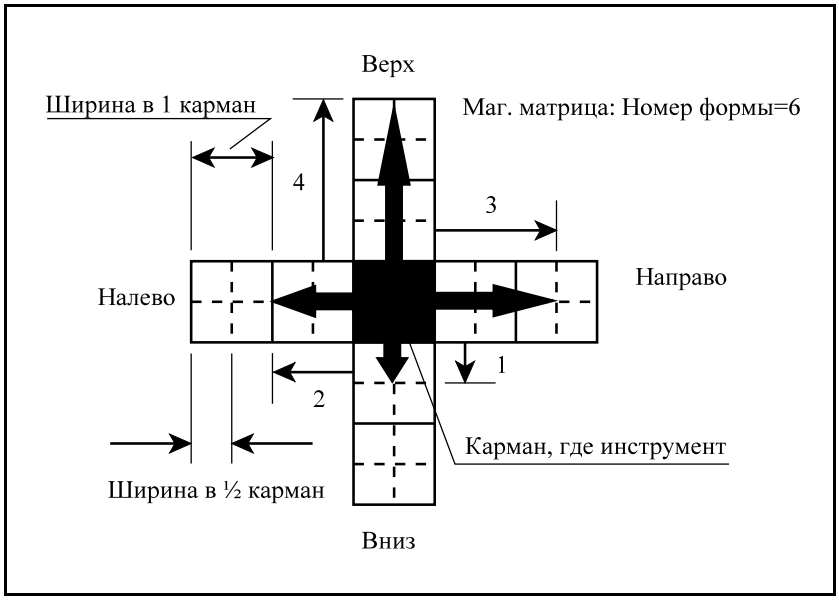
Налево: уменьшающиеся номера карманов (и в цепном магазине и в магазине в матрицы)

Направо: нарастающиеся номера карманов (и в цепном магазине и в магазине в матрицы)

Верх: уменьшающиеся порядковые номера

Вниз: нарастающиеся порядковые номера





В примерах рисунков выше потребность в мест инструментов надо задавать следующим образом:

Номер формы	Налево	Направо	Верх	Вниз	Геометрия
...					
5	4	1	0	0	
6	2	3	4	1	

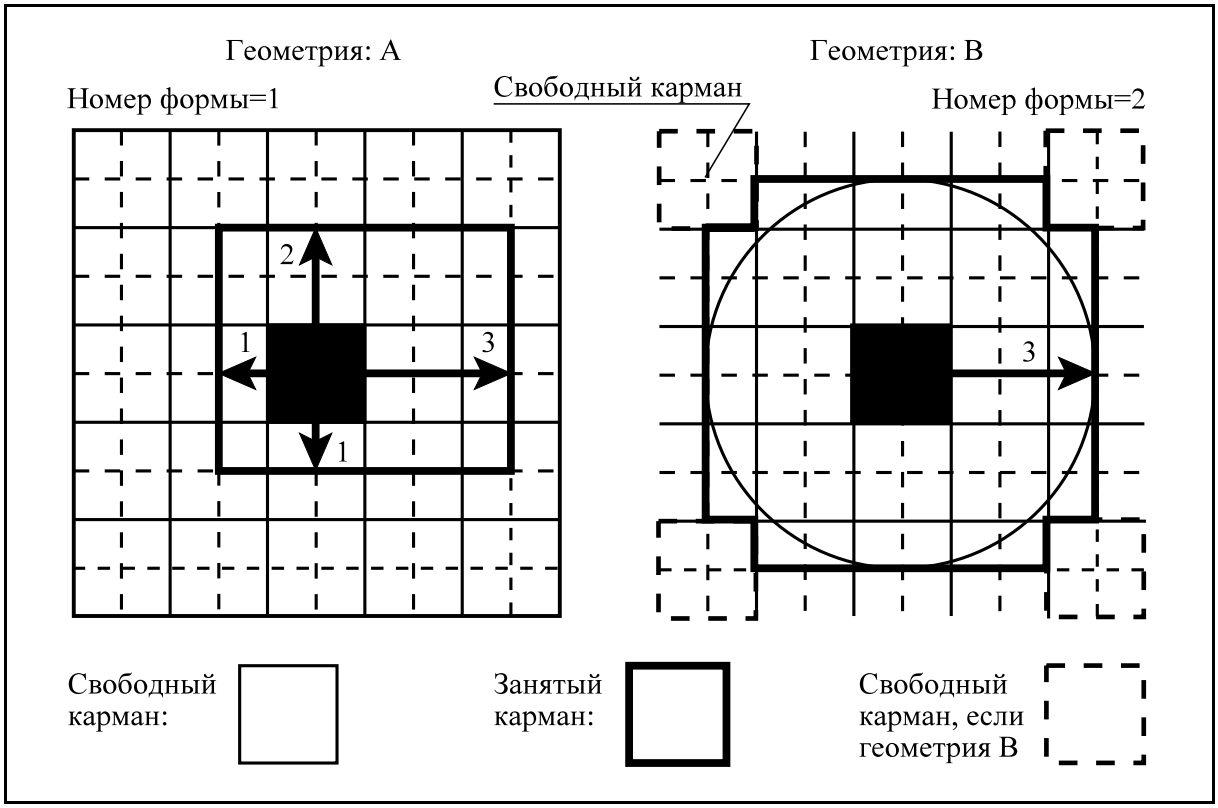
3 Геометрия

В столбце Геометрия можно задавать то, что место, занятое инструментом имеет форму:

A: прямоугольника,

B: круга.

У инструментов круглой формы, если геометрию задавать как B, на углах будет свободный карман, напротив тому, если задавать потребность в мест как прямоугольник.



Как это видно и из рисунка, столбец Геометрия имеет значение только в случае расположения магазинов в матрицы. На основании рисунка выше, заполнение таблицы выполняется следующим образхзом:

Номер формы	**Налево**	**Направо**	**Верх**	**Вниз**	**Геометрия**
1	1	3	2	1	A
2	3	3	3	3	B

6.26.4 Смена данных между двумя различными карманами двух различных магазинов

Команда: MW
Код функции: 100

Поменяет местами Номера данных, имеющихся в двух строчках, определённых Номером магазина и Номером кармана в Таблице места инстументов.
Процесс его использования:

отбор инструмента из магазина 1. и вставка в магазин готовности 1., затем
отбор инструмента из магазина готовности 1. и вставка в шпиндель, одновременно вставка имеющегося в шпинделе инструмента в магазин готовности, затем
вставка инструмента из магазина готовности в магазин 1.

Вслед за каждым движением инструмента такого рода, происходит приведение в порядок Таблицы места инстументов с помощью команды выше.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 4
2	Номер 1-го магазина
3	Номер 1-го кармана
4	Номер 2-го магазина
5	Номер 2-го кармана

*: данные не нужно заполнять

Выводные параметры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных: 4
2	Номер 1-го магазина
3	Номер 1-го кармана
4	Номер 2-го магазина
5	Номер 2-го кармана

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных (не 4)
3	Защита от записи
100	Заданный Номер 1-го магазина не существует
101	Заданный Номер 1-го кармана не существует
102	Заданный Номер 2-го магазина не существует
103	Заданный Номер 2-го кармана не существует

6.26.5 Поиск пустого кармана

Команда: MR
Код функции: 101

Инструмент, возвращающийся в целевой магазин, ищет пустой карман, с учётом кода Формы инструмета. Необходимо задавать, что с какого кармана целевого магазина выполнить поиск. Идентификация инструмента поисходит по Номеру магазина-источника и Номеру кармана. Поиск пустого кармана не выполняется в магазине готовности, а также в магазине шпинделя.

☞ ***Внимание!*** *На выполнение команды влияет бит #0 MRW параметра N1274 PLC PrgConfig!*

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных 7
2	Номер целевого магазина
3	Номер кармана целевого магазина: с какой позиции искать место
4	Направление поиска в таблице: =0: двухстороннее =1: в положительное направление =2: в отрицательное направление
5	Номер магазина-источника
6	Номер кармана магазина-источника
7	Номер найденного кармана *
8	Направление вращения магазина *

*: данные не нужно заполнять

Выводные параметры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных: 7
2	Номер целевого магазина
3	Номер кармана целевого магазина: с какой позиции искать место
4	Направление поиска: =0: двухстороннее =1: в положительное направление =2: в отрицательное направление
5	Номер магазина-источника
6	Номер кармана магазина-источника
7	Номер найденного кармана

Адрес	Данные
8	Направление вращения магазина: =1: в положительное направление =2: в отрицательное направление

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных (не 4)
3	Защита от записи
100	Заданный Номер целевого магазина не существует
101	Заданный Номер целевого кармана не существует
102	Заданный Номер магазина-источника не существует
103	Заданный Номер кармана-источника не существует
104	Нет пустого кармана: Номер найденного кармана=0

6.26.6 Регистрация нового инструмента в Таблице оператора инструментов

Команда: MW
Код функции: 102

Применяется тогда, когда желаем регистрировать новый инструмент через PLC. Команда поищет в Таблице оператора инструментов первую пустую, или недействительную строку (I), то есть Номер данных и в эту запишет данные нового инструмента.
Пример для применения: магазин заполняется через шпиндель. вручную:

В управлении переключаем в режим Загрузки инструмента.

В индивидуальном кадре занесём Номер типа инструмента по адресу T. При этом номер найденного и переданного для PLC магазина и кармана пропустим, на основании Номера типа инструмента и Номера магазина шпиднля, переданного по коду T, зарегистрируем инструмент, затем после регистрации вручную занесём дальнейшие элементы Таблицы оператора инструментов, например: Код формы, Стойкость и т.д.

После этого запустив процесс либо клавишей, либо кодом M, поищем пустое место для инструмента и вставим в магазин.

Вводные параматеры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 3, не более 55
2	Номер магазина
3	Номер кармана магазина: где инструмент находится
4	Номер типа
5	Информация об инструменте
6	Номер формы
7	Статус стойкости
8	Счётчик стойкости
9	Стойкость
10	Предупреждающая стойкость
11	H: Номер ячейки коррекции длины в фрезерном канале
12	D: Номер ячейки коррекции радиуса в фрезерном канале
13	G: Номер ячейки коррекции геометрии в токарном канале
14	W: Номер ячейки коррекции износа в токарном канале
15	S: Число оборотов шпинделя, относящееся к инструменту
16	F: подача, относящаяся к инструменту
17	
18	Пользовательские 1, бинарные
19	Пользовательские 2, с плавающей запятой
20	
21	Пользовательские 3, с плавающей запятой
22	
...	
...	
55	Пользовательские 20, с плавающей запятой
56	

*: данные не нужно заполнять
В длину данных надо записать не менее 3,так как запонение данных Номера магазина, Номера кармана и Номера типа не обязательно. Командой заполняется в Таблице оператора инструментов всего столько столбцев, сколько задавали параметром Длина данных, отнимая 2 (Номер магазина, Номер кармана).

Выводные параматеры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных: не менее 3, не более 55
2	Номер магазина
3	Номер кармана магазина: где инструмент находится
4	Номер типа
5	Информация об инструменте
6	Номер формы
7	Статус стойкости
8	Счётчик стойкости
9	Стойкость
10	Предупреждающая стойкость
11	H: Номер ячейки коррекции длины в в фрезерном канале
12	D: Номер ячейки коррекции радиуса в в фрезерном канале
13	G: Номер ячейки коррекции геометрии в токарном канале
14	W: Номер ячейки коррекции износа в токарном канале
15	S: Число оборотов шпинделя, относящееся к инструменту
16	F: подача, относящаяся к инструменту
17	
18	Пользовательские 1, бинарные
19	Пользовательские 2, с плавающей запятой
20	
21	Пользовательские 3, с плавающей запятой
22	

Адрес	Данные
...	
...	
55	Пользовательские 20, с плавающей запятой
56	

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных: не больше, чем 2, или не меньше, чем 55 или на заданную длину данных нельзя записать значение столбца с целым числом
3	Защита от записи
100	Заданный Номер магазина не существует
101	Заданный Номер кармана не существует
105	Ошибка Номера типа: не изобразимо на 8 десятичных числах
106	Ошибка Информации об инструмента:
107	Ошибка Номера формы:
108	Ошибка Статуса стойкости:
109	Ошибка Счётчика стойкости:
110	Ошибка Стойкости:
111	Ошибка Предупреждающей стойкости:
112	Ошибка задачи H: его значение больше, чем длина Таблицы коррекции
113	Ошибка задачи D: его значение больше, чем длина Таблицы коррекции
114	Ошибка задачи G: его значение больше, чем длина Таблицы коррекции

Код	Сообщение
115	Ошибка задачи W: его значение больше, чем длина Таблицы коррекции
116	S: Число оборотов шпинделя, относящееся к инструменту
117	F: подача, относящаяся к инструменту
118	Ошибка Пользовательские 1
119	Ошибка Пользовательские 2
120	Ошибка Пользовательские 3
...	
137	Ошибка Пользовательские 20
138	Таблица стойкости переполнена

6.26.7 Изменение данных инструмента в Операторе инструментов

Команда: MW
Код функции: 103

Изменение данных инструмента, идентифицированного на основании Номера магазна и Номера кармана в Таблице оператора инструментов.

Вводные параматеры:

Адрес	Данные
0	Нормальное выполнение *
1	Длина данных: не менее 3, не более 55
2	Номер магазина
3	Номер кармана магазина: где инструмент находится
4	Номер типа
5	Информация об инструменте
6	Номер формы
7	Статус стойкости
8	Счётчик стойкости
9	Стойкость

Адрес	Данные
10	Предупреждающая стойкость
11	H: Номер ячейки коррекции длины в в фрезерном канале
12	D: Номер ячейки коррекции радиуса в в фрезерном канале
13	G: Номер ячейки коррекции геометрии в токарном канале
14	W: Номер ячейки коррекции износа в токарном канале
15	S: Число оборотов шпинделя, относящееся к инструменту
16	F:подача, относящаяся к инструменту
17	
18	Пользовательские 1, бинарные
19	Пользовательские 2, с плавающей запятой
20	
21	Пользовательские 3, с плавающей запятой
22	
...	
...	
55	Пользовательские 20, с плавающей запятой
56	

*: данные не нужно заполнять

В длину данных надо записать не менее 3, так как запонение данных Номера магазина, Номера кармана и Номера типа не обязательно. Командой изменяется в Таблице оператора инструментов всего столько столбцев, сколько задавали параметром Длина данных, отнимая 2 (Номер магазина, Номер кармана).

Выводные параматеры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 3, не более 55
2	Номер магазина
3	Номер кармана магазина: где инструмент находится
4	Номер типа

Адрес	Данные
5	Информация об инструменте
6	Номер формы
7	Статус стойкости
8	Счётчик стойкости
9	Стойкость
10	Предупреждающая стойкость
11	H: Номер ячейки коррекции длины в в фрезерном канале
12	D: Номер ячейки коррекции радиуса в в фрезерном канале
13	G: Номер ячейки коррекции геометрии в токарном канале
14	W: Номер ячейки коррекции износа в токарном канале
15	S: Число оборотов шпинделя, относящееся к инструменту
16	F: подача, относящаяся к инструменту
17	
18	Пользовательские 1, бинарные
19	Пользовательские 2, с плавающей запятой
20	
21	Пользовательские 3, с плавающей запятой
22	
...	
...	
55	Пользовательские 20, с плавающей запятой
56	

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных: не больше, чем 2, или не меньше, чем 55 или на заданную длину данных нельзя записать значение столбца с целым числом
3	Защита от записи
100	Заданный Номер магазина не существует
101	Заданный Номер кармана не существует
105	Ошибка Номера типа: не изобразимо на 8 десятичных числах
106	Ошибка Информации об инструмента:
107	Ошибка Номера формы:
108	Ошибка Статуса стойкости:
109	Ошибка Счётчика стойкости:
110	Ошибка Стойкости:
111	Ошибка Предупреждающей стойкости:
112	Ошибка задачи H: его значение больше, чем длина Таблицы коррекции
113	Ошибка задачи D: его значение больше, чем длина Таблицы коррекции
114	Ошибка задачи G: его значение больше, чем длина Таблицы коррекции
115	Ошибка задачи W: его значение больше, чем длина Таблицы коррекции
116	S: Число оборотов шпинделя, относящееся к инструменту
117	F: подача, относящаяся к инструменту
118	Ошибка Пользовательские 1
119	Ошибка Пользовательские 2

Код	Сообщение
120	Ошибка Пользовательские 3
...	
137	Ошибка Пользовательские 20
139	В кармане заданного магазина нет инструмента

6.26.8 Ввод данных инструмента в Оператор инструментов

Команда: MR
Код функции: 104

Ввод данных инструмента, идентифицированного на основании Номера магазина и Номера кармана к числам переменных PLC.

☞ Внимание! На выполнение команды влияет бит #0 MRW параметра N1274 PLC PrgConfig!

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 3, не более 55
2	Номер магазина
3	Номер кармана магазина: где инструмент находится
4	Номер типа*
5	Информация об инструменте*
6	Номер формы*
7	Статус стойкости*
8	Счётчик стойкости*
9	Стойкость*
10	Предупреждающая стойкость*
11	H: Номер ячейки коррекции длины в в фрезерном канале*
12	D: Номер ячейки коррекции радиуса в в фрезерном канале*

Адрес	Данные
13	G: Номер ячейки коррекции геометрии в токарном канале*
14	W: Номер ячейки коррекции износа в токарном канале*
15	S: Число оборотов шпинделя, относящееся к инструменту*
16	F: подача, относящаяся к инструменту*
17	
18	Пользовательские 1, бинарные*
19	Пользовательские 2, с плавающей запятой*
20	
21	Пользовательские 3, с плавающей запятой*
22	
...	
...	
55	Пользовательские 20, с плавающей запятой*
56	

*:данные не нужно заполнять

В длину данных надо записать не менее 3, так как запонение данных Номера магазина, Номера кармана и Номера типа не обязательно. Командой зачитется из Таблицы оператора инструментов всего столько столбцев, сколько задавали параметром Длина данных, отнимая 2 (Номер магазина, Номер кармана).

Выводные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 3, не более 55
2	Номер магазина
3	Номер кармана магазина: где инструмент находится
4	Номер типа
5	Информация об инструменте
6	Номер формы
7	Статус стойкости

Адрес	Данные
8	Счётчик стойкости
9	Стойкость
10	Предупреждающая стойкость
11	H: Номер ячейки коррекции длины в в фрезерном канале
12	D: Номер ячейки коррекции радиуса в в фрезерном канале
13	G: Номер ячейки коррекции геометрии в токарном канале
14	W: Номер ячейки коррекции износа в токарном канале
15	S: Число оборотов шпинделя, относящееся к инструменту
16	F: подача, относящаяся к инструменту
17	
18	Пользовательские 1, бинарные
19	Пользовательские 2, с плавающей запятой
20	
21	Пользовательские 3, с плавающей запятой
22	
...	
...	
55	Пользовательские 20, с плавающей запятой
56	

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных: не больше, чем 2, или не меньше, чем 55 или на заданную длину данных нельзя записать значение столбца с целым числом

Код	Сообщение
100	Заданный Номер магазина не существует
101	Заданный Номер кармана не существует
139	В кармане заданного магазина нет инструмента

6.26.9 Удаление данных инструмента из Оператора инструментов

Команда: MW
Код функции: 105

Выполняет удаление данных инструмента идентифицированного на основании Номера магазина и Номера кармана в Таблице оператора инструментов. Удаляется вся срока.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 2
2	Номер магазина
3	Номер кармана магазина: идентификация инструмента

*: данные не нужно заполнять

Выводные параметры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных: 2
2	Номер магазина
3	Номер кармана магазина: идентификация инструмента

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция

Код	Сообщение
2	Недействительная длина данных (не 2)
3	Зашита от записи
100	Заданный Номер магазина не существует
101	Заданный Номер кармана не существует
139	В кармане заданного магазина нет инструмента

6.26.10 Изменение одних данных инструмента в Операторе инструментов

Команда: MW
Код функции: 106

Изменение данных инструмента, идентифицированного на основании Номера магазна и Номера кармана в Таблице оператора инструментов.

Вводные параметры:

Cím	Adat
0	Код выполнения *
1	Длина данных: не менее 4, не более 5
2	Номер магазина
3	Номер кармана магазина: идентификация инструмента
4	Номер данных: см. Таблицу номера данных
5	Данные 1-ое слово
6	Данные 2-ое слово: заполняется только тогда, если записывае-мые данные с плавающей запятой

*: данные не нужно заполнять
В Длину данных надо записать 3, при записи целых данных, далее 5,если с плавающей запятой.

Истолкование Номеров данных в Таблице оператора инструментов:

Номер данных	Сообщение
1	Номер итпа
2	Информация об инструменте

Номер данных	Сообщение
3	Номер формы
4	Статус Стойкости
5	Счётчик Стойкости
6	Стойкость
7	Предупреждающая стойкость
8	H: Номер ячейки коррекции длины в в фрезерном канале
9	D: Номер ячейки коррекции радиуса в в фрезерном канале
10	G: Номер ячейки коррекции геометрии в токарном канале
11	W: Номер ячейки коррекции износа в токарном канале
12	S: Число оборотов шпинделя, относящееся к инструменту
13	F: подача, относящаяся к инструменту
14	Пользовательские 1, бинарные
15	Пользовательские 2, с плавающей запятой
16	Пользовательские 3, с плавающей запятой
...	
33	Пользовательские 20, с плавающей запятой

Выводные параметры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных
2	Номер магазина
3	Номер кармана магазина: идентификация инструмента
4	Номер данных: см. Таблицу номера данных
5	Данные 1-ое слово
6	Данные 2-ое слово: заполняется только тогда, если записываемые данные с плавающей запятой

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных: не 4, или 5 не соответствует длине данных переменной, на которую имеется ссылка в Номере данных
3	Защита от записи
100	Заданный Номер магазина не существует
101	Заданный Номер кармана не существует
105	Ошибка Номера типа: не изобразимо на 8 десятичных числах
106	Ошибка Информации об инструмента:
107	Ошибка Номера формы:
108	Ошибка Статуса стойкости:
109	Ошибка Счётчика стойкости::
110	Ошибка стойкости::
111	Ошибка Предупреждающей стойкости:
112	Ошибка задачи H: его значение больше, чем длина Таблицы коррекции
113	Ошибка задачи D: его значение больше, чем длина Таблицы коррекции
114	Ошибка задачи G: его значение больше, чем длина Таблицы коррекции
115	Ошибка задачи W: его значение больше, чем длина Таблицы коррекции
116	S: Число оборотов шпинделя, относящееся к инструменту
117	F: подача, относящаяся к инструменту
118	Ошибка Пользовательские 1
119	Ошибка Пользовательские 2
120	Ошибка Пользовательские 3
...	

Код	Сообщение
137	Ошибка Пользовательские 20
139	В кармане заданного магазина нет инструмента
140	Недействительный Номер данных

6.26.11 Ввод одних данных инструмента в Оператор инструментов

Команда: MR
Код функции: 107

Ввод заданных данных данных инструмента, идентифицированного на основании Номера магазна и Номера кармана из Таблицы оператора инструментов.

☞ Внимание! На выполнение команды действует бит #0 MRW параметра N1274 PLC PrgConfig!

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 4, не более 5
2	Номер магазина
3	Номер кармана магазина: идентификация инструмента
4	Номер данных: см. Таблицу номера данных
5	Данные 1-ое слово *
6	Данные 2-ое слово: принимается только тогда, если зачитае-мые данные с плавающей запятой *

*: данные не нужно заполнять
В Длину данных надо записать 3, при записи целых данных, далее 5,если с плавающей запятой.

Истолкование Номеров данных в Таблице оператора инструментов:

Номер данных	Сообщение
1	Номер типа
2	Информация об инструменте

Номер данных	Сообщение
3	Номер формы
4	Статус Стойкости
5	Счётчик Стойкости
6	Стойкость
7	Предупреждающая стойкость
8	H: Номер ячейки коррекции длины в в фрезерном канале
9	D: Номер ячейки коррекции радиуса в в фрезерном канале
10	G: Номер ячейки коррекции геометрии в токарном канале
11	W: Номер ячейки коррекции износа в токарном канале
12	S: Число оборотов шпинделя, относящееся к инструменту
13	F: подача, относящаяся к инструменту
14	Пользовательские 1, бинарные
15	Пользовательские 2, с плавающей запятой
16	Пользовательские 3, с плавающей запятой
...	
33	Пользовательские 20, с плавающей запятой

Выводные параметры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных
2	Номер магазина
3	Номер кармана магазина: идентификация инструмента
4	Номер данных: см. Таблицу номера данных
5	Данные 1-ое слово
6	Данные 2-ое слово: принимается только тогда, если зачитаемые данные с плавающей запятой

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных: не 4, или 5 не соответствует длине данных переменной, на которую имеется ссылка в Номере данных
100	Заданный Номер магазина не существует
101	Заданный Номер кармана не существует
139	В кармане заданного магазина нет инструмента
140	Недействительный Номер данных

6.26.12 Поиск интрумента на основании Пользовательских данных

Команда: MR
Код функции: 108

На основании заданных Пользовательских данных она находит в Таблице оператора инструментов первый инструмент, данные которого показывают совпадение, и возвращает Номер магазина и Номер кармана.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 4, или 5
2	Номер магазина *
3	Номер кармана магазина: идентификация инструмента *
4	Номер данных: Номер Пользовательских данных, см. Таблицу номера данных
5	Данные 1-ое слово
6	Данные 2-ое слово: принимается только тогда, если зачитаемые данные с плавающей запятой

*: данные не нужно заполнять

При поиска на Пользовательские данные 1 надо принимать одно слово, при поиска на дальнейшие Пользовательские данные с плавающей запятой - то 2 слова.
Сравнение выполняется на основании системы приращения, установленной параметром. Значит, если например ISB=1, то есть установлена система приращения 0.001 мм, то считается, что два данных совпадают, если абсолютное значение их разности меньше, чем 0.001.

Истолкование Номеров данных в Таблице оператора инструментов:

Номер данных	**Сообщение**
14	Пользовательские 1, бинарные
15	Пользовательские 2, с плавающей запятой
16	Пользовательские 3, с плавающей запятой
...	
33	Пользовательские 20, с плавающей запятой

Выводные параметры:

Адрес	**Данные**
0	Код выполнения: см. таблицу
1	Длина данных: 4, или 5
2	Номер магазина
3	Номер кармана магазина: идентификация найденного инструмента
4	Номер данных
5	Данные 1-ое слово
6	Данные 2-ое слово

Код выполнения может быть следующий:

Код	**Сообщение**
0	Нормальное выполнение
1	Код Недействительная функция
2	Недействительная длина данных: не 4, или 5

Код	Сообщение
140	Недействительный Номер данных
141	Нет попаданий

6.27 Писание и чтение данных таблицы оператора палитры

Данные обращения палитрой сохранены в NC в таблице оператора палитры. Программа PLC с помощью команд MR и MW может добираться до данных таблицы, и может их читать и писать. Применение таблицы оператора палитры можно разрешить, записав в 1 бит #0 PAL параметра N3300 Pallet Contr.
Таблица оператора палитры является глобальной, то есть общей для всех каналов.
Последние две строки программы деталей будет следующая, предполагая, что для обмена палитры PLC выделена M60:

```
Программа деталей
...
...
M60 (обмен палитры)
M30
```

По коду M60 PLC выполняет команду M30, отыскивает из таблицы следующую палитру и выделит на выполнение программу, привязанную к палитре.

6.27.1 Таблица оператора палитры

В таблице оператора палитры сохранены данные палитр, имеющиеся в магазине палитры, на местах работы и монтажа, а также удалённые с выщеперечисленных мест.
Магазином палитры № 1 называем то место хранения, где размещены палитры, готовы к обработке. Параметром N3301 Pallet Pool Length можно задавать, сколько мест имеется в магазине для хранения.
Магазином палитры № 10 называем рабочую зону инструментального станка, где происходит обработка с резанием. В случае обмена палитры, например, по функции M60, обработанная заготовка попадает назад в магазин № 1, а новая заготовка в магазин № 10, то есть в рабочую зону.
Перед магазином № 1 может быть размещено место для монтажа, которое называем ***магазином № 11***. Если на станке имеется место для монтажа, тогда бит #1 MNT параметра N3300 Pallet Contr. надо установить в 1. С места для монтажа, с магазина № 11, палитру со собранной сырой заготовкой можно отправить в магазин №1, далее палитру, содержавщую уже готовую, обработанную заготовку, можно вынести на место монтажа.
Магазином № 0, или ***виртуальным магазином*** называем те строки таблицы, за которыми нет физического места для хранения, они храняют лишь данные палитр на основании их номера идентификации. Если с места для монтажа снять одну палитру, командой PLC можно перенести данные палитры в магазин № 0. Если на место для монтажа командой PLC попадает новая палитра (например, под действием чтения RFID), данные палитры могут быть вызваны из магазина № 0, т. е. из виртуального магазина, предполагая, что заготовку с таким номером идентификации уже обрабатывали и её данные сохранили в виртуальном магазине № 0.

Столбцы 1-5 таблицы оператора палитры:

Номера магазинов [] и Номера мест	**Номера данных**					
	1	**2**	**3**	**4**	**5**	**..**
	Идентификатор палитры	**Штат**	**Приоритет**	**Число выполненных программ**	**Число выделённых программ**	**..**
Рабочая [10] 1						
Монтажная [11] 1						
[1] 1						
[1] 2						
...						
...						
[1] n						
[0] 1						
[0] 2						
...						

Столбец с номером магазина и с номером места:

Рабочая строка [10] 1: она содержит данные палитры, имеющейся в рабочей зоне. В автоматическом режиме её нельзя редактировать. С точки зрения PLC она является магазином № 10.

Монтажная строка [11] 1: она содержит данные палитры, имеющейся на месте монтажа. Она видна только тогда, если бит #2 MNT параметра N3300 Pallet Contr. имеет значение 1. Редактируется только тогда, если PLC этого не запрещает, например потому, что уже запущен обмен палитр между местом монтажа и магазином. С точки зрения PLC она является магазином № 11.

Строки с Номером места [1] 1, 2, ... n Магазина палитры: Они содержат данные палитр, имеющихся в магазине палитры № 1. Столько строк имеется, сколько их задавали параметром N3301 Pallet Pool Length. Редактируется только тогда, если PLC этого не запрещает, например потому, что уже запущен обмен палитр между рабочей зоной и магазином.

Строки с Номером места [0] 1, 2, ... виртуального магазина: Магазин палитры № 0, виртуальный магазин содержит данные палитр, удалённых из физического магазина. Число строк, доступных в магазине № 0 равно 128, вычитав из него строк магазинов № 1, № 10 и № 11. Редактируется только тогда, если PLC этого не запрещает, например потому, что уже запущен обмен палитр между рабочей зоной и магазином.

Столбец № 1 Идентификации палитры:
Номер данных=1, данные DWORD. Он содержит произвольный номер для идентификации палитры. Оператор может его записать вручную, или может записать и PLC, например, после прочтения одного из кодов RFID. Если оператор палитры находит в магазине № 1 палитру с записанным/прочтённым номером для идентификации, данные соответствующей палитры автоматически переведёт в соответствующую (физическую) строку таблицы.
Команда MW41 PLC на основании номера для идентификации палитры рабочего места назначает программу (программы) к выполнению.

Столбец № 2 Штата:
Номер данных=2, значение DWORD. Он содержит состояние палитры. Оператор может его записать вручную, или может записать и PLC.
Его значение=0: ***Нельзя использовать*** Место палитры нельзя использовать, например, из-за механического повреждения. Нельзя туда разместить палитру.
Его значение=1: ***Пустой***. На данной станции нет ни палитры, ни платформы для палитры. Такой случай бывает тогда, когда палитры хранятся в магазине № 1 на платформе и вместе с палитрой может быть обменена и платформа, например, между магазином и местом для монтажа.
Его значение=2: ***Только платформа***. На данной станции нет палитры, но имеется платформа. Такой случай бывает тогда, если например, загрузить на рабочее место, но там не нужна платформа.
Его значение=3: ***Под монтажём***. Палитра ещё не разрешена к загрузке. Например, когда палитра попадает с места для монтажа в магазин №1, может принимать состояние № 3. Она физически попадает между загружаемые палитры, однако пока не разрешаем к загрузке, до тех пор нельзя её загрузить на рабочее место.
Его значение=4: ***Готова к загрузке***. Подготовленная и удостоверенная палитра. Палитра готова к загрузке для обработки.
Его значение=5: ***Под обработкой***. Палитра находится под обработкой, на ней идёт прогон программы деталей.
Его значение=6: ***Готова***. На палитре имеется обработанная, пригодная заготовка.
Его значение=7: ***Брак***. На палитре имеется обработанная, но бракованная заготовка.
Его значение=8: ***Переделка***. Палитра возвращена в рабочую зону для исправления.

Столбец № 3 Приоритета:
Номер данных=3, данные типа DWORD. Чем меньше значение приоритета, тем раньше приходит очередь обработки заготовки, имеющейся на палитре. Оператор может его записать вручную, или может записать и PLC.
Если приоритет несколько палитр совпадает, тогда загружается палитра, ближайшая к рабочей зоне, то есть команда поиска MR308 задаёт позицию этой палитры.

Столбец № 4 Число выполненных программ:
Номер данных=4, данные типа DWORD. В ходе выполнения команды M60 программа PLC инкрементирует число выполненных на рабочем месте программ, то есть в магазине № 10.
Пока команда MW41 выделения программы вернётся с кодом 0, не выполнены все программы в палитре. В таком случае PLC не ищет новую палитру в магазине, а со стартом запускает следующую программу.
Пока команда MW41 выделения программы вернётся с кодом 47, приходится искать новую палитру.

Столбец № 5 Число выделённых программ:
Номер данных=5, данные типа DWORD. К одной палитре можно выделить несколько главных программ к выполнению. Число выделённых программ находится в этом столбце.
Пока команда MW41 выделения программы вернётся с кодом 0, не выполнены все программы в палитре. Если число выполненных программ=числу выделённых программ, команда вернётся с кодом 47.

Столбцы 6-10 таблицы оператора палитры:

Номера магазинов [] и Номера мест	**Номера данных**					
	..	**6**	**7**	**8**	**9**	**10**
		Идентификатор программ	**Пользовательский 1**	**Пользовательский 2**	**Пользовательский 3**	**Пользовательский 4**
Рабочий [10] 1						
Монтажный [11] 1						
[1] 1						
[1] 2						
...						
...						
[1] n						
[0] 1						
[0] 2						
...						

Столбец № 6 Идентификатора программ:

Номер данных=6

Если значение параметра N3300 Pallet Contr. равно #3 PRI=0:

Выделённые программы отождествляются 4-, или 8-мизначными номерами программ. Программы должны находиться в корне каталоге Programs, с именем файла Ooooo.nct, или Ooooooooo.nct. В этом случае этот столбец может писать или читать программа DWORD переменного типа, принадлежащая к номеру данных и PLC.

Если значение параметра N3300 Pallet Contr. равно #3 PRI=1:

Выполняемые прораммы выбираются вручную по имени их файла и по маршруту добирания, пользуясь таблицой оператора палитры. В этом случае программа PLC не может писать, или читать этот столбец.

Параметром N3302 No. of Custom Columns можно выделить не более 4 пользовательского столбца.

Пользовательский 1, столбец 7:
Номер данных=7, Бинарные данные, с длиной 8 бита.
Ими можно пользоваться тогда, если значение параметра N3302 No. of Custom Columns >0.

Пользовательский 2-4, столбец 8-10:
Номер данных=8-10, данные типа double.
Ими можно пользоваться тогда, если значение параметра N3302 No. of Custom Columns >1.

6.27.2 Перестановка данных между двумя разными местами двух разных магазинов палитры

Команда: MW
Код функции: 300

Перестанавливаются данные, имеющиеся в двух строках, определённых номером магазинов и номером мест таблицы магазинов палитры.

Вводные параметры:

Адрес	Данные
0	Код выполнения*
1	Длина данных: 3
2	Номер 1-го магазина палитры (1, 10, 11)
3	Номер 1-го места
4	Номер 2-го магазина палитры (0, 10, или 11)

*: Данные не нужно заполнять

Если нужно переставить данные между магазином палитры и местом работы
- 2 Номер 1-го магазина палитры =1
- 3 Номер места 1-го магазина = номер соответствующего кармана
- 4 Номер 2-го магазина палитры =10

Если нужно переставить данные между магазином палитры и местом монтажа
- 2 Номер 1-го магазина палитры =1
- 3 Номер места 1-го магазина = номер соответствующего кармана
- 4 Номер 2-го магазина палитры =11

Если нужно переставить данные между местом монтажа и виртуальным магазином
- 2 Номер 1-го магазина палитры =11
- 3 Номер места 1-го магазина = 1
- 4 Номер 2-го магазина палитры =0

Если нужно переставить данные между магазином палитры и виртуальным магазином
- 2 Номер 1-го магазина палитры =1
- 3 Номер места 1-го магазина = номер соответствующего кармана
- 4 Номер 2-го магазина палитры =0

Выводные параметры:

Адрес	Данные
0	Код выполнения: см. таблицу
1	Длина данных: 3
2	Номер 1-го магазина палитры
3	Номер 1-го места
4	Номер 2-го магазина палитры

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительной функции
2	Недействительная длина данных (не 3)
3	Защита от записы
300	Заданный 1-й номер магазина палитры не существует
301	Заданное место палитры не существует
302	Заданный 2-й номер магазина не существует

6.27.3 Изменение данных палитры

Команда: MW
Код функции: 303

Изменение данных палитры, идентифицируемой на основании номера магазина и номера мест в Таблице оператора палитры.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 3, не более 14
2	Номер магазина палитры
3	Номер места магазина палитры: где имеется палитра
4	Идентификатор палитры
5	Штат
6	Приоритет
7	Число выполненных программ
8	Число выделённых программ
9	Пользовательский 1
10	Пользовательский 2 (double)
11	
12	Пользовательский 3 (double)
13	
14	Пользовательский 4 (double)
15	

*: Данные не нужно заполнять
На длину данных надо записать не менее 3, так как заполнение данных номера магазина, и номера мест обязательно и надо записать не менее Идентификатора палитры. Команда перепишет лишь столько столбцов в Таблице оператора палитры, сколько задавали параметром Длины данных, отняв 2 (номер магазина, номер места). Если изменить данные палитры, имеющиеся на месте работы (номера магазина 10), или на месте монтажа (номера магазина 11), номер места будет 1.

Команда ***Не обращается Идентификатором программ***, это достигается лишь индивидуальной записью данных палитры, командой MW306, в том случае, если бит #3 PRI параметра N3300 Pallet Contr. будет 0.

Для изменения Идентификатора палитры система перенесёт все данные из виртуальной таблицы и изменяет все возможные остальные данные, при условии, что находит строку с таким идентификатором.

Выводные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 3, не более 14
2	Номер магазина палитры
3	Номер места магазина палитры: где имеется палитра
4	Идентификатор палитры
5	Штат
6	Приоритет
7	Число Выполненных программ
8	Число Выделённых программ
9	Пользовательский 1
10	Пользовательский 2 (double)
11	
12	Пользовательский 3 (double)
13	
14	Пользовательский 4 (double)
15	

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительной функции

Код	Сообщение
2	Недействительная длина данных: не больше, чем 2, или не меньше, чем 14
3	Защита от записи
300	Заданный номер магазина палитры не существует
301	Заданный номер места не существует
305	Недействительный идентификатор палитры (<1, или >99999999)
306	Недействительный штат палитры (>9)
307	Ошибка приоритета (>128)
308	Число выполненных программ вне предела значения (больше, чем число выделённых программ)
309	Число программ вне предела значения (> 64)
339	Место палитры пустое

6.27.4 Загрузка данных палитры

Команда: MR
Код функции: 304

Загрузка данных палитры, идентифицированных на основании Номера магазина и Номера места из Таблицы оператора палитры.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: не менее 3, не более 14
2	Номер магазина палитры
3	Номер места магазина палитры: где имеется палитра
4	Идентификатор палитры*
5	Штат*
6	Приоритет*
7	Число выполненных программ*
8	Число выделённых программ*
9	Пользовательский 1*
10	Пользовательский 2* (double)
11	
12	Пользовательский 3* (double)
13	
14	Пользовательский 4* (double)
15	

*: данные не надо заполнять

На длину данных надо записать не менее 3, так как заполнение данных номера магазина, и номера мест обязательно и надо записать не менее Идентификатора палитры. Команда загружает лишь столько столбцов из Таблицы оператора палитры, сколько задавали параметром Длины данных, отняв 2 (номер магазина, номер места). Если загрузить данные палитры, имеющиеся на месте работы (номера магазина 10), или на месте монтажа (номера магазина 11), номер места будет 1.

Команда ***Не обращается Идентификатором программ***, это достигается лишь индивидуальной загрузкой данных палитры, командой MW307, в том случае, если бит #3 PRI параметра N3300 Pallet Contr. будет 0.

Выводные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: не менее 3, не более 14
2	Номер магазина палитры
3	Номер места магазина палитры: где имеется палитра
4	Идентификатор палитры
5	Штат
6	Приоритет
7	Число выполненных программ
8	Число выделённых программ
9	Пользовательский 1
10	Пользовательский 2 (double)
11	
12	Пользовательский 3 (double)
13	
14	Пользовательский 3 (double)
15	

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительной функции
2	Недействительная длина данных: не больше, чем 2, не меньше, чем 14
3	Защита от записи

Код	Сообщение
300	Заданный номер магазина палитры не существует
301	Заданный номер места не существует
339	Место палитры пустое

6.27.5 Изменение в палитре одних из данных оператора палитры

Команда: MW
Код функции: 306

Изменение в таблице оператора палитры одних из заданных данных палитры, идентифицированной на основании номера магазина палитры и номера места.

Вводные параметры:

Адрес	Данные
0	Код выполнения*
1	Длина данных: 4, или 5
2	Номер магазина палитры
3	Номер места магазина палитры
4	Номер данных: смотри Таблицу номера данных
5	Данные (если данные double, надо обеспечить 2 места)
6	

*: данные не надо запонять
Если изменить данные палитры, имеющейся на месте работы (номер магазина 10), или на месте монтажа (номер магазина 11), номер места 1 будет.
В результате изменения идентификатора палитры система переставит все данные из таблицы выбираемых палитр и изменяет все возможные остальные данные.
Идентификатор программ (номер данных=6) можно писать только тогда, если бит #3 PRI параметра N3300 Pallet Contr. равно 0.
Писание изменяет предыдущие выделения.
Можно выделить одновременно и несколько программ. В этом случае надо записать сперва данные Числа выделённых программ в то значение, сколько программ желаем выделить. После этого команда MW306 увеличивает параметр длины данных в зависимости от того, какое число программ желаем принять. Числа программ можно загрузить командой MW306.

Истолкование чисел программ в Таблице оператора палитры:

Номер данных	Сообщение
1	Идентификатор палитры
2	Штат
3	Приоритет
4	Число выполненных программ
5	Число выделённых программ
6	Идентификатор программ
7	Пользовательский 1
8	Пользовательский 2
9	Пользовательский 3
10	Пользовательский 4

Выводные параметры:

Адрес	Данные
0	Код выполнения: смотри таблицу
1	Длина данных: 4, или 5
2	Номер магазина палитры
3	Номер места магазина палитры
4	Номер данных: смотри Таблицу номера данных
5	Данные (если данные, надо обеспечивать 2 места)
6	

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных: не 4, или 5
3	Защита от записи

Код	Сообщение
300	Заданный номер магазина палитры не существует
301	Заданный номер места не существует
305	Недействительный идентификатор палитры (<1, или >99999999)
306	Недействительный штат палитры (>10)
307	Ошибка приоритета (>128)
308	Число выполненных программ вне предела значений (больше, чем число выделённых программ)
309	Ошибка чисел программ (>64)
339	Место палитры пустое

6.27.6 Загрузка в палитре одного из данных оператора палитры

Команда: MR
Код функции: 307

Загрузка из таблицы оператора палитры одних из заданных данных палитры, идентифицированной на основании номера магазина палитры и номера места.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 4, или 5
2	Номер магазина палитры
3	Номер места магазина палитры
4	Номер данных: смотри Таблицу номера данных
5	Данные (если данные double, надо обеспечивать 2 места)*
6	

*: данные не надо заполнять

Если загружаем данные палитры, имеющейся на месте работы (номер магазина 10), или на месте монтажа (номер магазина 11), номер места 1 будет.

Идентификатор программ (номер данных=6) можно прочитать только тогда, если бит #3 PRI параметра N3300 Pallet Contr. равно 0.
Сперва надо загрузить данные Чисел выделённых программ. После этого надо увеличить параметр длины данных команды MR307 в зависимости от того, сколько номеров программ надо загрузить. Номера программ можно загрузить командой MR307.

Истолкование номеров данных в Таблице оператора палитры:

Номер данных	**Сообщение**
1	Идентификатор палитры
2	Штат
3	Приоритет
4	Число выполненных программ
5	Число выделённых программ
6	Идентификатор программ
7	Пользовательский 1
8	Пользовательский 2
9	Пользовательский 3
10	Пользовательский 4

Выходные параметры:

Адрес	**Данные**
0	Код выполнения: смотри таблицу
1	Длина данных 4, или 5
2	Номер магазинов палитры
3	Номер мест магазинов палитры
4	Номер данных: смотри Таблицу номеров данных
5	Данные (если данные double, надо обеспечить 2 места)
6	

Код выполнения может быть следующий:

Код	**Сообщение**
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных: не 4, или 5
3	Защита от записи
300	Заданный номер магазина палитры не существует
301	Заданный номер места не существует
339	Место палитры пустое

6.27.7 Поиск палитры на основании значения номера данных

Команда: MR
Код функции: 308

Команда находит место палитры, заданной на вводном параметре с номером данных и значением из магазина палитры № 1 и возвращает найденный номер места.
Если выполняем поиск по месту палитры, Готовой к загрузке по 2-ому номеру данных с 4-ым штатом, и в магазине имеется несколько палитр с таким же штатом, возвращается место палитры с наибольшим паритетом (номер паритета которого является наименьшим). Если в магазине имеется несколько палитр с одинаковым паритетом, возвращается место палитры, лежащей ближе всего к заданной исходной позиции.
Если ведётся поиск по 2-ому номеру данных не по месту палитры с 4-ым штатом, или по другому номеру данных ищем палитру с заданным значением, возвращается место палитры, лежащей ближе всего к заданной исходной позиции.
В зависимости от механического оформления магазина палитры, то есть в качестве вводного параметра, вращаемого только в одно или в оба направления, можно задавать и направление поиска. В зависимости от этого команда возвращает направление вращения магазина.

Вводные параметры:

Адрес	**Данные**
0	Код выполнения *
1	Длина данных: 6
2	Номер данных

Адрес	Данные
3	Искомое значение
4	Номер места магазина палитры: от какой позиции следует искать
5	Направление поиска в таблице: =0: в двух направлениях =1: в положительное направление =2: в отрицательное направление
6	Найденный номер места *
7	Направление вращения магазина *

*: данные не надо заполнять

Выходные параметры:

Адрес	Данные
0	Код выполнения: смотри таблицу
1	Длина данных: 6
2	Номер данных
3	Искомое значение
4	Номер места магазина палитры
5	Направление поиска в таблице
6	Найденный номер места
7	Вращение магазина =1: в положительное направление =2: в отрицательное направление

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных: не 2
12	Ошибка по номеру данных
301	Заданный номер места не существует
339	Место палитры пустое
341	В магазине нет палитры с искомым штатом

6.27.8 Прочтение значений магазина палитры с данным номером данных

Команда: MR
Код функции: 309

С помощью команды можно прочитать значения произвольного столбца магазина из Таблицы оператора палитры. Можно задавать, что с какой позиции магазина (с какой строки таблицы) начиналось прочтение и сколько строк был прочитан. Если прочтение доходит до конца магазина, и не кончилось заданное количество штук, прочтение продолжается с начала таблицы.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: n ≥4
2	Номер данных: смотри Таблицу Номера данных
3	Номер магазина палитры
4	Номер места магазина палитры
5	Значение, имеющееся на заданном номере места*
...	...*
4+n	...*

*: Данные не надо заполнять

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: n ≥4
2	Номер данных: смотри Таблицу номера данных
3	Номер магазина палитры
4	Номер места магазина палитры
5	Значение, имеющееся на заданном номере места
...	...
4+n	...

Пример:

Пусть имеет 1-й магазин палитры 5 мест и данные прочитаем начиная с 4-го номера мест. Прочитаем первый столбец таблицы, то есть коды идентификации палитры. Результат, полученный после прочтения, будет следующий:

0: 0 (операция без ошибки)
1: Длина данных: 10
2: Длина данных 1 (Идентификатор палитры)
3: Номер магазина палитры:1
4: Номер места магазина палитры: 4
5: Прочтённый идентификатор: 12345679 (4-ое место)
6: Прочтённый идентификатор: 98764 (5-ое место)
7: Прочтённый идентификатор: 123 (1-ое место)
8: Прочтённый идентификатор: 456 (2-ое место)
9: Прочтённый идентификатор: 789 (3-ье место)
10: Прочтённый идентификатор: 12345679 (4-ое место)
11: Прочтённый идентификатор: 98764 (5-ое место)

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительная Функция
2	Недействительная длина данных: не n ≥4
12	Ошибка по номеру данных
300	Заданный номер магазина палитры не существует

Код	Сообщение
301	Заданный номер места не существует
339	Место палитры пустое (внутренняя ошибка)

6.27.9 Выделение к автоматическому выполнению программы, доступной с его идентификатором палитры

Команда: MW
Код функции: 41

Операция писания выполняется только тогда, когда в автоматическом режиме нет прогона программы и нет состояния прерывания в данном канале, то есть верные следующие условия:

CN_START=CN_STOP=CN_INTD=0.

Нужно задавать всегда идентификатор палитры, находящейся в рабочей зоне (в магазине № 10).

Если в палитре выделено несколько программ, следующая программа выделяется таблицей на основании Числа Выполненных программ.

Вводные параметры:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 2
2	Идентификатор палитры, находящейся в магазине № 10
3	Номер канала: 1...8

*: данные не надо заполнять

На длину данных надо писать всегда 2.

Выходные параметры:

Адрес	Данные
0	Код выполнения
1	Длина данных: 2
2	Идентификатор палитры
3	Номер канала: 1...8

Код выполнения может быть следующий:

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных: не 2
41	Недействительный номер канала
45	Файл не существует
46	По данному каналу ещё идёт прогон выполнения программы
47	В палитре нет выполняемой программы
342	Место обработки пустое (в нём нет палитры)

Если в палитре выделено несколько программ к выполнению и код выполнения команды MW41 покажет 0, ещё не выполнена каждая программа и в функции перестановки палитр M60 со стартом можно запускать следующую программу.

Если код выполнения команды MW41 покажет 47, каждая программа выполнена в палитре и функция M60 должна искать новую палитру, готовую к загрузке. После этого для новой палитры надо применить команду выделения программы MW41.

6.28 Коммуникация Mailbox между программой PLC и произвольным средством EtherCAT

Mailbox-ы EtherCAT, средства, применяющие известный протокол, не являются продуктом производства NCT, но они доступны из программы PLC с помощью команд MR201, MW202.

Используемые протоколы могут быть следующими:

SoE: Sercos over EtherCAT,

CoE: CANopen over EtherCAT,

VoE: Vendor over EtherCAT (протокол, определённый производителем)

Использование команд коммуникации mailbox может понадобиться например, в следующих случаях:

SoE, CoE удаление ошибки приводов, ток,

SoE, CoE чтение и индикация данных по току приводов, чисел оборотов .

6.28.1 Чтение данных EtherCAT mailbox

Команда: MR
Код функции: 201

Входные параметры:

В случае протокола VoE:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 4 (VoE)
2	Код протокола: 15
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные

*: данные не надо заполнять

Протокол CoE, когда нет CompleteAccess и не важен код ошибки:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 5 (CoE)

Адрес	Данные
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Индекс: 2 Байт (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)

*: данные не надо заполнять

Протокол CoE, когда нет CompleteAccess:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 6 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Индекс: 2 Байт (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возращённый средством *

*: данные не надо заполнять

Протокол SoE:

Адрес	Данные
0	Код выполнения*
1	Длина данных: 7 (SoE)
2	Код протокола: 5
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT

Адрес	Данные
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	IDN: 2 Байт (0-15 бит): опознавательное число элемента
	Drive Index: 1 Байт (16-23 бит)
7	Код ошибки, возращённый средством *
8	ElementFlags **

*: данные не надо заполнять

**** ElementFlags**

Можно запрашивать данные элемента, установленного на битах ElementFlags, выбранного на IDN. Запрашивать можно хоть все данные.

Истолкование данных ElementFlags по битам:

бит	наименование
0	DataState (состояние параметра)
1	Name (имя кодировано по ASCII)
2	Attribute (атрибут)
3	Unit (единица измерения)
4	Min (минимальное значение)
5	Max (максимальное значение)
6	Value (актуальное значение)
7	DefaultValue (базовое значение)

Протокол CoE:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 7 (CoE)
2	Код протокола: 3

Адрес	Данные
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возвращённый средством *
8	CompleteAccess (может быть 0/1) **

*: данные не надо заполнять

**** CompleteAccess:**

– Если CompleteAccess=0, скачает объект, определённый в поле Index и SubIndex.

– Если CompleteAccess=1 и SubIndex=0, или 1, скачает все объекты, заданные в поле Index.

Выходные параметры:

В случае протокола VoE:

Адрес	Данные
0	Код выполнения
1	Длина данных: 4 (VoE)
2	Код протокола: 15
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные

Протокол CoE, когда нет CompleteAccess и не важен код ошибки:

Адрес	Данные
0	Код выполнения
1	Длина данных: 5 (CoE)
2	Код протокола: 3

Адрес	Данные
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)

Протокол CoE, когда нет CompleteAccess:

Адрес	Данные
0	Код выполнения
1	Длина данных: 6 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возращённый средством

Протокол SoE:

Адрес	Данные
0	Код выполнения
1	Длина данных: 7 (SoE)
2	Код протокола: 5
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные

Адрес	Данные
6	IDN: 2 Байта (0-15 бит): опознавательное число элемента
	Drive Index: 1 Байт (16-23 бит)
7	Код ошибки, возвращённый средством
8	ElementFlags

Протокол CoE:

Адрес	Данные
0	Код выполнения
1	Длина данных: 7 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возвращённый средством
8	CompleteAccess (может быть 0/1)

Код выполнение может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Фнкции
2	Недействительная длина данных: не 3
200	Нет средства EtherCAT с заданным индексом
201	Коммуникацию SOE, или COE не поддерживает средство с заданным индексом
202	Заданный адрес памяти PLC неверный, или неверная длина данных mailbox

Код	Значение
203	Чтение вне времени
204	
205	Средство сообщило об ошибке: возвращённый средством код ошибки можно прочитать из 7-го поля кода ошибки, возвращённого средством

6.28.2 Писание данных EtherCAT mailbox

Команда: MW
Код функции: 202

Входные параметры:

В случае протокола VoE:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 4 (VoE)
2	Код протокола: 15
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные

*: данные не надо заполнять

Протокол CoE, когда нет CompleteAccess и не важен код ошибки:

Адрес	Данные
0	Код выполнения *
1	Длина данных: 5 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах

Адрес	Данные
5	Адрес памяти PLC, куда ожидают данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)

*: данные не надо заполнять

Протокол CoE, когда нет CompleteAccess:

Адрес	Данные
0	Код выполнения*
1	Длина данных: 6 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возращённый средством *

*: данные не надо заполнять

Протокол SoE:

Адрес	Данные
0	Код выполнения*
1	Длина данных: 7 (SoE)
2	Код протокола: 5
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	IDN: 2 Байта (0-15 бит): опознавательное число элемента
	Drive Index: 1 (16-23 бит)

Адрес	Данные
7	Код ошибки, возвращённый средством *
8	ElementFlags **

*: данные не надо заполнять

**** ElementFlags**

Можно писать данные элемента, установленного на битах ElementFlags, выбранного на IDN. Записать можно хоть все данные.

Истолкование данных ElementFlags по битам:

бит	наименование
0	DataState (состояние параметра)
1	Name (имя кодировано по ASCII)
2	Attribute (атрибут)
3	Unit (единица измерения)
4	Min (минимальное значение)
5	Max (максимальное значение)
6	Value (актуальное значение)
7	DefaultValue (базовое значение)

Протокол CoE:

Адрес	Данные
0	Код выполнения*
1	Длина данных: 7 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные

Адрес	Данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возвращённый средством *
8	CompleteAccess (может быть 0/1) **

*: данные не надо заполнять

**** CompleteAccess:**

– Если CompleteAccess=0, пишется объект, определённый в поле Index и SubIndex.

– Если CompleteAccess=1 и SubIndex=0, или 1, пишутся все объекты, определённые в поле Index.

Выходные параметры:

В случае протокола VoE:

Адрес	Данные
0	Код выполнения
1	Длина данных: 4 (VoE)
2	Код протокола: 15
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные

Протокол CoE, когда нет CompleteAccess и не важен код ошибки:

Адрес	Данные
0	Код выполнения
1	Длина данных: 5 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные

Адрес	Данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)

Протокол CoE, когда нет CompleteAccess:

Адрес	Данные
0	Код выполнения
1	Длина данных: 6 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидаются данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возвращённый средством

Протокол SoE:

Адрес	Данные
0	Код выполнения
1	Длина данных: 7 (SoE)
2	Код протокола: 5
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	IDN: 2 Байта (0-15 бит): опознавательное число элемента
	Drive Index: 1 Байт (16-23 бит)
7	Код ошибки, возращённый средством
8	ElementFlags

Протокол CoE:

Адрес	Данные
0	Код выполнения
1	Длина данных: 7 (CoE)
2	Код протокола: 3
3	Индекс средства ECAT: порядковый номер средства, видного в окне EtherCAT
4	Длина данных для средства в Байтах
5	Адрес памяти PLC, куда ожидают данные
6	Index: 2 Байта (0-15 бит)
	SubIndex: 1 Байт (16-23 бит)
7	Код ошибки, возвращённый средством
8	CompleteAccess (может быть 0/1)

Код выполнение может быть следующий:

Код	Значение
0	Нормальное выполнение
1	Недействительный код Функции
2	Недействительная длина данных: не 3
200	Нет средства EtherCAT с заданным индексом
201	Коммуникацию SOE, или COE не поддерживает средство с заданным индексом
202	Заданный адрес памяти PLC неверный, или неверная длина данных mailbox
203	
204	Писание вне времени
205	Средство сообщило об ошибке: возвращённый средством код ошибки можно прочитать из 7-го поля кода ошибки, возвращённого средством

6.29 Коды выполнения команд MR, MW

Код	Сообщение
0	Нормальное выполнение
1	Код Недействительной Функции
2	Недействительная длина данных: не 3
4	Ошибочная контрольная сумма магазина
6	Недействительный номер канала
10	Опознавательный знак переменной не попадает среди 0...1023
11	Начальный адрес первой переменной не болшее или равно PLCNVRAM
12	Ошибка числа данных: (опознавательный знак первой выписываемой переменной)+ (число выписываемых переменных)>1024
20	Ссылка на несуществующую макропеременную
21	Макропеременная не глобальная (индекс макропеременной = 0)
22	Макропеременная глобальная (индекс макропеременной > 0)
23	Непригодный код писания/читания: формат макропеременной (DWORD, double) не соответствует коду писания/чтения
24	Макропеременная не читается
25	Макропеременная не пишется
30	Ссылка на несуществующый параметр/недействительный индекс оси
31	Параметр не глобальный (индекс параметра = 0)
32	Параметр глобальный (индекс параметра> 0)
33	Непригодный код писания/читания: формат параметра (бит, DWORD, double) не соответствует коду писания/чтения

Код	Сообщение
40	Несуществующая программа: в накопителе нет заданной программы
41	Недействительный номер канала
45	Файл не существует
46	По данному каналу ещё идёт прогон выполнения программы
47	В палитре нет выполняемой программы
60	Недействительное опознавательное число связи
61	Создание связи безуспешное
62	В ходе приёма данных произошла ошибка
63	Заданная связь не открыта
64	Длина блока данных ошибочная: =0, или >350
65	Не поступили данные
66	Ошибочные входные параметры
70	Недействительный номер окна
71	Нет связи с индикатором (индикатор не готов к приёму)
80	Недействительный адрес привода
81	Недействительный тип данных
82	Недействительное значение переданных данных
90	Внутреняя ошибка, команду нельзя использовать
91	Индекс оси указывает на несуществующую ось, или контур регулирования позиции не открыт (AN_OPNA=0), или приём позиции от измерительного средства хода не запрещён (AP_EFD=0)

Код	Сообщение
100	Заданный Номер 1-го магазина не существует
101	Заданный Номер 1-го кармана не существует
102	Заданный Номер 2-го магазина не существует
103	Заданный Номер 2-го кармана не существует
104	Нет пустого кармана: Номер найденного кармана=0
105	Ошибка Номера типа: не изобразимо на 8 десятичных числах
106	Ошибка Информации об инструмента:
107	Ошибка Номера формы:
108	Ошибка Статуса стойкости:
109	Ошибка Счётчика стойкости:
110	Ошибка Стойкости:
111	Ошибка Предупреждающей стойкости:
112	Ошибка задачи H: его значение больше, чем длина Таблицы коррекции
113	Ошибка задачи D: его значение больше, чем длина Таблицы коррекции
114	Ошибка задачи G: его значение больше, чем длина Таблицы коррекции
115	Ошибка задачи W: его значение больше, чем длина Таблицы коррекции
116	S: Число оборотов шпинделя, относящееся к инструменту
117	F: подача, относящаяся к инструменту
118	Ошибка Пользовательские 1
119	Ошибка Пользовательские 2
120	Ошибка Пользовательские 3
...	
137	Ошибка Пользовательские 20
138	Таблица стойкости переполнена
139	В кармане заданного магазина нет инструмента

Код	Сообщение
140	Недействительный Номер данных
141	Нет попаданий
200	Нет средства EtherCAT с заданным индексом
201	Коммуникацию SOE, или COE не поддерживает средство с заданным индексом
202	Заданный адрес памяти PLC неверный, или неверная длина данных mailbox
203	Чтение вне времени
204	Писание вне времени
205	Средство сообщило об ошибке: возвращённый средством код ошибки можно прочитать из 7-го поля кода ошибки, возвращённого средством
300	Заданный 1-й номер магазина палитры не существует
301	Заданное место палитры не существует
302	Заданный 2-й номер магазина палитры не существует
305	Недействительный идентификатор палитры (<1, или >99999999)
306	Недействительный штат палитры (>9)
307	Ошибка приоритета (>128)
308	Число выполненных программ вне предела значения (больше чем число выделённых программ)
309	Число программ вне предела значения (> 64)
339	Место палитры пустое
341	В магазине нет палитры с искомым штатом
342	Место обработки пустое (нет в нём палитры)

7 Коммуникация между программой PLC и NC

Коммуникация между программой PLC и NC, далее окружающим миром происходит через память, использованную программой PLC. Объём памяти начинается вслед за словом, содержащим Штатные биты (FLAGS) и длится до адреса PLCNVRAM. Этот объём памяти содержит так называемые ***символи NC***.
Через этот объём памяти, через символи NC, ведёт коммуникацию друг с другом программа PLC и

– периферии NC, то есть выводные и вводные блоки аппаратурного обеспечения, такие, как:
  станочные панели оператора,
  маховички,
  выводы и вводы аналоговых интерфейсов и интерфейсов с двумя состояниями,
  щупы,
  приводы EtherCAT,
  приёмный датчик, далее блоки управления приводами аналогового типа, или с CAN бусом,
– некоторые модули программного обеспечения NC, такие, как:
  модули различных услуг, как например функциональные клавиши, параметры PLC, и т.д.,
  глобальные модули,
  модули обращения осями,
  модули обращения шпинделями,
  модули обращения каналами.

Адрес выводных и вводных блоков аппаратурного обеспечения устанавливаются в управлении на панеле установки EtherCAT соответствующего элемента, и эти адреса отвечают соответствующему объёму памяти PLC.

Из указателей коммуникации глобальные, или общие указатели не индексируются с модулями программного обеспечения NC, в противоположность переменными по обращению осями, шпинделями, или каналами, которые индексируются по осям, шпинделям, или каналам. Под этим понимается, что простая ссылка на символ относится всегда к первой оси, к первому шпинделю, или каналу. Переменная остальных осей, шпинделей, или каналов достигается с индексированной ссылкой.

Символ NC может быть значением
  бинарным
  с двойным словом (DWORD), или
  с плавающей запятой (double).

На каждый объём памяти коммуникации надо ссылаться символично. В нашем справочнике задан адрес памяти коммуникации, то есть физический адрес символо в NC, так как это может меняться при различных версиях программного обеспечения.

Из-за изложенных выше причин, в программе PLC нельзя определить символи с постоянным адресом для объёма памяти от адреса FLAGS до адреса PLCNVRAM, а только такие, которые содержат относительную ссылку по сравнению символом NC.

1-й пример:
Символ MB_JOG1 в матрице клавиш jog ссылается на левую верхнюю клавишу. На конкретном станке эта кдавиша перемещает ось X в отрицательном направлении.
Если желаем ссылаться на клавишу символом B_JOG_XN, то символ надо принять в PLC Editor к символу MB_JOG1, как относящийся к базе со ссылкой референции, с нулевым смещением – и ни в коем случае не записать нумерический адрес символа MB_JOG1 для символа B_JOG_XN.

2-й пример:
Мы желаем ссылаться на 17-й бит двойного слова INP000 символом MGZ_RPT (магазин на включателе референтной точки).
Символ MGZ_RPT надо принять к символу INP000, как относящийся к базе со ссылкой референции со смещением 0 и в число битов задавать 17.

7.1 Станочные панели оператора NCT

Система принимает сигнали станочной панели оператора, оборудованной под

MK19: 19 дюймовый

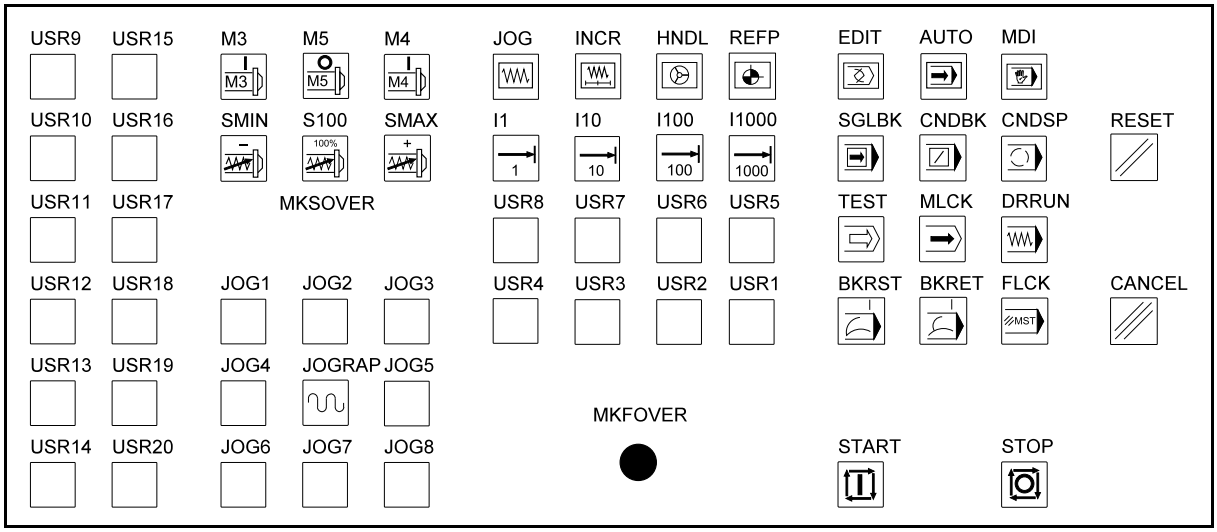
MK15: 15 дюймовый

монитор через сеть EtherCAT.

Распределение клавиш станочной панели оператора видно на рисунке ниже. К каждой клавиши принадлежит и лампа. Над каждой клавишей указан и общий фрагмент текста символа клавиши и лампы. К панели оператора принадлежат вводы клавиш, и выводы ламп свободного назначения. Их можно подключить к произвольному месту, рисунок этих не содержит.

Под клавишами override шпинделя имеется символ MKSOVER, далее над override подачи символ MKFOVER, они являются символами регистра, передающего соответствующее значение override.

☞ ***Внимание!*** *Панель оператора MK15 не содержит клавиши от USR9 до USR20!*



Перечисленные здесь символы клавиш и ламп станочной панели оператора все являются символами ***с бинарной*** ссылкой.

Бинарные переменные панели оператора:

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
MB_START	Клавиша Старта	**ML_START**	Лампа Старта
MB_STOP	Клавиша Стопа	**ML_STOP**	Лампа Стопа
MB_FLCK	Клавиша Функция закрыта	**ML_FLCK**	Лампа Функция закрыта
MB_INP1	Общий ввод 1 станочной панели оператора	**ML_OUT1**	Общий вывод 1 станочной панели оператора

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
MB_M3	Клавиша M3	**ML_M3**	Лампа M3
MB_M4	Клавиша M4	**ML_M4**	Лампа M4
MB_M5	Клавиша M5	**ML_M5**	Лампа M5
MB_INP2	Общий ввод 2 станочной панели оператора	**ML_OUT2**	Общий вывод 2 станочной панели оператора
MB_JOG1	Клавиша Jog1	**ML_JOG1**	Лампа Jog1
MB_JOG2	Клавиша Jog2	**ML_JOG2**	Лампа Jog2
MB_JOG3	Клавиша Jog3	**ML_JOG3**	Лампа Jog3
MB_JOG4	Клавиша Jog4	**ML_JOG4**	Лампа Jog4
MB_JOG5	Клавиша Jog5	**ML_JOG5**	Лампа Jog5
MB_JOG6	Клавиша Jog6	**ML_JOG6**	Лампа Jog6
MB_JOG7	Клавиша Jog7	**ML_JOG7**	Лампа Jog7
MB_JOG8	Клавиша Jog8	**ML_JOG8**	Лампа Jog8
MB_REFP	Клавиша режима Референтной точки	**ML_REFP**	Лампа режима Референтной точки
MB_HNDL	Клавиша режима Маховичка	**ML_HNDL**	Лампа режима Маховичка
MB_INCR	Клавиша режима Сдвиг по шагам	**ML_INCR**	Лампа режима Сдвиг по шагам
MB_JOG	Клавиша режима Перемещения	**ML_JOG**	Лампа режима Перемещения
MB_B20	Не используется	**ML_B20**	Не используется
MB_MDI	Клавиша режима Ручный ввод данных	**ML_MDI**	Лампа режима Ручный ввод данных
MB_AUTO	Клавиша Автоматического режима	**ML_AUTO**	Лампа Автоматического режима
MB_EDIT	Клавиша режима Редактор	**ML_EDIT**	Лампа режима Редактор
MB_TEST	Клавиша Теста	**ML_TEST**	Лампа Теста
MB_MLCK	Клавиша Станок закрыт	**ML_MLCK**	Лампа Станок закрыт
MB_DRRUN	Клавиша Сухой бег	**ML_DRRUN**	Лампа Сухой бег
MB_BKRST	Клавиша Кадр снова	**ML_BKRST**	Лампа Кадр снова
MB_BKRET	Клавиша Кадр назад	**ML_BKRET**	Лампа Кадр назад
MB_CNDSP	Клавиша Условный стоп	**ML_CNDSP**	Лампа Условный стоп
MB_CNDBK	Клавиша Условный кадр	**ML_CNDBK**	Лампа Условный кадр
MB_SGLBK	Клавиша режима По кадрам	**ML_SGLBK**	Лампа режима По кадрам
MB_I1	Клавиша приращений 1	**ML_I1**	Лампа приращений 1
MB_I10	Клавиша приращений 10	**ML_I10**	Лампа приращений 10
MB_I100	Клавиша приращений 100	**ML_I100**	Лампа приращений 100
MB_I1000	Клавиша приращений 1000	**ML_I1000**	Лампа приращений 1000

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
MB_SMAX	Клавиша S+%	**ML_SMAX**	Лампа клавиши S+%
MB_S100	Клавиша S100%	**ML_S100**	Лампа клавиши S100%
MB_SMIN	Клавиша S-%	**ML_SMIN**	Лампа клавиши S-%
MB_JOGRAP	Клавиша быстрого хода Jog	**ML_JOGRAP**	Лампа быстрого хода Jog
MB_USR1	Клавиша 1, определённая в PLC	**ML_USR1**	Лампа 1, определённая в PLC
MB_USR2	Клавиша 2, определённая в PLC	**ML_USR2**	Лампа 2, определённая в PLC
MB_USR3	Клавиша 3, определённая в PLC	**ML_USR3**	Лампа 3, определённая в PLC
MB_USR4	Клавиша 4, определённая в PLC	**ML_USR4**	Лампа 4, определённая в PLC
MB_USR5	Клавиша 5, определённая в PLC	**ML_USR5**	Лампа 5, определённая в PLC
MB_USR6	Клавиша 6, определённая в PLC	**ML_USR6**	Лампа 6, определённая в PLC
MB_USR7	Клавиша 7, определённая в PLC	**ML_USR7**	Лампа 7, определённая в PLC
MB_USR8	Клавиша 8, определённая в PLC	**ML_USR8**	Лампа 8, определённая в PLC
MB_USR9	Клавиша 9, определённая в PLC	**ML_USR9**	Лампа 9, определённая в PLC
MB_USR10	Клавиша 10, определённая в PLC	**ML_USR10**	Лампа 10, определённая в PLC
MB_USR11	Клавиша 11, определённая в PLC	**ML_USR11**	Лампа 11, определённая в PLC
MB_USR12	Клавиша 12, определённая в PLC	**ML_USR12**	Лампа 12, определённая в PLC
MB_USR13	Клавиша 13, определённая в PLC	**ML_USR13**	Лампа 13, определённая в PLC
MB_USR14	Клавиша 14, определённая в PLC	**ML_USR14**	Лампа 14, определённая в PLC
MB_B23	Не используется	**ML_RESET**	Лампа клавиши Reset
MB_B24	Не используется	**ML_CANCEL**	Лампа клавиши Cancel
MB_USR15	Клавиша 15, определённая в PLC	**ML_USR15**	Лампа 15, определённая в PLC
MB_USR16	Клавиша 16, определённая в PLC	**ML_USR16**	Лампа 16, определённая в PLC
MB_USR17	Клавиша 17, определённая в PLC	**ML_USR17**	Лампа 17, определённая в PLC
MB_USR18	Клавиша 18, определённая в PLC	**ML_USR18**	Лампа 18, определённая в PLC
MB_USR19	Клавиша 19, определённая в PLC	**ML_USR19**	Лампа 19, определённая в PLC
MB_USR20	Клавиша 20, определённая в PLC	**ML_USR20**	Лампа 20, определённая в PLC
MB_USR21	Клавиша 21, определённая в	**ML_USR21**	Лампа 21, определённая в PLC

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
	PLC		
MB_USR22	Клавиша 22, определённая в PLC	**ML_USR22**	Лампа 22, определённая в PLC
NB_RESET	Клавиша Reset		
NB_CANCEL	Клавиша Cancel		
NB_NCPC	Не используется		

Панель оператора передаёт через регистр 2 DWORD и положение включателей подачи и override шпиделя.

Переменные с двойным словом к панели оператора:

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
MKBTNS0	Нижние 32 клавиши станочной панели оператора (DWORD)	**MKLEDS0**	Нижние 32 лампы станочной панели оператора (DWORD)
MKBTNS1	Верхние 32 клавиши станочной панели оператора (DWORD)	**MKLEDS1**	Верхние 32 лампы станочной панели оператора (DWORD)
NCTBTNS	Клавиши (DWORD)		
MKFOVER	Положение включателя override подачи: 0, 1, 2...		
MKSOVER	Значение override шпинделя: 0...10 (DWORD)		

Адресовка панелей оператора

К управлению можно подключить не более 4 станочной панели оператора.

Установку адреса станочной панели оператора можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT***. На левой панеле выбрать соответствующий блок и выбирая ушко ***Setting,*** установить ***адрес*** блока.

Значения адресов могут быть:

1, 2, 3, 4.

Адреса 1-й панели оператора достигаются непосредственной ссылкой на символ.

Например:

MB_START

ссылается на клавишу СТАРТа 1-й панели оператора.

Остальные символы панели оператора (2., 3., 4.) достигаются с индексированной ссылкой.

Например:

MB_START,#2

ссылается на клавишу СТАРТа 3-й панели оператора (с индексом 2).

Клавиши, определённые в PLC

Клавишам и лампам, определённым в PLC, может придавать функцию програмист PLC. То же самое относится к клавишам и лампам JOG1...JOG8. Для них програмист PLC может присвоить индивидуальные символы в соответствии тому, что какие оси перемещают данные клавиши..

Эти бинарные символы должны быть заданы всегда к соответствующему символу NC, как к базе (например MB_JOG1), со смещением 0.

Пример:

Символ MB_JOG1 в матрице клавиш jog ссылается на левую верхнюю клавишу. На конкретном станке эта клавиша перемещает ось X в отрицательном направлении.

Если желаем ссылаться на клавишу символом B_JOG_XN, то в PLC Editor надо принять этот символ к символу MB_JOG1, ссылкой референции по сравнению с базой, с нулевым смещением – и ни в коем случае не записать нумерический адрес символа MB_JOG1 для символа B_JOG_XN.

Клавиши, имеющие вывод и ввод в сторону канала

Перечисленные ниже клавиши смены режима, устанавливающие условия работы и клавиши старта, стопа имеют вводы, индексированные по каналам в сторону NC с первой частью CP_ и выводы, индексированные по каналам от NC с первой частью CN_ для ламп:

MB_JOG	– CP_JOG	CN_JOG	– ML_JOG
MB_INCR	– CP_INCR	CN_INCR	– ML_INCR
MB_HNDL	– CP_HNDL	CN_HNDL	– ML_HNDL
MB_REFP	– CP_REFP	CN_REFP	– ML_REFP
MB_EDIT	– CP_EDIT	CN_EDIT	– ML_EDIT
MB_AUTO	– CP_AUTO	CN_AUTO	– ML_AUTO
MB_MDI	– CP_MDI	CN_MDI	– ML_MDI
MB_TEST	– CP_TEST	CN_TEST	– ML_TEST
MB_MLCK	– CP_MLCK	CN_MLCK	– ML_MLCK
MB_DRRUN	– CP_DRRUN	CN_DRRUN	– ML_DRRUN
MB_BKRST	– CP_BKRST	CN_BKRST	– ML_BKRST
MB_BKRET	– CP_BKRET	CN_BKRET	– ML_BKRET
MB_FLCK	– CP_FLCK	CN_FLCK	– ML_FLCK
MB_START	– CP_START	CN_START	– ML_START
MB_STOP	– CP_STOP	CN_STOP	– ML_STOP

Программа PLC должна просмотреть, что нажав клавишу, будет ли вызвано желаемое действие со стороны станка. Например, нажав клавишу MB_START, включён ли станок и закрыта ли рабочая зона.

Если со стороны станка нет препятствий, программа PLC через указатель оператора канала CP запросит желаемое действие от NC. NC также просмотрит, что нажатие клавиши имеет

ли правомочие. Например, в случае CP_START=1 можно ли пускать прогон программы в данном режиме, и т.д.
Если NC принял нажатие клавиши, через соответствующий указатель оператора канала CN даст сигнал об этом для PLC. Например, состоянием CN_START=1.
После этого PLC уже безусловно включает лампу, относящуюся к клавише. Оставаясь при нашем примере ML_START=1.

Клавиши, имеющие только вывод в сторону канала
Перечисленные ниже клавиши условного включателя имеют вводы, индексированные по каналам в сторону NC с первой частью CP_, но не принадлежит к ним сигнал со стороны NC (указатель CN):

MB_JOGRAP	– CP_JOGRAP	– ML_JOGRAP
MB_SGLBK	– CP_SGLBK	– ML_SGLBK
MB_CNDBK	– CP_CNDBK	– ML_CNDBK
MB_CNDSP	– CP_CNDSP	– ML_CNDSP

У этих условных включателей лампой клавиши надо обращаться на основании состояния, сохранённого в программе PLC, или просто скопировать состояние клавиши на лампу.

Клавиши, имеющие вывод в сторону оператора осями
Клавиши jog надо привязать к вводу желаемой перемещать оси с первой частью AP_, индексированной по каналам:

MB_JOGn – AP_JOGP (+направление), или AP_JOGN (- направление) – ML_JOGn

Лампы jog можно подключить непосредственно к клавише.

Обращение клавишами выбора приращения
Величину шага, расчитанную из команд клавиш выбора приращения, надо записать в регистр с плавающей запятой управления, индексированный по каналам, в форме с плавающей запятой.
Например: *0.001, *0.01, *0.1, *1.
На основании указателя CN_INCH необходимо принимать во внимание, что управление используется с метрическим, или дюймовым вводом данных, и если надо, данные конвертируются под систему мер вывода. Система мер вывода: параметр N0104 Unit of Measure #0 IND бит.

MB_I1, MB_I10, MB_I100, MB_I1000 – CP_INC

Обращение override подачи и быстрого хода

Значение override, расчитанное из положения включателя override подачи MKFOVER (DWORD) надо записать в регистры (double) с плавающей запятой NC, индексированные по каналам, в форме с плавающей запятой.

В обоих случаях override *1.0 соответствует 100%-у.

Значение, полученное в регистре MKFOVER, может измениться в зависимости от оформления аппаратурного обеспечения включателя override.

Ввод override быстрого хода также можно подключить к включателю override подачи.

MKFOVER – CP_FOVER, CP_ROVER

Обращение override шпинделя

Обращение override шпинделя выполняется с клавиши MB_SMAX, MB_S100, MB_SMAX. Нажав эти клавиши, или с использованием данных регистра MKSOVER, программа PLC расчитывает значение override с плавающей запятой.

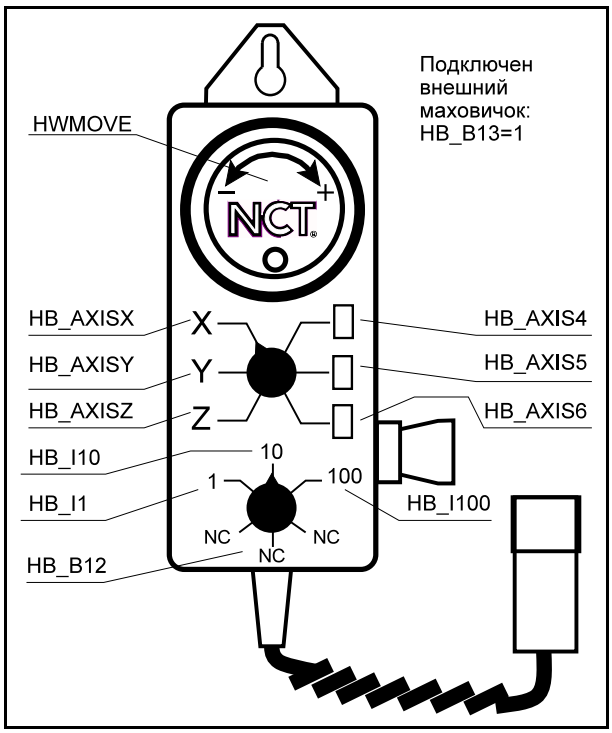
MKSOVER: DWORD, передаёт для PLC значения от 0 до 10.

Для значения override шпинделя стоят на распоряжение регистры с плавающей запятой, индексированные по шпинделям, с первой частью SP. 100%-у соответствует *1.0:

MB_SMAX, MB_S100, MB_SMAX или MKSOVER – SP_SOVER

7.2 Маховички NCT

В случае оборудованного на передную панель маховичка выбор оси и приращения выполняется клавишами панели оператора. Внешним маховичком называется тот блок, который размещён в отдельной коробке, и на коробке имеются поворотные включатели оси и приращения. К положениям поворотных включателей внешнего маховичка принадлежат встроенные бинарные символы.



Бинарные переменные внешнего маховичка следующие:

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
HB_AXISX	Ввод оси X (внешн/маховичок)	**HL_AXISX**	Не используется
HB_AXISY	Ввод оси Y (внешн/маховичок)	**HL_AXISY**	Не используется
HB_AXISZ	Ввод оси Z (внешн/маховичок)	**HL_AXISZ**	Не используется
HB_AXIS4	Ввод оси 4 (внешн/маховичок)	**HL_AXIS4**	Не используется
HB_AXIS5	Ввод оси 5 (внешн/маховичок)	**HL_AXIS5**	Не используется
HB_AXIS6	Ввод оси 6 (внешн/маховичок)	**HL_AXIS6**	Не используется
HB_AXIS7	Ввод оси 7 (внешн/маховичок)	**HL_AXIS7**	Не используется
HB_AXIS8	Ввод оси 8 (внешн/маховичок)	**HL_AXIS8**	Не используется
HB_I1	Ввод приращения 1 (внешн/маховичок)	**HL_I1**	Не используется
HB_I10	Ввод приращения 10 (внешн/маховичок)	**HL_I10**	Не используется
HB_I100	Ввод приращения 100 (внешн/маховичок)	**HL_I100**	Не используется
HB_I1000	Ввод приращения 1000	**HL_I1000**	Не используется

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
	(внешн/маховичок)		
HB_B12	Не используется	**HL_B12**	Не используется
HB_B13	Не используется	**HL_B13**	Не используется
HB_B14	Не используется	**HL_B14**	Не используется
HB_B15	Не используется	**HL_B15**	Не используется

В случае внешнего маховичка, указанного в рисунке выше, подключены ещё следующие символы:

HB_B12: положение **NC** включателя выбора приращения

HB_B13: положение 1 указателя означает, что к управлению **подключен внешний маховичок**.

Переменные с двойным словом маховичка следующие:

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
HWMOVE	Маховичок: перемещение по циклам (только в модуле Int0!) (DWORD)		
HWBITS	Вводы внешнего маховичка (DWORD)	**HWLEDS**	Не используется
		P_H1AS	Номер оси, назначенной к маховичку 1 (=0 неактивный) (DWORD)
		P_H2AS	Номер оси, назначенной к маховичку 2 (=0 неактивный) (DWORD)
		P_H3AS	Номер оси, назначенной к маховичку 3 (=0 неактивный) (DWORD)
		P_H4AS	Номер оси, назначенной к маховичку 4 (=0 неактивный) (DWORD)

HWMOVE: перемещение маховичка по сравнению к предыдущему циклу можно прочитать из регистра. Поскольку обновление регистра происходит с частотой времени цикла TimeSlice, поэтому регистр следует прочитать в модуле Int0 программы PLC. Регистр можно прочитать и в случае встроенного и внешнего маховичка.

P_HnAS: в регистр, назначенный к маховичку 1-4, надо записать номер той оси, которую желаем перемещать данным маховичком. Если значение регистра 0, маховичок является неактивным.

Адресовка маховичка

К управлению можно подключить не более 4 маховичка.

Установку адреса маховичка можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT,*** при установок ***станочной панели оператора***. На левой панеле выбрать соответствующий блок и выбирая ушко ***Setting,*** установить ***адрес*** маховичка.

Значения адреса могут быть:

1, 2, 3, 4.

Адрес встроенного маховичка всегда 4.

Адрес внешнего маховичка в базовом исполнении (заводская установка): 4.

☞ ***Внимание!*** *Поскольку в каждом базовом исполнении используется маховичок 4, не надо забывать об индексировании символов (symbol,#3)!*

7.3 Двухпозиционные вводы/выводы интерфейса на 24 в

К управлению можно подключить следующие блоки ввода и вывода интерфейса 24в, с EtherCAT, с двумя состояниями, производства NCT:

I16: ввод интерфейса 24в с 16 битами,

I16S: ввод интерфейса 24в с 3x16 битами, к каждой точке ввода относится тупиковая точка с напряжением 24в и 0в (для щупов),

I32: ввод интерфейса 24в с 32 битами,

O16: интерфейс 24в с 16 битами с транзистором на выводе,

O8RM: интерфейс 24в с 8 битами с реле на выводе с переключающим контактом

O8R: интерфейс 24в с 8 битами с реле на выводе с замыкающим контактом

Символы NC с *32 битами* вводов и выводов интерфейса 24в с двумя состояниями являются ссылками (DWORD).

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
INP000	Вводы интерфейса 0. DWORD	**OUT000**	Выводы интерфейса 0. DWORD
INP001	Вводы интерфейса 1. DWORD	**OUT010**	Выводы интерфейса 1. DWORD
INP010	Вводы интерфейса 2. DWORD	**OUT020**	Выводы интерфейса 2. DWORD
INP011	Вводы интерфейса 3. DWORD	**OUT030**	Выводы интерфейса 3. DWORD
INP020	Вводы интерфейса 4. DWORD	**OUT040**	Выводы интерфейса 4. DWORD
INP021	Вводы интерфейса 5. DWORD	**OUT050**	Выводы интерфейса 5. DWORD
INP030	Вводы интерфейса 6. DWORD	**OUT060**	Выводы интерфейса 6. DWORD
INP031	Вводы интерфейса 7. DWORD	**OUT070**	Выводы интерфейса 7. DWORD
INP040	Вводы интерфейса 8. DWORD		
INP041	Вводы интерфейса 9. DWORD		
INP050	Вводы интерфейса 10. DWORD		
INP051	Вводы интерфейса 11. DWORD		
INP060	Вводы интерфейса 12. DWORD		
INP061	Вводы интерфейса 13. DWORD		
INP070	Вводы интерфейса 14. DWORD		
INP071	Вводы интерфейса 15. DWORD		

Адресовка плат I/O

Число вводов и выводов, подключенных к управлению *блоков аппаратурного обеспечения* представляет собой многократное целое байта (8 битов).

Установку начального адреса блоков вводов и выводов можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT.*** На левой панели выбрать соответствующий блок и выбирая ушко ***Setting,*** установить ***начальный адрес*** блока.

Адресовка происходит по байтам. Адреса блоков выполняется отдельно для блоков ввода и вывода от 1 до 64. Блоки на 8 битов занимают 1 адрес, а блоки на 16 битов - 2 адреса и т.д.
1-й адрес

в случае блока ввода - это 0-й байт слова INP000

в случае блока вывода - это 0-й байт слова OUT000,

то есть их 00...07 биты. 2-й адрес - это 0. DWORD 1-й байт,то есть их 08...15 биты и так далее.
Порядок адресов не должен последовать за физическим порядком нанизывания в цепь EtherCAT.

В таблице ниже изложен простой образец для заполнения начальных адресов и для места указателей в памяти.

Блок аппаратурного обеспечения	Начальный адрес	Ссылка на сигнал
Блок ввода на 8 битов	1	INP000: 00...07 битов
Блок ввода на 16 битов	2	INP000: 08...23 битов
Блок ввода на 16 битов	4	INP000: 24...31, INP001: 00...07 битов
Блок ввода на 8 битов	6	INP001: 08...15 битов
Блок вывода на 16 битов	1	OUT000: 00...15 битов
Блок вывода на 8 битов	3	OUT000: 16...23 битов
Блок вывода на 16 битов	4	OUT000: 24...31, OUT010: 00...07 битов
Блок вывода на 8 битов	6	OUT010: 08...15 битов

Символы индивидуальных бинарных выводов и вводов может сам оператор PLC опердедлить. Эти бинарные символы следует задавать всегда к соответствующему символу NC, как базовому (например INP000). Выбор бита внутри двойного слова происходит численно (00, ..., 31), согласно распределению I/O.

Пример:
Желаем ссылаться на 17-й бит двойного слова INP000 символом MGZ_RPT (магазин на включателе референтной точки).
Символ MGZ_RPT надо принять к символу INP000, как относящийся к базе с ссылкой референции со смещением 0 и на число битов задавать 17.

7.4 Вводы/выводы плат припасовки щупа NCT

К управлению можно подключить плату, пригодную для приёма сигналов щупа через бус EtherCAT. Их типы:

ETPC: Плата для обращения щупом с 2-мя каналами. На плате помимо сигнала щуп нажат, имеется ещё 3 шт. вводов на 24в и 2 шт. выводов на 24в по щупам.

Управление может обращаться сигналами 8 щупов.

Бинарные вводы и выводы плат припасовки щупа:

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
TN_TS1	Щуп 1 нажат	**TP_OUT11**	Выводной сигнал 1 щупа 1
TN_INP11	Вводной сигнал 1 щупа 1	**TP_OUT12**	Выводной сигнал 2 щупа 1
TN_INP12	Вводной сигнал 2 щупа 1	**TP_OUT21**	Выводной сигнал 1 щупа 2
TN_INP13	Вводной сигнал 3 щупа 1	**TP_OUT22**	Выводной сигнал 2 щупа 2
TN_TS2	Щуп 2 нажат	**TP_OUT31**	Выводной сигнал 1 щупа 3
TN_INP21	Вводной сигнал 1 щупа 2	**TP_OUT32**	Выводной сигнал 2 щупа 3
TN_INP22	Вводной сигнал 2 щупа 2	**TP_OUT41**	Выводной сигнал 1 щупа 4
TN_INP23	Вводной сигнал 3 щупа 2	**TP_OUT42**	Выводной сигнал 2 щупа 4
TN_TS3	Щуп 3 нажат	**TP_OUT51**	Выводной сигнал 1 щупа 5
TN_INP31	Вводной сигнал 1 щупа 3	**TP_OUT52**	Выводной сигнал 2 щупа 5
TN_INP32	Вводной сигнал 2 щупа 3	**TP_OUT61**	Выводной сигнал 1 щупа 6
TN_INP33	Вводной сигнал 3 щупа 3	**TP_OUT62**	Выводной сигнал 2 щупа 6
TN_TS4	Щуп 4 нажат	**TP_OUT71**	Выводной сигнал 1 щупа 7
TN_INP41	Вводной сигнал 1 щупа 4	**TP_OUT72**	Выводной сигнал 2 щупа 7
TN_INP42	Вводной сигнал 2 щупа 4	**TP_OUT81**	Выводной сигнал 1 щупа 8
TN_INP43	Вводной сигнал 3 щупа 4	**TP_OUT82**	Выводной сигнал 2 щупа 8
TN_TS5	Щуп 5 нажат		
TN_INP51	Вводной сигнал 1 щупа 5		
TN_INP52	Вводной сигнал 2 щупа 5		
TN_INP53	Вводной сигнал 3 щупа 5		
TN_TS6	Щуп 6 нажат		
TN_INP61	Вводной сигнал 1 щупа 6		
TN_INP62	Вводной сигнал 2 щупа 6		
TN_INP63	Вводной сигнал 2 щупа 6		
TN_TS7	Щуп 7 нажат		
TN_INP71	Вводной сигнал 1 щупа 7		
TN_INP72	Вводной сигнал 2 щупа 7		

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
TN_INP73	Вводной сигнал 3 щупа 7		
TN_TS8	Щуп 8 нажат		
TN_INP81	Вводной сигнал 1 щупа 8		
TN_INP82	Вводной сигнал 2 щупа 8		
TN_INP83	Вводной сигнал 3 щупа 8		

Адресовка припасовки щупа

Управление может обращаться сигналами не более 8 щупов.

Одна плата припасовки щупа припасует 2 ввода, вывода щупа.

Установку адресов платы припасовки можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT,*** при установок ***припасовки щупа.*** На левой панеле выбрать соответствующий блок и выбирая ушко ***Setting,*** установить ***адреса*** ввода и вывода двух щупов, использованных на плате.

Значения адресов могут быть:

Not used, 1, 2, ..., 8.

Значения адресов соответствуют индексам символов вводов и выводов щупа.

Вводы припасовки щупа

TN_TSn: Щуп n нажат

Этот ввод даст сигнал, что стержень щупа с адресом n отклонился, или клавиша щупа нажата.

Сигнал щуп нажат обязательно активный 0! Плату припасовки следует установить так, чтобы это условие удовлетворилось.

TN_INPn1: Сигнал ввода 1 щупа n

TN_INPn2: Сигнал ввода 2 щупа n

TN_INPn3: Сигнал ввода 3 щупа n

Щуп с адресом n имеет 3 шт опционального ввода на 24в.

Вводы можно использовать например, для показывания состояния готовности щупа к работе, или заряженности батареи.

Выводы припасовки щупа

TP_OUTn1: Вывод 1 щупа n

TP_OUTn2: Вывод 2 щупа n

Щуп с адресом n имеет 2 шт опционального вывода на 24в.

Выводы можно использовать например для включения/выключения щупа.

7.5 Вводы щупа NCT

Тип платы, подключаемой к сети EtherCAT, содержащей ввод для щупа:

SENS: Плата содержит 8 шт. аналоговых ввода термометра KTY84/130 и 1 шт. преобразователь A-D 4-20 мА с 12 битом. Для ввода термометра можно установить значение сравнивания.

Бинарные вводы щупа

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
IN_1EN0	анал. ввод 0. > значение сравнения		
IN_1EN1	анал. ввод 1. > значение сравнения		
IN_1EN2	анал. ввод 2. > значение сравнения		
IN_1EN3	анал. ввод 3. > значение сравнения		
IN_1EN4	анал. ввод 4. > значение сравнения		
IN_1EN5	анал. ввод 5. > значение сравнения		
IN_1EN6	анал. ввод 6. > значение сравнения		
IN_1EN7	анал. ввод 7. > значение сравнения		
IN_1EN8	Не используется		

Вводы щупа с двойным словом

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
ANINPUTS	анал. вводы: 32шт DWORD		
IN_1	штат анал. вводов сравнения (DWORD)		

Адресовка платы SENS и установка значения сравнивания

Управление может обращаться сигналами не более 32 плат SENS.

Установку адресов платы SENS можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT,*** при установок ***платы.*** На левой панеле выбрать соответствующий блок и выбирая ушко ***Setting,*** установить адрес платы.

Значения адресов могут быть:

Not used, 1, 2, ..., 32.

Символы IN_1ENn, IN_1 и ANINPUTS относятся к плате 1. (с индексом 0.), сигналы остальных плат доступны с индексированной ссылкой.

Установку значений сравнения можно выполнить в градусах, по каналам после загрузки окна ***Установки ECAT,*** при установок платы.

Бинарные вводы щупа:

IN_1ENn: n-й аналоговый ввод > значение сравнения

Если на n-ом вводе платы величина аналогового сигнала превосходит установленного значения сравнения, сигнал переходит в 1.

Вводы щупа с двойным словом:

ANINPUTS: Аналоговые вводы: 32 шт. DWORD

На переменной ANINPUTS, при соответствующем индексировании, можно вычитать значение аналогового ввода платы.

IN_1: Штат вводов аналогового сравнения (DWORD)

С переменной доступны вводы сравнения в форме двойного слова..

7.6 Аналоговые вводы NCT

Тип платы, содержащей аналоговые вводы, подключаемой к сети EtherCAT:

DANI: Плата содержит 6 шт. аналоговых цифровых преобразователей с 12 битом. Аналоговые вводы можно конфигурировать на +/-10 в, или 0-20 мА.

Вводы аналоговых сигналов с двойным словом

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
ANINPUTS	Аналоговые вводы: 32 шт DWORD		

Адресовка платы DANI

Управление может обращаться сигналами не более 32 аналоговых вводов.

Установку адресов платы DANI можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT,*** при установок ***платы.*** На левой панеле выбрать соответствующий блок и выбирая ушко ***Setting,*** установить по одному адреса аналоговых вводов платы.

Значения адресов могут быть:

Not used, 1, 2, ..., 32.

Символ ANINPUTS относится к аналоговому вводу 1. (с индексом 0.), сигналы остальных вводов доступны с индексированной ссылкой.

Вводы аналоговых сигналов с двойным словом

ANINPUTS: Аналоговые вводы: 32 шт. DWORD

На переменной ANINPUTS, при соответствующем индексировании, можно вычитать значение соответствующего аналогового ввода платы.

7.7 Вводы/выводы приводов NCT с EtherCAT

К управлению можно подключитьчерез бус EtherCAT следующие типы приводов NCT:

DS-i/I EE: синхронные серводвигатели,

DA-i/I EE: асинхронные двигатели,

где: i: номинальный ток

I: максимальный ток

EE: интерфейс EtherCAT, EnDat 2.2.

Система может обращаться сигналами не более 48 шт. приводов EtherCAT (32 осей + 16 шпинделей).

Приводы EtherCAT передают для NC через сеть EtherCAT позицию, измеренную с оборудованого на двигатель датчика, ток двигателя, число оборотов и т.д., их штат, далее их коды состояния ошибки. Таким же образом принимают от NC основной сигнал скорости, далее прочие команды, поступающие от NC.

Сигналы указанных выше типов приводов NCT EtherCAT следующие:

Сигналы бинарного щтата и контроля приводов

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
DN_ENA	Привод разрешён	**DP_ENA1**	Разрешение привода 1 бит
DN_RDY	Привод готов к работе	**DP_ENA2**	Разрешение привода 2 бита
DN_INC	Датчик приращения на приводе	**DP_EMG**	Нет аварийного торможения
DN_PRM1	Активная таблица параметра 1 бит	**DP_POSLCK**	Одержание привода в позиции вкл
DN_PRM2	Активная таблица параметра 2 бита	**DP_PRM1**	Выбор таблицы параметра привода 1 бит
		DP_PRM2	Выбор таблицы параметра привода 2 бита
		DP_MOD1	Выбор режима регулирования привода 1 бит
		DP_MOD2	Выбор режима регулирования привода 2 бита
		DP_SILCK	Блокировка интегратора регулятора скорости
		DP_ERRCLR	Удаление ошибки привода

Бинарные сигналы об ошибке приводов

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
DN_ERROR	Ошибка привода (DN_ERR>0)		
DN_EDERR1	Ошибка датчика		
DN_EDERR2	Температура датчика высокая		
DN_IEERR1	Не используется		
DN_IEERR2	Не используется		
DN_BVERR	Ошибка сверхнапряжения на шине		
DN_CURERR	Слышком большой ток двигателя		
DN_CMEERR	Ошибка измерения тока		
DN_HALERR	Ошибка коммутирующего сигнала Холла		
DN_HASERR	Ошибка сигнала последовательности Холла		
DN_SRTERR	Ошибка превышения времени		
DN_CWDER1	Ошибка CAN WatchDog		
DN_CWDER2	Не используется		
DN_CHERR1	Прочая ошибка CAN		
DN_CHERR2	Не используется		
DN_ECTERR	Ошибка превышения времени EtherCAT		
DN_PDPINT	Ошибка IGBT		
DN_PRMERR	Ошибка таблицы параметра		
DN_PRGERR	Ошибка программы привода		
DN_FOLERR	Ошибка слежения скорости		
DN_OVHERR	Теплозащита		

Переменные с двойным словом для приводов

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
DN_STAT	Регистр штата привода (DWORD)	**DP_CTRL**	Контрольный регистр привода (DWORD)

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
DN_ERR	Регистр сообщений об ошибке привода (DWORD)		

Адресовка приводов EtherCAT

Управление может обращаться сигналами не более 48 шт. приводов EtherCAT.

Установку адресов приводов можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT, выбирая соответствующий привод.*** На левой панеле выбрать соответствующий блок и выбирая ушко ***Setting,*** установить ***адрес*** привода.

Значения адресов могут быть:

Not used, 1, 2, ..., 48.

Назначение друг к другу адресов приводов и осей

Управление может обращаться не более 32 осями. Адреса в управлении могут быть:

1, 2, ..., 32.

Выдача основного сигнала:

Параметром N0502 Axis Output Type надо выбирать опцию EtherCAT-32.

На параметр N0503 Axis Output Address надо записать адрес того привода, которым желаем перемещать данную ось.

Пример:

Параметр

N0100 Axis Name1 A4=C

определяет адрес 4-й оси как C.

Адрес привода оси C среди установок EtherCAT: 6.

При этом N0503 Axis Output Address A4=6,

то есть 4-ая ось управления выдаёт основной сигнал приводу 6.

Приём сигналов датчика:

Параметром N0500 Axis Input Type надо выбирать опцию EtherCAT.

На параметр N0501 Axis Input Address надо записать адрес того привода, от которого желаем получить сигналы датчика.

☞ ***Внимание!*** *Параметр N0503 Axis Output Address и параметрN0501 Axis Input Address могут отличаться друг от друга!*

Пример:

Параметр

N0100 Axis Name1 A4=C

определяет адрес 4-й оси как C

Адрес привода оси C среди установок EtherCAT: 6.

Сигналы датчика желаем брать с датчика, которым привод обращается.

При этом N0501 Axis Input Address A4=6,

то есть 4-ая ось управления берёт сигналы датчика с привода 6.

Назначение друг к другу адресов приводов и шпинделей

Управление может обращаться не более 16 шпинделями. Адреса шпинделей в управлении могут быть:

1, 2, ..., 16.

Выдача основного сигнала:

Параметром N0602 Spindle Output Type надо выбирать опцию EtherCAT.

На параметр N0603 Spindle Output Address надо записать адрес того привода, с которым желаем перемещать шпиндель.

Пример:

Параметр

N0605 Spindle Name2 S3=3

определяет адрес 3-го шпинделя как S3.

Адрес привода шпинделя S3 среди установок EtherCAT: 8.

При этом N0603 Spindle Output Address S3=8,

то есть 3-й шпиндель управления выдаёт основной сигнал приводу 8.

Приём сигналов датчика:

Параметром N0600 Spindle Input Type надо выбирать опцию EtherCAT.

На параметр N0601 Spindle Input Address надо записать адрес того привода, от которого желаем получить сигналы датчика.

☞ ***Внимание!*** *Параметр N0603 Spindle Output Address и параметр N0601 Spindle Input Address могут отличаться друг от друга!*

Пример:

Параметр

N0605 Spindle Name2 S3=3

определяет адрес 3-го шпинделя как S3.

Адрес привода шпинделя S3 среди установок EtherCAT: 8.

Сигналы датчика желаем брать с датчика, которым привод обращается.

При этом N0601 Spindle Input Address S3=8,

то есть 3-й шпиндель управления берёт сигналы датчика с привода 8.

Ссылка в программе PLC на оси, шпиндели и приводы

Символы NC

AN_ xx и AP_xx

относятся всегда к осям. На эти символы надо ссылаться всегда на основании индекса оси:

#0: 1-я ось, ..., #31: 32-я ось..

Символы NC

SN_ xx и SP_xx

относятся всегда к шпинделям. На эти символы надо ссылаться всегда на основании индекса шпинделя:

#0: 1-й щпиндель, ..., #15: 16-й шпиндель.

Символы NC

DN_ xx и DP_xx

относятся всегда к приводам. На эти символы надо ссылаться всегда на основании индекса привода:

#0: 1-й привод, ..., #47: 48 -й привод.

Индекс оси, или шпинделя данной оси, или шпинделя может отличаться от индекса привода данной оси, или шпинделя!

Штатные сигналы приводов (вводы)

DN_ENA: Привод разрешён

Указатель подтверждения сигналов DP_ENA1 и DP_ENA2. Если значение указателя 1, двигатель работает от привода, он находится под напряжением.

DN_RDY: Привод готов к работе

Если значение сигнала 1, привод готов к работе, можно его разрешать.

DN_INC: Датчик приращения на приводе.

Если указатель DN_INC=0, на двигателе имеется абсолютный датчик EnDAT, если указатель DN_INC=1, на двигателе имеется датчик приращения.

DN_PRM1: Активная таблица параметра 1 бит

DN_PRM2: Активная таблица параметра 2 бита

Указанные выше два бита подскажет, что из какой таблицы параметра должен работать привод. Смотри ещё: описание битов DP_PRM1 и DP_PRM2.

DN_PRM1	DN_PRM2	Активная таблица параметров
0	0	1-я таблица параметров активная
0	1	2-я таблица параметров активная
1	0	3-я таблица параметров активная
1	1	4-я таблица параметров активная

Контрольные сигналы приводов (выводы)

DP_ENA1: Разрешение привода 1 бит

DP_ENA2: Разрешение привода 2 бита

Разрешение привода происходит после смены ниже битов

DP_ENA1 1 –> 0, и

DP_ENA2 0 –> 1.

DP_ENA1	DP_ENA2	Состояние разрешения
0	0	Не разрешено
0	1	Не разрешено
1	0	Разрешено
1	1	Не разрешено

DP_EMG: Нет аварийного торможения

Если DP_EMG=0, привод, не зависимо от полученного на вводе основного сигнала, останавливает двигатель. Останов длится до тех пор, пока привод разрешён.

DP_EMG=1 нормальная работа, согласно полученному основному сигналу.

DP_POSLCK: Одержание привода в позиции вкл.

DP_POSLOCK=0, нормальный режим работы,

DP_POSLOCK=1, при смене сигнала 0 –> 1 привод одерживает двигатель в позиции, измеренной с датчика, не зависимо от полученного основного сигнала.

DP_PRM1: Выбор таблицы параметра привода 1 бит

DP_PRM2: Выбор таблицы параметра привода 2 бита

Указанные выше два бита подскажут, что из какой таблицы параметра должен работать привод. После смены параметра надо дождаться, чтобы привод на битах DN_PRM1 и DN_PRM2 вернул установленный образец бита.

DP_PRM1	DP_PRM2	Выбор таблицы параметра
0	0	Выбор 1-й таблицы параметра
0	1	Выбор 2-й таблицы параметра
1	0	Выбор 3-й таблицы параметра
1	1	Выбор 4-й таблицы параметра

DP_MOD1: Выбор режима регулирования привода 1 бит

DP_MOD2: Выбор режима регулирования привода 2 бита

Установкой указателей можно выбирать из приведенных ниже режимов регулирования:

DP_MOD1	DP_MOD2	Выбор режима регулирования
0	0	Режим регулирования скорости
0	1	Режим регулирования тока (момента)
1	0	Режим регулирования позиции
1	1	Режим регулирования скорости увеличенной точности

DP_SILCK: Блокировка интегратора регулятора скорости

DP_SILCK=0, нормальный режим работы,

DP_SILCK=1: выключение интегратора регулятора скорости.

Пример:

На токарном станка с контршпинделем, работающем с прутком, надо перенять заготовку в контршпиндель. Контршпиндель надо синхронизировать со шпинделем, затем на контршпинделе зажимать патрон, чтобы заготовка одержалась после отрезания в контршпинделе. На то время, пока заготовка не отрезана, и одержена обоими шпинделями, необходимо отключить интегратор регулятора скорости на приводе контршпинделя, иначе вследствие сверхопределимости системы может возникнуть колебание.

DP_ERRCLR: Удаление ошибки привода

Если на приводе имеется ошибка, то есть DN_ERR>0 (регистр ошибки не 0) ошибку надо удалить.

Состояние DP_ERRCLR=1 удаляет ошибку. Сигнал надо одержать в 1 до тех пор, пока регистр ошибки DN_ERR не будет 0, или не наступит состояние DN_ERROR=0.

Бинарные сигналы об ошибке приводов (вводы)

Программа PLC не должна выводить различные сообщение об ошибке привода, это выполняет NC. Надо выдавать только команду для удаления ошибки путём установки указателя DP_ERRCLR=1.

DN_ERROR: Имеется ошибка привода (DN_ERR>0)

DN_ERROR=1, если один из дальнейших битов регистра ошибок выводит сообщение об ошибке.

DN_EDERR1: Ошибка датчика

DN_EDERR1=1, если привод получает сообщение об ошибке от оборудованного на двигатель датчика..

DN_EDERR2: Температура датчика высокая

DN_EDERR2=1, если на двигателе оборудован датчик EnDat, и встроенным в него термодатчиком измеряется температура, выше установленного значения.

DN_BVERR: Ошибка сверхнапряжения на шине

DN_BVERR=1, если напряжение шины превышает установленного значения.

DN_CURERR: Ток двигателя слышком большой

DN_CURERR=1, если ток двигателя превышает установленного на приводе значения $I_{макс}$.

DN_CMEERR: Ошибка измерения тока

DN_CMEERR=1, если наступила ошибка измерения тока.

DN_HALERR: Ошибка коммутирующего сигнала Холла

DN_HALERR=1, если наступила ошибка коммутирующего сигнала.

DN_HASERR: Ошибка сигнала последовательности Холла

DN_HASERR=1, если последовательность коммутирующего сигнала не по коду Грея.

DN_SRTERR: Ошибка превышения времени

DN_SRTERR=1, если в приводе срабатывал центральный таймер WatchDog. Привод неисправный.

DN_CWDER1: Ошибка CAN WatchDog

DN_CWDER1=1, если в приводе срабатывал таймер WatchDog коммуникации CAN. Коммуникация CAN неисправная.

DN_CHERR1: Прочая ошибка CAN

DN_CHERR1=1, если в коммуникации CAN возникла ошибка.

DN_ECTERR: Ошибка превышения времени EtherCAT

DN_ECTERR=1, если в приводе срабатывал таймер WatchDog коммуникации EtherCAT. Коммуникация EtherCAT неисправная.

DN_PDPINT: Ошибка IGBT

DN_PDPINT=1, если возникла ошибка при питании двигателя. Это может быть ошибкой контура, ошибкой монтажа кабеля.

DN_PRMERR: Ошибка таблицы параметра

DN_PRMERR=1, если повреждена активная таблица параметра.

DN_PRGERR: Ошибка программы привода

DN_PRGERR=1, если повреждена рабочая программа привода, её контрольная сумма неверная.

DN_FOLERR: Ошибка слежения скорости

DN_FOLERR=1, если привод не может следить за основным сигналом скорости за установленный период времени.

DN_OVHERR: Теплозащита

DN_OVHERR=1, если срабатывала теплозащита двигателя. Это может быть PTC, или температурный модель.

Вводные регистры приводов с двойным словом

DN_STAT: Штатный регистр привода (DWORD)

Этим символом могут быть доступны штатные биты привода с двойным словом.

DN_ERR: Регистр указателей ошибок привода (DWORD)

Этим символом могут быть доступны биты ошибки привода с двойным словом.

Выводные регистры приводов с двойным словом

DP_CTRL: Котрольный регистр привода (DWORD)

Этим символом могут быть доступны контрольные биты привода с двойным словом.

7.8 Вводы приёма датчика и выводы припасовки аналоговые/шагового двигателя /CAN

Приёмные вводы датчика и выводы припасовки аналоговые/шагового двигателя/CAN можно припасовать к бусу EtherCAT.

Блоки припасовки ввода и вывода используются в следующих случаях:

– Приводы EtherCAT передают позицию, измеренную оборудованным на двигателе датчиком для NC. В тех случаях, когда вместо оборудованного на двигателе датчика приходится пользоваться другим, сигнал надо принимать из соответствующей электроники припасовки. Такими случаями являются например оборудованные на токарный шпиндель датчики инструмента для нарезания резьбы, или если на оси используется измерительная линейка для измерения позиции.

– Другой случай, когда приходится припасовать к управлению приводы с вводом не EtherCAT, а например с аналоговым вводом, или с бусом CAN, или приводы с шаговым двигателем.

На нашем распоряжении имеются следующие блоки припасовки:

– **ENDAT**:

Его вводы:

Пригодные для приёма сигналов 2 шт. датчиков EnDat 2.2, имеющих протокол коммуникации (измерительная линейка, угломер).

– **TTLAI**:

Его вводы:

Блок может принимать на его вводе сигналы 2 шт. датчиков TTL с приращением.

Его выводы:

В случае версии программного обеспечения ECAT-TTLASM на 2 шт. выводах можно выдавать

аналоговый основной сигнал, или

ряд импульсов Pulse-Dir и бит направления для шагового двигателя, или

ряд импульсов CW-CCW двух направлений для шаговых двигателей.

Все три вывода выдаётся по каналам, их подключением выбирается, что каким будем пользоваться.

В случае версии программного обеспечения ECAT-TACHO на 2 шт. выводах можно выдавать

сигнал распоряжения, образованный из разности аналогового основного сигнала и тахосигнала для приводов с аналоговым вводом.

– **TTLCAN**:

Его вводы i:

Блок может принимать на его вводе сигналы 2 шт. датчиков TTL с приращением.

Его выводы:

пригодны для обращения приводом NCT с 2 шт. вводами с бусом CAN.

Регистры коммуникации и указатели блоков припасовки поддерживают связь с PLC через указатели привода DN_, DP_. Система может обращаться сигналами не более 48 шт. приводов EtherCAT и блоков припасовки (32 осей + 16 шпинделей).
Ниже приводим только те сигналы, которыми могут обращаться блоки припасовки.

Бинарные штатные сигналы и контрольные сигналы блоков припасовки

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
DN_INC	На приводе датчик с приращением	**DP_ERRCLR**	Удаление ошибки привода

Бинарные указатели ошибки блоков припасовки

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
DN_ERROR	Имеется ошибка привода (DN_ERR>0)		
DN_EDERR1	Ошибка датчика		
DN_ECTERR	Ошибка превышения времени EtherCAT		

Переменные юлоков припасовки с двойным словом

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
DN_STAT	Штатный регистр привода (DWORD)	**DP_CTRL**	Котрольный регистр привода (DWORD)
DN_ERR	Регистр указателей ошибки привода (DWORD)		

Адресовка блоков припасовки EtherCAT
Управление может обращаться сигналами не более 48 шт. привода и блоков припасовки EtherCAT.
Установку адресов блоков припасовки можно выполнить в ***меню Сервис*** управления, после загрузки окна ***Установки ECAT, выбирая соответствующий блок припасовки.*** На левой панеле выбрать соответствующий блок и выбирая ушко ***Setting,*** установить ***адрес*** блока припасовки.

Значения адресов могут быть:

Not used, 1, 2, ..., 48.

Назначение друг к другу адресов блоков припасовки и осей

Управление может обращаться не более 32 осями. Адреса осей в управлении могут быть:

1, 2, ..., 32.

Выдача основного сигнала:

Параметром N0502 Axis Output Type надо выбирать опцию EtherCAT-32.

На параметр N0503 Axis Output Address надо записать адрес того блока припасовки, которым желаем перемещать данную ось.

Пример:

Параметр

N0100 Axis Name1 A4=C

определяет адрес оси 4 как C.

Ось C перемещает привод с аналоговым вводом. Адрес блока припасовки TTLAI среди установок EtherCAT: 6.

При этом N0503 Axis Output Address A4=6,

то есть 4-я ось управления выдаёт основной сигнал блоку припасовки 6.

Приём сигналов датчика:

Параметром N0500 Axis Input Type надо выбирать опцию EtherCAT.

На параметр N0501 Axis Input Address надо записать адрес того блока припасовки, с которого желаем получить сигналы датчика.

☞ ***Внимание!*** *Параметр N0503 Axis Output Address и параметр N0501 Axis Input Address могут быть различными!*

Пример:

Параметр

N0100 Axis Name1 A2=Y

определяет адрес оси 2 как Y.

Пусть будет адрес привода EtherCAT оси Y среди установок EtherCAT 2. (N0503 Axis Output Address A2=2)

На оси Y оборудована измерительная линейка EnDat. Сигналами измерительной линейки Y обращается блок припасовки ENDAT с адресом 12.

При этом N0501 Axis Input Address A2=12,

то есть 2-я ось управления берёт сигнал датчика с блока припасовки 12.

Назначение друг к другу адресов блоков припасовки и шпинделей

Управление может обращаться не более 16 шпинделями. Адреса шпинделей в управлении могут быть:

1, 2, ..., 16.

Выдача основного сигнала:

Параметром N0602 Spindle Output Type надо выбирать опцию EtherCAT.

На параметр N0603 Spindle Output Address надо записать адрес того блока припасовки, с которым желаем перемещать данный шпиндель.
Пример:
Параметр
N0605 Spindle Name2 S1=1
определяет адрес шпинделя 1 как S1.
Привод шпинделя S1 имеет ввод с бусом CAN, привод получает основной сигнал от блока припасовки TTLCAN, адрес которого среди установок EtherCAT: 8.
При этом N0603 Spindle Output Address S1=8,
то есть 1-й шпиндель управления выдаёт основной сигнал блоку припасовки 8.

Приём сигналов датчика:

Параметр N0600 Spindle Input Type надо выбирать опцию EtherCAT.
На параметр N0601 Spindle Input Address надо записать адрес того блока припасовки, с которого желаем получить сигналы датчика.

☞ ***Внимание!*** *Параметр N0603 Spindle Output Address и параметр N0601 Spindle Input Address могут быть различными!*

Пример:
Параметр
N0605 Spindle Name2 S1=1
определяет адрес шпинделя 1 как S1.
Адрес блока припасовки, или привода шпинделя S1среди установок EtherCAT: 8.
Сигналы датчика желаем получить сдатчика, оборудованного на шпинделе (не с двигателя)
Сигналы датчика, оборудованного на шпинделе принимает блок припасовки TTLAI, адрес которого 21.
При этом N0601 Spindle Input Address S1=21,
то есть шпиндель 1 управления получает сигнал датчика с блока припасовки 21.

Ссылка в программе PLC на оси, шпиндели и приводы

Символы NC
AN_ xx и AP_xx
относятся всегда к осям. На эти символы надо ссылаться всегда на основании индекса оси:
#0: 1-я ось, ..., #31: 32-я ось.
Символы NC
SN_ xx и SP_xx
относятся всегда к шпинделям. На эти символы надо ссылаться всегда на основании индекса шпинделя:
#0: 1-й шпиндель, ..., #15: 16 -й шпиндель ó.
Символы NC
DN_ xx и DP_xx
относятся всегда к приводам, или блокам припасовки. На эти символы надо ссылаться всегда на основании индекса привода:
#0: 1-й привод, ..., #47: 48-й привод.

Индекс оси, или шпинделя данной оси, или шпинделя может отличаться от индекса привода данной оси, или шпинделя!

Штатные сигналы блоков припасовки (вводы)

DN_INC: На приводе датчик с приращением

Если указатель DN_INC=0, на двигателе имется абсолютный датчик EnDAT, если указатель DN_INC=1, на двигателе имется датчик с приращением.

Контрольные сигналы блоков припасовки (выводы)

DP_ERRCLR: Удаление ошибки привода (Удаление ошибки блока припасовки)

Если в блоке припасовки имеется ошибка, то есть DN_ERR>0 (регистр ошибки не 0) ошибку надо удалить.

Состояние DP_ERRCLR=1 удаляет ошибку. Сигнал надо одержать в 1 до тех пор, пока регистр ошибки DN_ERR не будет 0, или не наступает состояние DN_ERROR=0.

Бинарные указатели ошибок блоков припасовки (вводы)

Программа PLC не должна выводить различные сообщение об ошибке привода, это выполняет NC. Надо выдавать только команду для удаления ошибки путём установки указателя DP_ERRCLR=1.

DN_ERROR: Имеется ошибка привода (DN_ERR>0) (Имеется ошибка блока припасовки)

DN_ERROR=1, если один из дальнейших битов регистра ошибок выводит сообщение об ошибке.

DN_EDERR1: Ошибка датчика

DN_EDERR1=1, если блок припасовки наблюдает ошибку с датчика.

DN_ECTERR: Ошибка превышения времени EtherCAT

DN_ECTERR=1, если в блоке припасовки срабатывал таймер WatchDog коммуникации EtherCAT. Коммуникация EtherCAT непригодная.

Вводные регистры блоков припасовки с двойным словом

DN_STAT: Штатный регистр привода (DWORD) (Штатный регистр блока припасовки)
Штатные биты блока припасовки доступны символом с двойным словом.

DN_ERR: Регистр указателей ошибки привода (DWORD) (Регистр указателей ошибки блока припасовки)
Биты ошибки блока припасовки доступны символом с двойным словом.

Выводные регистры приводов с двойным словом

DP_CTRL: Контрольный регистр привода (DWORD) (Контрольный регистр блока припасовки)
Контрольные сигналы припасовки доступны символом с двойным словом.
Обращение контрольным словом в случае программного обоеспечения ECAT-TACHO при блоке припасовки TTLAI:
Через PLC в верхнее слово DP_CTRL (от 16 до 31 битов) надо записать целое число по следующей формуле:
(число импульсов датчика*4)*(максимальное число оборотов двигателя/60)/5000
Если сюда записать ноль, тогда блок не выполняет тахокомпенсацию, он выдаёт аналоговый сигнал согласно основному сигналу.

7.9 Клавиши функции, доступные из PLC

Для программы PLC доступны 32 шт. клавиш функции на экране управления. Надпись клавиш функции надо задавать в программе PLC на нижней правой панели PLC Editor, щёлкнув на ушко клавиш PLC.
К 32 клавишам функции принадлежат 2 регистра с двойным словом: вводной регистр к клавишам, а выводной - к лампе клавиши.

Переменные клавиш функции с двойным словом:

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
N_MSG	32 клавиш функции на экране, которые может использовать PLC (DWORD)	**P_MSG**	32 ламп клавиш функции на экране, которые может использовать PLC (DWORD)

К регситрам N_MSG и P_MSG, как референции можно создать бинарные символы, обозначающие отдельные клавиши функции и их лампы.

Пример:
Пусть будет символ B_LUB символом ручного запуска насоса смазки, который желаем запускать с клавиши функции F23.
В коробку текста клавиши F23 на панеле клавиш PLC написана надпись СМАЗКА СУППОРТА, которая появится на экране на соответствующей клавиши функции.
Панель Добавка символа заполнять следующим образом:

Выбрать задание референции,
Символическое имя: B_LUB, клавиша смазки суппорта
База: N_MSG,
Смещение: 0,
Бит: 22 (23-я клавиша = 22-й бит).
Символическое имя: L_LUB, лампа клавиши смазки суппорта
База: P_MSG,
Смещение: 0,
Бит: 22 (23-я клавиша = 22-й бит).

7.10 Включатели позиции, устанавливаемые параметром

На параметре можно принять 32 шт. включателей позиции.
Параметр

N1100+n SWn Axis Number

подскажет, что на какую ось установить n-й включатель.
Параметр

N1132+n SWn Min Pos n

определяет станочную позицию конца n-го включателя, выходящего в отрицательную сторону на данной оси.
Параметр

N1164+n SWn Max Pos n

определяет станочную позицию конца n-го включателя, выходящего в положительную сторону на данной оси.
Перед каждым **циклом PLC** определяется управлением, что существует ли n-й включатель. Если да, тогда на основании абсолютной позиции, расчитанной ***с датчика*** выделённой для включателя оси, оно будет обращаться следующими сигналами для n-го включателя (указатель PLC с индексом n-1).
Сигналы включателей позиции являются вводными сигналами для PLC.

Бинарные переменные включателей позиции:

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
N_SW	1-я ось находится на включателе с параметром		
N_SWN	1-я ось находится в отрицательном направлении от включателя с параметром		
N_SWP	1-я ось находится в положительном направлении от включателя с параметром		

Вводы включателей позиции

N_SW: 1-я ось находится на включателе с параметром

Если выделённая ось находится на 1-ом включателе, то есть

N1133 SW1 Min Pos 1 < Позиция < N1165 SW1 Max Pos 1

указатель примет состояние 1.

N_SWN: 1-я ось находится в отрицательном направлении от включателя с параметром
Если выделённая ось находится в отрицательном направлении от 1-го включателя, то есть
Позиция < N1133 SW1 Min Pos 1
указатель примет состояние 1.

N_SWP: 1-я ось находится в положительном направлении от включателя с параметром
Если выделённая ось находится в положительном направлении от 1-го включателя, то есть
Позиция > N1133 SW1 Max Pos 1
указатель примет состояние 1.

Остальные включатели доступны с индексированной ссылкой по сравнению к 1-му включателю. Отдельным включателям можно присвоить и символическое название, но надо следить за тем, чтобы эти названия принимали с относительной ссылкой по сравнению к символам N_SW, ... и т.д., как к базе..

☞ ***Внимание**: Надо принимать во внимание у применении знаков максимальную скорость движения осей, длину включателя и частоту квантования (время цикла PLC).*

Переменные включателей позиции с двойным словом:

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
N_SW0	Регистр 1-го включателя, заданного параметром Position Switches (DWORD)		

N_SW0: Регистр 1-го включателя, заданного параметром Position Switches (DWORD)
Сигналы 1-го включателя позиции доступны в регистре N_SW0 и с двойным словом. Регистр остальных включателей можно вычитать с индексированным доступом.

7.11 Доступ к группе параметров PLC Constants

В группе параметров PLC Constants стоит на распоряжение программы PLC параметр типа

64 шт. бинарных,

32 шт. целых с двойным словом и

32 шт. с плавающей запятой,

свободного использования. Эти параметры программа PLC может непосредственно читать из памяти PLC, для бинарных параметров может определить контакт.
Параметры PLC нельзя непосредственно писать из программы PLC!

Бинарные параметры PLC:

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
N_P00	Бит P00 параметра PLCBits0		
N_P01	Бит P01 параметра PLCBits0		
N_P02	Бит P02 параметра PLCBits0		
N_P03	Бит P03 параметра PLCBits0		
N_P04	Бит P04 параметра PLCBits0		
N_P05	Бит P05 параметра PLCBits0		
N_P06	Бит P06 параметра PLCBits0		
N_P07	Бит P07 параметра PLCBits0		
N_P10	Бит P10 параметра PLCBits1		
N_P11	Бит P11 параметра PLCBits1		
N_P12	Бит P12 параметра PLCBits1		
N_P13	Бит P13 параметра PLCBits1		
N_P14	Бит P14 параметра PLCBits1		
N_P15	Бит P15 параметра PLCBits1		
N_P16	Бит P16 параметра PLCBits1		
N_P17	Бит P17 параметра PLCBits1		
N_P20	Бит P20 параметра PLCBits2		
N_P21	Бит P21 параметра PLCBits2		
N_P22	Бит P22 параметра PLCBits2		
N_P23	Бит P23 параметра PLCBits2		
N_P24	Бит P24 параметра PLCBits2		
N_P25	Бит P25 параметра PLCBits2		
N_P26	Бит P26 параметра PLCBits2		
N_P27	Бит P27 параметра PLCBits2		
N_P30	Бит P30 параметра PLCBits3		
N_P31	Бит P31 параметра PLCBits3		

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
N_P32	Бит P32 параметра PLCBits3		
N_P33	Бит P33 параметра PLCBits3		
N_P34	Бит P34 параметра PLCBits3		
N_P35	Бит P35 параметра PLCBits3		
N_P36	Бит P36 параметра PLCBits3		
N_P37	Бит P37 параметра PLCBits3		
N_P40	Бит P40 параметра PLCBits4		
N_P41	Бит P41 параметра PLCBits4		
N_P42	Бит P42 параметра PLCBits4		
N_P43	Бит P43 параметра PLCBits4		
N_P44	Бит P44 параметра PLCBits4		
N_P45	Бит P45 параметра PLCBits4		
N_P46	Бит P46 параметра PLCBits4		
N_P47	Бит P47 параметра PLCBits4		
N_P50	Бит P50 параметра PLCBits5		
N_P51	Бит P51 параметра PLCBits5		
N_P52	Бит P52 параметра PLCBits5		
N_P53	Бит P53 параметра PLCBits5		
N_P54	Бит P54 параметра PLCBits5		
N_P55	Бит P55 параметра PLCBits5		
N_P56	Бит P56 параметра PLCBits5		
N_P57	Бит P57 параметра PLCBits5		
N_P60	Бит P60 параметра PLCBits6		
N_P61	Бит P61 параметра PLCBits6		
N_P62	Бит P62 параметра PLCBits6		
N_P63	Бит P63 параметра PLCBits6		
N_P64	Бит P64 параметра PLCBits6		
N_P65	Бит P65 параметра PLCBits6		
N_P66	Бит P66 параметра PLCBits6		
N_P67	Бит P67 параметра PLCBits6		
N_P70	Бит P70 параметра PLCBits7		
N_P71	Бит P71 параметра PLCBits7		
N_P72	Бит P72 параметра PLCBits7		
N_P73	Бит P73 параметра PLCBits7		
N_P74	Бит P74 параметра PLCBits7		
N_P75	Бит P75 параметра PLCBits7		

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
N_P76	Бит P76 параметра PLCBits7		
N_P77	Бит P77 параметра PLCBits7		

Отдельным параметрам N_Pij можно присвоить и символические названия, но надо следить за тем, чтобы названия принимали с относительной ссылкой по сравнению к символам N_Pij, как к базе.

Параметры PLC с двойным словом:

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
N_PDW1	Значение параметра PLC DWord1 (DWORD)		

N_PDW1: Значение параметра PLC DWord1 (DWORD)

Из регистра N_PDW1 можно вычитать значение параметра N1209 PLC DWord1. Доступ к остальным значениям параметров возможно с индексированной командой.

Отдельным параметрам можно присвоить и символические названия, но надо следить за тем, чтобы названия принимали с относительной ссылкой по сравнению к символам N_PDW1, как к базе.

Параметры PLC с плавающей запятой:

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
N_PDB1	Значение параметра PLC Double1 (double)		

N_PDB1: Значение параметра PLC Double1 (double)

Из регистра N_PDB1 можно вычитать значение параметра N1241 PLC Double1. Доступ к остальным значениям параметров возможно с индексированной командой.

☞ ***Внимание!*** *При индексированной ссылке надо принимать во внимание, что значение double изображено в 2 DWORD, поэтому индексом надо шагать по 2.*

Отдельным параметрам можно присвоить и символические названия, но надо следить за тем, чтобы названия приняли с относительной ссылкой по сравнению к символам N_PDB1, как к базе.

7.12 Глобальные переменные

Глобальными или общими переменными являются такие переменные, идущие от NC в PLC, или из PLC в NC, которые общие для всех каналов.
Переменные,

начинающиеся с N идут от NC в PLC, а
начинающиеся с P - от PLC в NC.

7.12.1 Бинарные глобальные переменные

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
N_P2MS	Сигнал часов с периодом 2 Time Slice (только в модуле Int0, непосредственный запрос)	**P_MONREQ**	Запрос включения Machine On из PLCl
N_P2T	Сигнал часов с периодом 2 PLC	**P_HOLD0**	Стоп подачи в каждом канале
N_P100MS	Сигнал часов с периодом 100 мсек	**P_SHTDNREQ**	Запрос останова NC
N_P1S	Сигнал часов с периодом 1сек		
N_P1M	Сигнал часов с периодом 1мин		
N_ON	всегда 1		
N_OFF	всегда 0		
N_B7	Сохранено		
N_NVRAMOK	Незабывающие переменные PLC успешно скачались обратно		
N_FIRSTCC	Первый прогон после включения		
N_NCREADY	NC готов к работе		
N_MONST	Состояние сигнала Machine ON		
N_MONDIS	Включение сигнала Machine ON запрещено		
N_CLRMSG	Удаление сообщения		
N_MSGA	Сообщение на экране		
N_MSG0	Сообщение PLC на экране		
N_MSG1	Сообщение оси, или шпинделя на экране		
N_MSG2	Сообщение о подготовке кадра канала на экране		
N_MSG3	Сообщение о выполнении канала на экране		
N_MSG4	Макросообщение об ошибке		

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
	(#3000) на экране		
N_MSG5	Макросообщение (#3006) на экране		
N_MSG6	Сохранено		
N_MSG7	Сохранено		
N_MSG8	Сообщение о системе реального времени на экране		
N_MSG9	Сообщение о связи человек-станок на экране		
N_TLSRCH	Поиск инструмента под выполнением		
N_TLMD	Таблицы оператора инструментов под изменением		
N_TLSV	Таблицы оператора инструментов под сохранением		
N_TLEDT	Таблицы оператора инструментов под редактированием		
N_PTEDT	Таблица оператора палитры под редактированием		
N_SIMU	Прогон программного обеспечения NC идёт на симуляторе (на PC)		
N_NVECAT	Незабывающие переменные доступны через EtherCAT		
		P_DIR	Запрет доступа: Операции с каталогом
		P_PRGE	Запрет доступа: Редактирование программы
		P_WOFFS	Запрет доступа: Нулевые точки заготовок
		P_COMPG	Запрет доступа: Геометрические коррекции
		P_COMPW	Запрет доступа: Коррекции по износу
		P_TLTAB	Запрет доступа: Таблицы инструментов
		P_MAC	Запрет доступа: Макропеременные
		P_TRCTR	Запрет доступа: Счётчики времени, заготовок

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
		P_RUNAUT	Запрет доступа: Автопрогон
		P_RUNMDI	Запрет доступа: Прогон MDI
		P_PAR	Запрет доступа: Параметры
		P_PLC	Запрет доступа: Программа PLC
		P_SVRC	Запрет доступа: Сервисные операции
		P_PTTAB	Запрет доступа: Таблица палитры

Бинарные глобальные переменные, идущие от NC в PLC

N_P2MS: Сигнал часов с периодом 2 Time Slice (только в модуле Int0, непосредственный запрос)
N_P2T: Сигнал часов с периодом цикла 2 PLC
N_P100MS: Сигнал часов с периодом 100 мсек
N_P1S: Сигнал часов с периодом 1 сек
N_P1M: Сигнал часов с периодом 1 мин

Из сигналов часов, подающихся NC, сигналом N_P2MS из-за частоты сигнала имеет смысл использоваться только в модуле Int0. Остальные сигналы часов можно использовать и в Главной программе.

N_ON: всегда 1
N_OFF: всегда 0

Этими двумя символами можно пользоваться тогда, если, например, надо привязать к какому-то вводу коробки команд твёрдое значение, или один из контактов желаем комментировать.

N_NVRAMOK: Незабываюшие переменные PLC успешно скачались обратно

Если в незабывающем магазине PLC сохранены данные, в первом цикле после включения (N_FIRSTCC=1) можно просмотреть указатель. Если значение указателя 0, переменные PLC повреждены.

N_FIRSTCC: Первый пробег после включения

После включения, или перезагрузки управления указатель в период цикла 1 PLC будет 1. За это время должна выполнить программа PLC необходимые инициирования.

N_NCREADY: NC готов к работе

Он указывает готовое состояние NC к работе.

N_MONST: Состояние сигнала Machine ON

Если указатель во включенном состоянии станка переходит в 0, надо инициировать аварийный останов из PLC.

N_MONDIS: Включение сигнала Machine ON запрещено

Он инициирует включение станка, то есть включить указатель P_MONREQ разрешено только тогда, если включение станка не запрещено, то есть указатель N_MONDIS будет 0.

N_CLRMSG: Удаление сообщения (активное 1)

Сообщение выводится в самой верхней строке дисплея.

Если одновременно ожидается несколько сообщений, в поле сообщения будет видно всегда сообщение, поступившее последним по времени. (Last in First Out).

Клавиша

CANCEL

удаляет всегда видное на дисплее, поступившее последним сообщение, и указатель N_CLRMSG установит в 1.

Если щёлкнуть в поле сообщения, в раскручивающем списке видны будут все сообщения. При этом можно выбирать произвольно сообщение и с клавишей функции

Удаление (выбранное)

можно его удалить. При этом работает (и без нажатия клавиши CANCEL) указатель N_CLRMSG.

Можно выбирать и клвишу функции

Удалить всё,

под действием которого удаляются все сообщения из буфера. И в этом случае работает указатель N_CLRMSG.

В программе PLC надо пользоваться всегда указателем N_CLRMSG для удаления сообщений.

N_MSGA: Сообщение на экране

Значение указателя будет 1, если любое сообщение имеется в самой верхней строке экрана, в строке сообщений. Указатель можно использовать например для включения станочной лампы сообщения об ошибке.

N_MSG0: Сообщение PLC на экране

N_MSG1: Сообщение оси, или шпинделя на экране

N_MSG2: Сообщение о подготовке кадра на экране

N_MSG3: Сообщение о выполнении канала на экране

N_MSG4: Макросообщение об ошибке (#3000) на экране
N_MSG5: Макросообщение (#3006) на экране
N_MSG6: Сохранено
N_MSG7: Сохранено
N_MSG8: Сообщение о системе реального времени на экране
N_MSG9: Сообщение о связи человек-станок на экране

Указатели N_MSG0, ..., N_MSG9 показывают род сообщения, поступившее последним, видного в самой верхней строке экрана, в строке сообщений.

N_TLSRCH: Поиск инструмента под выполнением
N_TLMD: Таблицы оператора инструментов под изменением
N_TLSV: Таблицы оператора инструментов под сохранением
N_TLEDT: Таблицы оператора инструментов под редактированием

Приведенные выше указатели дают информацию о причине занятости Таблицы оператора инструментов.

N_PTEDT: Таблица оператора палитры под редактированием

Редактируется какая-то строка таблицы оператора палитры.

N_SIMU: Прогон программного обеспечения NC идёт на симулаторе (на PC)

Если прогон программного обеспечения NC идёт на симулаторе, то есть на PC, об этом факте PLC получает сообщение на основании сигнала N_SIMU, состояние которого 1.

N_NVECAT: Незабывающие переменные доступны через EtherCAT

Если нет аккумулятора к управлению, а оно подключено к блоку FRAM, работающий через EtherCAT, то N_NVECAT=1. При этом при выключении объём памяти PLCот PLCNVRAM до PLCRAM не будет сохранён.

О сохранении сохраняемых переменных PLC, далее об их зачитании должен позаботиться програмист PLC, с использованием команд MR10 и MW11.

Бинарные глобальные переменные, идущие от PLC в NC

P_MONREQ: Запрос включения Machine On изPLC

Вывод MON (Machine ON), выполняющий включение станка, имеется на каждом головном блоке EtherCAT.

Установку адреса вывода MON можно выполнить ***в меню Сервис*** управления, загрузив окно ***установки ECAT***, при установок ***станочная панель оператора.*** Выбирая на левой панели блок EtherCAT-Head, загрузить ушко ***Setting,*** и установить ***адрес*** вывода MON. Если на головном блоке выбрать опцию

Not used,

на этом блоке обращения выводом MON не будет. Если установить адрес

1, 2, и т.д,.

на том блоке будет обращение выводом MON.

Включение указателя P_MONREQ включает все выводы MON на головном блоке, которые установлены не на Not used. Указатель разрешено включить только тогда, если включение станка не запрещено, то есть указатель N_MONDIS будет 0.

P_HOLD0: Подача стоп в каждом канале

Включение указателя в 1 останавливает движение всех осей во всех каналах даже тогда, если override и стоп запрещены. Если override и стоп запрещены, в случае нарезания резьбы метчиком, или обточкой, PLC должен остановить шпиндель.

P_SHTDNREQ: Запрос останова NC (Shutdown Request)

NC можно снабжать и с аккумуляторным блоком питания, способным обеспечивать питание управления и необходимую электронику EtherCAT в течение 1-2 минут. В случае выпадания напряжения сети, или выключения главного включателя станка блок питания даст сигнал произвольному вводу интерфейса для PLC. Программа PLC может запросить останов системы включением указателя P_SHTDNREQ.

P_DIR: Запрет доступа: Операции с каталогом

В состоянии P_DIR=1

нельзя удалить,

нельзя изменить файл, или папку.

Может создать новый файл, далее может скопировать файл.

P_PRGE: Запрет доступа: Редактирование программы

В состоянии P_PRGE=1 запрещено редактирование всех программ деталей.

P_WOFFS: Запрет доступа: Нулевые точки заготовок

В состоянии P_WOFFS=1 нельзя изменить нулевые точки заготовок, ни в таблице, ни с замером.

P_COMPG: Запрет доступа: Геометрические коррекции

В состоянии P_COMPG=1 нельзя изменить геометрические коррекции.

P_COMPW: Запрет доступа: Коррекции по износу

В состоянии P_COMPW=1 нельзя изменить коррекции по износу.

P_TLTAB: Запрет доступа: Таблицы инструментов

В состоянии P_TLTAB=1 нельзя изменить ни одного элемента Таблицы оператора инструментов.

P_MAC: Запрет доступа: Макропеременные

В состоянии P_MAC=1 нельзя изменить ни одну локальную, или глобальную макропеременную.

P_TRCTR: Запрет доступа: Счётчики времени, заготовок

В состоянии P_TRCTR=1 нельзя изменить ни одного значения таблицы подсчёта времени, заготовок.

P_RUNAUT: Запрет доступа: Автопрогон

В состоянии P_RUNAUT=1

нельзя выделить программу для прогона в автоматическом режиме,

нельзя удалить программу, выделённую для прогона в автоматическом режиме.

P_RUNMDI: Запрет доступа: Прогон MDI

В состоянии P_RUNMDI=1

нельзя выделить программу для прогона в режиме ручного ввода данных,

нельзя удалить программу, выделённую для прогона в режиме ручного ввода данных.

P_PAR: Запрет доступа: Параметры

В состоянии P_PAR=1

нельзя изменить (нельзя редактировать) параметр,

нельзя загрузить параметр,

но может сохранить параметр, или экспортировать.

P_PLC: Запрет доступа: Программа PLC

В состоянии P_PLC=1

нельзя изменить (нельзя редактировать) программу PLC,

нельзя загрузить программу PLC,

но может сохранить программу PLC, может смотреть зелёный поток.

P_SVRC: Запрет доступа: Сервисные операции

Состояние P_SVRC=1 может повлиять на сервисные операции следующим образом:

Тест I/O: операция BitSet, BitReset, HexaSet запрещена,

Szimb. I/O, Логический анализатор: при сохранении изменение запрещено

Установки ECAT: редактирование панелей запрещено.

P_PTTAB: Запрет доступа: Таблица палитры

В состоянии P_PTTAB=1 таблица палитры не редактируемая.

7.12.2 Глобальные переменные с двойным словом

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
N_ACTMSG	Опознавательное число активного сообщения, водное на дисплее (DWORD)	**P_CHSEL**	Для какого канала действительна тастатура (0,1,2...) (DWORD)

Глобальные переменные с двойным словом, идущие от NC в PLC

N_ACTMSG: Опознавательное число активного сообщения, видное на дисплее (DWORD)

В самом верхнем окне, расположенном слева от строки сообщения можно вычитать 8-мизначный десятичный код последнего сообщения, выведённого на дисплее. В регистре N_ACTMSG появится этот код.

Глобальные переменные с двойным словом, идущие от PLC в NC

P_CHSEL: Для какого канала действительна тастатура (0,1,2...) (DWORD)

Это имеет значение в многоканальном управлении. При запросе отображения экрана, например в точке меню Смещения, выбирая позицию Замера, управлением отображается первым экран Замера того канала, номер которого было установлен в регистре P_CHSEL. После этого уже можно пролистать клавишами PgUp, PgDn между различными каналами.

В одноканальном управлении им не надо пользоваться.

7.13 Переменные оператора для оси

Переменные оператора для оси являются такимипеременные, идущие от NC в PLC, илииз PLC в NC, которые индексируются по осям. Все перечисленные здесь символы относятся к первой (с индексом 0) оси. Соответствующая переменная остальных осей доступна с индексированной адресовкой.Управление обращается не более 32 осями.
Переменные,

начинающиеся с AN, идут от NC в PLC (Вводы), а
начинающиеся с AP от PLC в NC (Выводы).

7.13.1 Бинарные переменные для оси

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
AN_DETCHA	Подтверждение выключения оси	**AP_DETCHR**	Запрос выключения оси
AN_OPNA	Подтверждение открытия контура оси	**AP_OPNR**	Запрос открытия контура регулятора позиции
AN_INPOS	Ось в позиции	**AP_END**	Запрет наблюдения за датчиком
AN_AXALM	Ошибка на оси	**AP_FLWU**	Включение сервослежения
AN_RAPR	Запрос быстрого хода	**AP_JOGP**	Запрос Jog в + направлении
AN_MTNRP	Запрос движения в положительное направление	**AP_JOGN**	Запрос Jog в - направлении
AN_MTNRN	Запрос движения в отрицательное направление	**AP_DECSW**	Замедлитель на включателе
AN_LUBR	Запрос смазки на оси	**AP_RAPD**	Запрет движения быстрого хода
AN_RPE	Имеется действительная точка референции	**AP_MTNDP**	Запрет движения в положительное направление
AN_REFEND	Конец приёма референции	**AP_MTNDN**	Запрет движения в отрицательное направление
AN_OTP	В конечном положении в положительном направлении	**AP_LIMP**	На включателе положительного конечнего положения
AN_OTN	В конечном положении в отрицательном направлении	**AP_LIMN**	На включателе отрицательного конечнего положения
AN_REFP1	Ось в точке Референции	**AP_LIMSELP**	Выбор диапазона 1В положительного конечнего положения с параметром
AN_REFP2	Ось в Reference Position 2	**AP_LIMSELN**	Выбор диапазона 1В отрицательного конечнего положения с параметром
AN_REFP3	Ось в Reference Position 3	**AP_LCK**	Ось закрыта: Запрет любого движения
AN_REFP4	Ось в Reference Position 4	**AP_DISPD**	Индикация оси выключена

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
AN_PARKA	Подтверждение ось паркует	**AP_PARKR**	Запрос парковки оси
AN_SYNCA	Подтверждение синхронизации оси	**AP_SYNCR**	Запрос синхронизации оси
AN_MIXA	Подтверждение смены оси.	**AP_MIXR**	Запрос смены оси
AN_SPRPNA	Подтверждение совмещения на ось	**AP_SPRPNR**	Запрос совмещения на ось
AN_EGBS	EGB оси slave	**AP_DIARAD**	Ось запрограммирована по диаметру
AN_INDP	Index оси в позиции	**AP_SSLOP**	Размыкание контура оси-аппликата и приём только основного сигнала
AN_MIXM	Мастер-ось смены оси	**AP_FEEDD**	Запрет движения подачи
AN_SYNCM	Мастер-ось синхронного движения	**AP_PLCR**	PLC запрашивает ось
AN_SPRPNM	Мастер-ось движения совмещения	**AP_GOR**	Пусть идёт движение по оси PLC
AN_PLCA	NC отдал ось для PLC	**AP_RES**	Ось PLC reset
AN_BEPTY	Буфер кадра оси PLC пустой	**AP_EFD**	Запрет измерения хода
AN_GOA	Ось PLC перемещается	**AP_RPE**	Имеется действительная референтная точка
AN_IEPTY	Перемещение оси PLC закончилось	**AP_MIRR**	Запрос смены направления оси (отражения)
AN_MIRA	Подтверждение через PLC запроса смены направления оси (отражения)		

Бинарные переменные оси, идущие от NC в PLC

AN_DETCHA: Подтверждение выключения оси

AN_DETCHA является указателем подтверждения сигнала запроса выключения оси AP_DETCHR.

На запрос выключения оси AP_DETCHR=1 оператор оси даст обратно AN_DETCHA=1. На запрос включения AP_DETCHR=0 оператор оси даст обратно AN_DETCHA=0. См.: Описание указателя AP_DETCHR.

AN_OPNA: Подтверждение размыкания контура оси

AN_OPNA является указателем подтверждения сигнала запроса размыкания контура регулятора позиции AP_OPNR.

На запрос размыкания контура AP_OPNR=1 регулятор позиции даст обратно AN_OPNA=1. На запрос замыкания контура AP_OPNR=0 регулятор контура даст обратно AN_OPNA=0. См.: Описание указателя AP_OPNR.

AN_INPOS: Ось в позиции

Если разность позиции команды оси и позиции, измеренной от датчика находится в пределах окна, определённого параметром N0516 Inpos, значение сигнала 1.

AN_AXALM: Ошибка на оси

Если оператор оси обнаруживает сервоошибку у любого элемента сервоконтура (датчик, привод, работающий через EtherCAT, контур регулятора позиции) он устанавливает указатель AN_AXALM в 1.

☞ ***Внимание!*** *Задачей программы PLC является реагировать на сообщение об ошибке, остановить привод, вызывать аварийное состояние! NC не выключает станок автоматически, лишь по команде PLC!*

AN_RAPR: Запрос быстрого хода

Интерполятор обращается указателем вместе с указателями запроса движения AN_MTNRP, AN_MTNRN.

Если AN_RAPR=0, интерполятор хочет перемещаться с подачей, и ожидает состояние AP_FEEDD=0 (движение подачи разрешено) и AP_RAPD=1 (движение быстрого хода запрещено).

Если AN_RAPR=1, интерполятор хочет перемещаться с быстрым ходом, и ожидает состояние AP_FEEDD=1 (движение подачи запрещено) и AP_RAPD=0 (движение быстрого хода разрешено).

См. ещё указатели AN_MTNRP, AN_MTNRN, AP_RAPD, AP_FEEDD.

Обращение этими указателями можно пользовать например при изменении передачи между движениями подачи и быстрого хода.

AN_MTNRP: Запрос движения в положительном направлении

AN_MTNRN: Запрос движения в отрицательном направлении

Перед запуском каждого движения интерполятор переключает указатель запроса движения соответствующего направления в 1. В случае круговой интерполяции он запросит движение на осях в плоскости круга в обе направление.

Если PLC разрешает движение в соответствующее направление, он должен установить указатели AP_MTNDP=0, или AP_MTNDN=0.

См. ещё указатели AN_RAPR, AP_RAPD, AP_FEEDD.

Указатели можно использовать например для фиксирования, разжима осей.

AN_LUBR: Запрос смазки на оси

Если значение параметра N1273 Lubrication Distance не 0, и величина смещения на данной оси превысила расстояния, записанного в параметр, указатель AN_LUBR в силу 1 цикла PLC записывается в 1.

По сигналу этому можно например запускать насос смазки..

AN_RPE: Имеется действительная точка Референции

В состоянии AN_RPE=1 на оси уже выполнен приём референтной точки. В случае абсолютных датчиков указатель всегда 1.

AN_REFEND: Конец приёма референции

Указатель работает в режиме приёма референтной точки. Он будет записан в 1 тогда, если на данной оси выполнен приём референтной точки и ось уже не движется.

AN_OTP: В конечном положении в положительном направлении

AN_OTN: В конечном положении в отрицательном направлении

Если ось дошла до конечного положения, установленного в группе параметров Axis Limits, указатель соответствующего направления записывается в 1.

AN_REFP1: Ось в референтной точке

|Референтная точка – станочная позиция, измеренная от датчика| < N0516 Inpos

Если станочная позиция оси, измеренная от датчика

референтная точка оси

находится внутри круга с радиусом, определённым параметром N0516 Inpos, значение сигнала 1.

AN_REFP2: Ось в Reference Position 2

|N0201 Reference Position2 – станочная позиция, измеренная от датчика| < N0516 Inpos

Если станочная позиция оси, измеренная от датчика

позиция, заданная параметром N0201 Reference Position2

находится внутри круга с радиусом, определённым параметром N0516 Inpos, значение сигнала 1.

AN_REFP3: Ось в Reference Position 3

|N0202 Reference Position3 – станочная позиция, измеренная от датчика| < N0516 Inpos

Если станочная позиция оси, измеренная от датчика

позиция, заданная параметром N0202 Reference Position2

находится внутри круга с радиусом, определённым параметром N0516 Inpos

значение сигнала 1.

AN_REFP4: Ось в Reference Position 4

|N0203 Reference Position4 – станочная позиция, измеренная от датчика| < N0516 Inpos

Если станочная позиция оси, измеренная от датчика

позиция, заданная параметром N0203 Reference Position4

находится внутри круга с радиусом, определённым параметром N0516 Inpos

значение сигнала 1.

Позиции, относящиеся к сигналам AN_ REFP2, AN_ REFP3, AN_ REFP4 являются замеренными позициями места смены. Прежде, чем PLC выполнил бы какую-то деятельность, для которой нужна позиция смены оси, анализируя PLC сигнал, может убедиться о том, что ось находится ли в правильном положении.

AN_PARKA: Подтверждение ось паркует

AN_PARKA является указателем подтверждения запроса парковки AP_PARKR.

На запрос парковки AP_PARKR=1 оператор оси даст обратно AN_PARKA=1. На запрос выключения парковки AP_PARKR=0 оператор оси даст обратно AN_PARKA=0.См.: Описание указателя AP_PARKR.

AN_SYNCA: Подтверждение синхронизации оси

AN_SYNCA является указателем подтверждения запроса синхронизации оси AP_SYNCR. Он даст обратно сигнал на указателе (на индексе), относящимся к оси (к оси-аппликата), запрашивающей синхронизацию.

На запрос синхронизации AP_SYNCR=1 оператор оси даст обратно AN_SYNCA=1.

На запрос выключения синхронизации AP_SYNCR=0 оператор оси даст обратно AN_SYNCA=0. См.: Описание указателя AP_SYNCR.

AN_MIXA: Подтверждение смены оси

AN_MIXA является указателем подтверждения запроса смены оси AP_MIXR. Он даст обратно сигнал на указателе (на индексе), относящимся к сменяемой оси.

На запрос смены оси AP_MIXR=1 оператор оси даст обратно AN_MIXA=1.На запрос выключения смены оси AP_MIXR=0 оператор оси даст обратно AN_MIXA=0. См.: Описание указателя AP_MIXR.

AN_SPRPNA: Подтверждение совмещения на ось

AN_SPRPNA является указателем подтверждения запроса совмещённого движения AP_SPRPNR. Он даст обратно сигнал на указателе (на индексе), относящимся к оси (к оси-аппликата), запрашивающей совмещение.

На запрос совмещённого движения AP_SPRPNR=1 оператор оси даст обратно На запрос выключения совмещённого движения AN_SPRPNA=1. AP_SPRPNR=0 оператор оси даст обратно AN_SPRPNA=0. См.: Описание указателя AP_SPRPNR.

AN_EGBS: EGB ось slave

Указатель переключается в 1,

если данная ось выделена параметром N1802 EGB Slave осью-аппликату электронного привода и

находится в силе команда связь привода (зубонарезание) G81.8.

AN_INDP: Index оси в позиции

Значение указателя 1, если позиция оси находится внутри круга позиции индекса с радиусом, определённым параметром N0516 Inpos.

Позицией индекса называем с какой-то точки зрения выделённые дискретные позиции на одной оси, расположенные на равное расстояние друг от друга. Например, поворотный стол с диском Хирта, которую можно посадить по 5 градусам, индексировать по 5 градусам.

С установкой #3 IDX=1 параметра N0106 Axis Properties можно выделить ось как индексированную ось. На параметре N0110 Indexing Amount можно установить расстояние индексирования. Например, в случае указанного выше поворотного стола с диском Хирта 5.

AN_MIXM: Мастер-ось смены оси

Если ось является сменной осью смены оси, указатель переключает в 1. *Он устанавливает указатель AN_MIXA на сменяемую ось (два индекса оси различные)!*

AN_SYNCM: Мастер-ось синхронного движения

Если ось является мастером-осью синхронного движения, указатель переключает в 1. *Он устанавливает указатель AN_SYNC на синхронную ось-аппликат (два индекса оси различные)!*

AN_SPRPNM: Мастер-ось движения совмещения

Если ось является мастером-осью движения совмещения, указатель переключает в 1. *Он устанавливает указатель AN_SPRPNM на ось-аппликат совмещения (два индекса оси различные)!*

AN_PLCA: NC отдал ось для PLC

AN_PLCA является указателем подтверждения запроса оси AP_PLCR со стороны PLC.

На запрос PLC оси AP_PLCR=1 оператор оси даст обратно AN_PLCA=1. На запрос возвращения для NC оси AP_PLCR=0 оператор оси даст обратно AN_PLCA=0. См.: Описание указателя AP_PLCR.

AN_BEPTY: Буфер кадра оси PLC пустой

Команды перемещения оси MOVCMD PLC имеют буфер с одним кадром по осям. Это означает, что тогда, если интерполятор взял один кадр из буфера для выполнения, PLC может загрузить следующий кадр. См. главу 6.15 Команда для перемещения оси: MOVCMD. На самом деле команда MOVCMD не выполняется до того, пока не может кадр вставить в буфер.

Если AN_BEPTY=1, буфер кадра пустой.

AN_GOA: Ось PLC перемещается

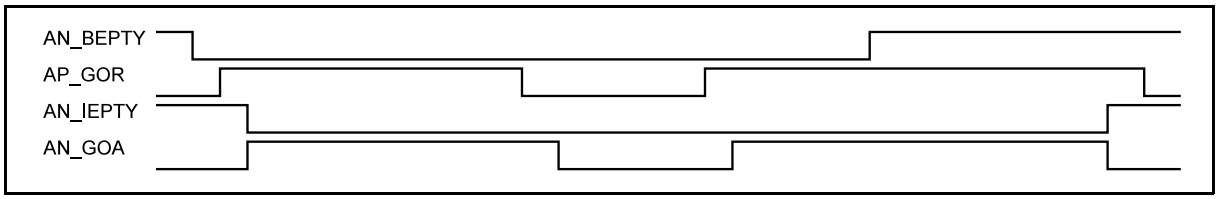
Если интерполятор оси PLC не пустой, то есть в состоянии запроса движения оси PLC AN_IEPTY=0, **И** AP_GOR=1, PLC, ось сдвинется и указатель примет состояние AN_GOA=1.

Если интерполятор оси PLC пустой, то есть в состоянии нет запроса движения оси PLC AN_IEPTY=1, **ИЛИ** AP_GOR=0 PLC, ось остановится и указатель примет состояние AN_GOA=0.

См. указатель AN_GOR.

AN_IEPTY: Перемещение оси PLC закончилось

Если перемещение оси PLC закончилось, то есть интерполятор опустел, указатель примет состояние 1.



AN_MIRA: Подтверждение через PLC запроса смены направления оси (отражения)

Программа PLC через индикатор AP_MIRR может запросить смену направления оси на одной из осей. Сигнал AN_MIRA является знаком подтверждения запроса. При этом в случае перемещения оси из программы деталей направление перемещения данной оси переключается в противоположное.

Направление ручного перемещения (клавиши jog, маховичок) не меняется.

Если значение сигнала:

=0: ось перемещается в первоначальное направление, установленное параметром из программы деталей,

=1: ось перемещается в направление, противоположное установленному параметром из программы деталей.

☞ ***Внимание!*** *Смену направления оси следует выполнить всегда в функции, предотвращающей буферизации кадра (смотри Группу параметров Program).*

Бинарные переменные оси, идущие от PLC в NC

AP_DETCHR: Запрос выключения оси

Если необходимо выключить работу какой-то оси, это можно запросить от системы с записыванием указателя AP_DETCHR соответствующей оси в 1. После того, что оператор оси остановил работу оси, даст обратно сигнал подтверждения AN_DETCHA=1. При включении обратно, указатель AP_DETCHR надо записать в 0, и дождаться, пока оператор оси восстановит состояние AN_DETCHA=0.

Если сигнал AN_DETCHA находится в состоянии 1,оператор оси не работает:

удаляет на сои заметку “имеется референтная точка”: AN_RPE=0,
не измеряет и не берёт в учёт позицию о датчике,
не замыкает контур регулирования позиции,
не выдаёт никакого сигнала в сторону привода оси.,
в программе деталей и в PLC нельзя ссылаться на соь,

ось не существует.

И наоборот, если AP_DETCHR=0 и NC выключает укащатель AN_DETCHA=0, ось снова существует.

Ось существует тогда, если

значение параметра N0002 Axis Assign> 0, и
AN_DETCHA=0.

☞ ***Внимание!*** *Выключение оси, а потом включение обратно надо выполнять всегда в функции, предотвращающей буферирования кадра (см. Группа параметров программы)*

Пример:

Токарный шпиндель надо преобразовать в ось C (поворотный стол), затем обратно в шпиндель. Пусть будет

номер оси C: 3 (N0002 Axis Assign A3=1, N0100 Axis Name1 A3=C) а
Номер шпинделя S1: 1. (N0605 Spindle Name2 S1=1)

После включения установится как шпиндель:

SP_SDETCHR,#0=0, SN_SDETCHA,#0=0
AP_DETCHR,#2=1, AN_DETCHA,#2=1

При желании преобразовать шпиндель S1 в ось C, надо запрограммировать соответствующую функцию M для подавления буфера и установить указатели следующим образом:

SP_SDETCHR,#0=1, SN_SDETCHA,#0=1
AP_DETCHR,#2=0, AN_DETCHA,#2=0

Начиная с этого сигналы датчика будет брать с того же привода шпинделя не оператор шпинделя S1, а оператор оси C. Основной сигнал будет выдавать не оператор оси S1 для привода шпинделя, а оператор оси C. Ось C можно перемещать, на адрес C можно ссылаться в программе деталей.

Преобразование оси C обратно в шпиндель S1 происходит подобным образом.

AP_OPNR: Запрос размыкания контура регулирования позиции

Если приходится размыкать контур регулирования позиции какой-то оси, это можно запросить от системы, записав указатель AP_OPNR соответствующей оси в 1. После того, что регулятор позиции остановил замыкание контура, возвращает сигнал подтверждения AN_OPNA=1. При переключении обратно указатель AP_OPNR надо записать в 0, и дождаться, пока регулятор позиции возвращает состояние AN_OPNA=0. В сторону привода выходит сигнал команды 0.

Битом #5 FUP параметра N0514 Servo Control и указателем AP_FLWU вместе определяется, что включение замыкания контура происходит ли сервослежением, или без него. (См.: Описание сигнала AP_FLWU)

☞ ***Внимание!*** *Размыкание контура регулирования позиции, а потом замыкание обратно надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров программы)*

AP_END: Запрет наблюдения за датчиком

Если указатель включить в 1, сервоконтур не ведёт наблюдение за ошибки датчика (Не даст Сообщение об ошибке датчика).

AP_FLWU: Включение сервослежения

При включении контура регулирования позиции он регулирует способ замыкания контура:

Если

параметр N0514 Servo Control #5 FUP=0 **И** указатель PLC AP_FLWU=0

позиция команды примет значение абсолютной (измеренную от датчика) позиции до замыкания контура регулирования позиции и на оси ***перемещения не происходит***. Станок остаётся в сдвинутой позиции до тех пор, пока не запрограммировать абсолютное движение.

Если

параметр N0514 Servo Control #5 FUP=1 **ИЛИ** указатель PLC AP_FLWU=1

позиция команды примет значение абсолютной (измеренную от датчика) позиции до замыкания контура регулирования позиции и ***на оси происходит перемещение***, выполняется шагами перемещение, накопившееся во выключенном состоянии серво.

AP_JOGP: Запрос Jog в + направлении

AP_JOGN: Запрос Jog в - направлении

Если в режиме Jog, или jog с приращением включить указатель в 1, ось перемещается в соответствующее (+/-) направление, если включить указатель в 0, ось остановится.

К этим сигналам можно прикрепить клавиши MB_JOGn.

Если в режиме приёма референтной точки, включить любой указатель AP_JOGP, или AP_JOGN в 1, ось будет перемещаться не в направление jog, а в направление, определённое параметрами N0901 Reference Distance и N0900 Reference Type #5 DIR до тех пор, пока любой из указателей стоит в 1, остановится, если любой из указателей не в 1, или произошёл приём референтной точки.

AP_DECSW: Замедлитель на включателе

Если на оси установлено положение параметра N0900 Reference Type #0 SWT=1, приём референтной точки происходит с набегом на включатель.

Если значение указателя AP_DECSW 1, ось совершил набег на включатель референтной точки.

AP_RAPD: Запрет движения быстрого хода

Это есть ответный сигнал на указатель запроса движения быстрого хода AN_RAPR.

Если AN_RAPR=1, движение быстрого хода запускается тогда, если PLC установил сигнал AP_RAPD=0.

Если AP_RAPD=1 движение быстрого хода не идёт.

Указателем AP_MTNDP, AP_MTNDN надо обращаться вместе с указателями запрета движения.

Обращение этим указателем можно использовать например для изменения передачи между движением подачи и быстрого хода.

AP_MTNDP: Запрет движения в положительном направлении
AP_MTNDN: Запрет движения в отрицательном направлении

Это есть ответные сигналы на указатель запроса движения AN_MTNRP и AN_MTNRN.

Перед запуском каждого движения интерполятор включает в 1 указатель запроса движения соответствующего направления (AN_MTNRP=1, AN_MTNDN=1). В случае круговой интерполяции на осях в плоскости круга, в оба направления запросит движение.

Ось не сдвинется до тех пор, пока PLC не разрешает движение в соответствующее направление, при положении указателя AP_MTNDP=0, или AP_MTNDN=0.

Ось остановится, если PLC включает указатель запрета движения соответствующего направления в 1.

См. ещё указатели AN_RAPR, AP_RAPD, AP_FEEDD.

Указатели можно использовать например для фиксирования, разжима осей.

AP_LIMP: На включателе положительного конечнего положения
AP_LIMN: На включателе отрицательного конечнего положения

Если ось в каком-то направлении набегал на включатель конечнего положения, соответствующий указатель надо записать в 1: AP_LIMP=1, или AP_LIMN=1. После установки указателя интерполятор остановится с замедлением и запускать снова можно только в другое направление. Ход замедления, необходимый для останова, надо принимать во внимание при установке включателя.

AP_LIMSELP: Выбор диапазона 1B положительного конечнего положения с параметром
AP_LIMSELN: Выбор диапазона 1B отрицательного конечнего положения с параметром

Выбор конечнего положения с параметром по осям и по направлениям из PLC.

Если

параметр N1001 StrkCont #4 ABA=0

В данном канале действительны указатели PLC, позволяющие выбор параметров по осям и по направлениям: AP_LIMSELP в положительном направлении, AP_LIMSELN в отрицательном направлении, может выбирать между диапазонами 1A и 1B.

В положительном направлении выбирается

параметр AP_LIMSELP=0: N1002 Range1A Positive,
параметр AP_LIMSELP=1: N1004 Range1B Positive.

В отрицательном направлении выбирается

параметр AP_LIMSELN=0: N1003 Range1A Negative,
параметр AP_LIMSELN=1: N1005 Range1B Negative.

Изменение выбора конечнего положения надо выполнить в стоячем состоянии осей.

☞ ***См. ещё:*** *указатель CP_LIMSEL.*

AP_LCK: Ось закрыта: все движения запрещены

В состоянии AP_LCK=1 интерполятором не выдаётся команда движения в сторону оси. Движение видно на индикаторе позиции..

При переходе в AP_LCK=0, интерполятором обновляются позиции из измерительной системы и ось можно снова перемещать..

Изменить указатель разрешено только тогда, если не идёт прогон программы деиалей и ось стоит!

AP_DISPD: Индикация оси выключена

Если у какой-то оси параметр N0002 Axis Assign не 0 и параметрами N0100 Axis Name1, ... присвоили ей название, ось автоматически появится на индикаторе позиции.

Если указатель AP_DISPD=1, данная ось изчезает с индикатора позиции.

Например, если какую-то ось выключить указателем AP_DETCHR, то можно выключить и её индикацию..

AP_PARKR: Запрос Ось паркует

Парковку можно запросить только для оси, учавствующей в синхронном движении, то есть ось является то ли

AN_SYNCA=1:синхронным-аппликатором, то ли

AN_SYNCM=1: синхронным-мастером.

Парковку инициирует желающая парковать ось, с записыванием принадлежащего к ней указателя запроса парковки AP_PARKR. Парковка установится тогда, если оператор оси записывает в 1 сигнал подтверждения AN_PARKA. При выключения парковки сигнал AP_PARKR надо записать в 0, и дождаться, чтобы оператор оси вернул состояние AN_PARKA=0.

Парковкой является, когда программа деталей написана с использованием синхронной работы, но то ли синхронную ось-аппликатор, то ли синхронную мастер-ось не желаем перемещать потому, что например, на той стороне нет заготовки. При этом то ли включателем, то ли функцией M можно включить парковку на соответствующей оси. См. ещё сигнал AP_SYNCR.

☞ ***Внимание!*** *Включение, затем выключение парковки надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров программы). Сигнал подтверждения не приходит до тех пор, пока буфер не пустой. Задействовать парковку разрешено только тогда, когда нет прогона программы и ось находится в покое.*

AP_SYNCR: Запрос синхронизации оси

Синхронное перемещение может инициировать синхронная ось-аппликат с записыванием принадлежащего к ней указателя запроса синхронизации AP_SYNCR. Синхронная связь создаётся тогда, если оператор оси записывает сигнал подтверждения AN_SYNCA в 1. При выключении синхронного перемещения, сигнал AP_SYNCR надо записать в 0, и дождаться, чтобы оператор оси вернул состояние AN_SYNCA=0.

Синхронной обработкой называется, когда одна ось, синхронная ось-аппликатор перемещается под действием команды движения другой оси, синхронной мастера-оси. В этом состоянии нельзя выдавать команду движения на синхронную ось-аппликатор. Движение синхронной оси можно включить или выключить из программы деталей, например под действием функции M.

Номер оси мастера синхронной оси-аппликата надо записать в параметре N2101 Synchronous Master в номер оси, принадлежащего к синхронной оси-аппликату. Если значение параметра 0, ось не является синхронной осью-аппликатом. Например, если 4-ая ось является синхронной осью-аппликатом и номер её мастера-оси 2, параметр надо заполнять следующим образом:

N2101 Synchronous Master A4=2

Направление перемещения синхронной оси-аппликата по сравнению к мастеру-оси определяется битом #0 MSY параметра N2102 Synchron Config paraméter #0 MSY

☞ ***Внимание!*** *Синхронное перемещение оси нельзя перепутать с синхроном позиции, когда например на станке, имеющей станину (gantry) два двигателя приводит в движение одну и ту же ось. При этом параметр N2102 Synchron Config будет #4 PSN=1, а при синхронном перемещении PSN=0.*

☞ ***Внимание!*** *Включение, затем выключение синхронной обработки надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров программы). Сигнал подтверждения не приходит до тех пор, пока буфер не пустой.*

AP_MIXR: Запрос смены оси

Смену оси инициирует сменяемая ось с записыванием принадлежащего к ней указателя запроса смены AP_MIXR. Смена происходит тогда, если оператор оси записал в 1 сигнал подтверждения AN_MIXA. При выключении смены оси сигнал AP_MIXR надо записать 0, и дождаться, чтобы оператор оси вернул состояние AN_MIXA=0.

В случае смены оси по адресу сменяемой оси (N0100 Axis Name1, ...определённые параметром адреса) можно перемещать ось с другим номером (сменную ось) и наоборот, по адресу сменной оси перемещается сменяемая ось. Смену оси можно включить или выключить из программы деталей, например под действием функции M. *Смена оси относится только к ссылке по адресу оси из программы деталей, к перемещению jog и маховичка нет, потому что эти относятся к номерам осей, поэтому этими случаями надо обращаться из программы PLC!*

Параметр N2104 Composit Axis, относящийся к сменяемой оси подскажет, что сменяемую ось с какой по номеру осью надо заменить. Если значение параметра 0, ось не сменяемая. Например, если 4-я ось является сменяемой осью и номер сменной оси 1, параметр надо заполнять следующим образом:

N2104 Composit Axis A4=1

При смене оси обе оси могут работать и в другом канале (не в том, к которому параметр Axis Assign выделил), оттуда получает команды движения и из другого канала получает смещение нулевой точки первоначальной оси. После смены в обоих каналах можно перемещать оси, кроме случая, если одна из осей является гипотетичной. Если ось выделена гипотетичной в данном канале (N0106 Axis Properties #7 HYP=1), после смены нельзя на неё ссылаться в данном канале, или в том же канале.

Параметром N2105 Composit Config #0 MMI определяется, что сменяемая ось, далее сменная ось перемещались в том же направлении по сравнению к первоначальному, или наоборот.

☞ ***Внимание!*** *Включение, затем выключение смены оси надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров программы). Сигнал подтверждения не приходит до тех пор, пока буфер не пустой.*

AP_SPRPNR: Запрос совмещения на ось

Совмещённое перемещение инициирует всегда ось-аппликат совмещения с записыванием указателя запроса совмещения AP_SPRPNR. Связь наложения создаётся тогда, если оператор оси записывает в 1 сигнал подтверждения AN_SPRPNA. При выключении совмещённого перемещения в сигнал AP_SPRPNR надо записать 0, и дождаться, чтобы оператор оси вернул состояние AN_SPRPNA=0.

Обработкой с наложением называется, когда ось (ось-аппликат совмещения) перемещается под действием двух команд движения: сдвиги, полученные по собственному адресу и сдвиги, полученные от оси-мастера совмещения складываются и так формируется движение. Ось-аппликат и ось-мастер могут быть в том же канале, но и в разных. Перемещение оси совмещения можно включить, или выключить из программы деталей, например под действием функции M.

Номер оси-мастера оси-аппликата совмещения записывается в номер оси, относящийся к оси-аппликата совмещения в параметре N2107 Superimposed Master. Если значение параметра 0, ось не является осью-аппликатом совмещения. Например, если 4-я ось является осью-аппликатом совмещения и номер оси-мастера 2, параметр надо заполнять следующим образом:

N2107 Superimposed Master A4=2

Направление смещения оси-аппликата совмещения по сравнению к оси-мастера определяется битом #0 MSU параметра N2108 Superimposed Config.

☞ ***Внимание!*** *Включение, затем выключение движения наложения надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров*

программы). Сигнал подтверждения не приходит до тех пор, пока буфер не пустой.

AP_DIARAD: Ось запрограммирована по диаметру

Состояние указателя принимается во внимение системой тогда, если значение бита #5 MGD параметра N0106 Axis Properties равно =0.

Он регулирует способ ввода данных и индикацию позиции на данной оси:

Если

параметр N0106 Axis Properties #0 DIA=0 **И** указатель PLC AP_DIARAD=0

ввод данных и индикация позиции на данной оси выполняется **по радиусу**.

Если

параметр N0106 Axis Properties #0 DIA=1 **ИЛИ** указатель PLC AP_DIARAD=1

ввод данных и индикация позиции на данной оси выполняется **по** диаметру.

Ввод данных распостраняется на программу деталей, на коррекции и на смещения нулевой точки.

☞ ***Внимание!*** *Переключение из радиуса в диаметр, или из диаметра в радиус надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров программы).*

AP_SSLOP: Размыкание контура оси-аппликата и приём только основного сигнала

Если ось **не является синхронной осью-аппликата**, его действие совпадает действием сигнала AP_OPNR.

Если ось **является синхронной осью-аппликата**, то есть на оси AN_SYNCA=1, он размыкает контур регулирования позиции, ось-аппликат берёт на себя сигнал команды скорости от оси-мастера и отправит его для привода оси-аппликата.

Пример:

Шпиндель и контршпиндель за одно является и осью Z, то есть оба можно перемещать. При желании обработать длинную ось, контршпинделем надо подпирать заготовку. Ось Z контршпинделя надо выделить синхронной осью-аппликата Z шпинделя и соединить два движения (AP_SYNCR=1). После этого, каждый раз, когда патрон будет зажат и на шпинделе и на контршпинделе, для избежания сверхопределимости, установкой AP_SSLOP=1 надо разомкнуть контур регулирования позиции на оси-аппликате. Ось Z шпинделя можно перемещать, ось-аппликат берёт на себя сигнал команды скорости от оси-мастера перемещаются вместе. См. ещё сигнал привода DP_SILCK.

AP_FEEDD: Запрет движения подачи

Это есть ответный сигнал для указателя запроса движения быстрого хода AN_RAPR.

Если AN_RAPR=0, движение подачи запускается только тогда, если PLC установил сигнал AP_FEEDD=0.

Если AP_FEEDD=1, движение подачи не идёт.

Указателем надо обращаться вместе с указателями запрета движения AP_MTNDP, AP_MTNDN.

Обращение этим указателем используется при изменении передачи между движениями подачи и быстрого хода.

AP_PLCR: PLC запросит ось

После включения любой оси, заданной параметром N0002 Axis Assign, обращение выполняет NC. PLC может запросить любую ось от NC и может выполнить перемещение согласно описанию 6.15 Команда по перемещению оси: в главе MOVCMD. Ось может запросить PLC с записыванием указателя AP_PLCR в 1. Всегда надо дождаться, чтобы оператор оси вернул состояние AN_PLCA=1. И наоборот, когда PLC возвращает ось, запишет указатель AP_PLCR в 0, и надо дождаться состояния AN_PLCA=0.

☞ ***Внимание!*** *Запрос оси через PLC и её возвращение надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров программы). Сигнал подтверждения не приходит до тех пор, пока буфер не пустой.*

AP_GOR: Пусть идёт движение по оси PLC

Если интерполятор оси PLC не пустой, то есть AN_IEPTY=0, по состоянию указателя AP_GOR=1 запускается движение, а под действием AP_GOR=0 остановится. О том, что идёт ли движение, даст информацию указатель AN_GOA. См. указатель AN_GOA.

AP_RES: Ось PLC reset

Если сигнал **AP_RES** переходит в 1:

ось PLC остановится с замедлением,

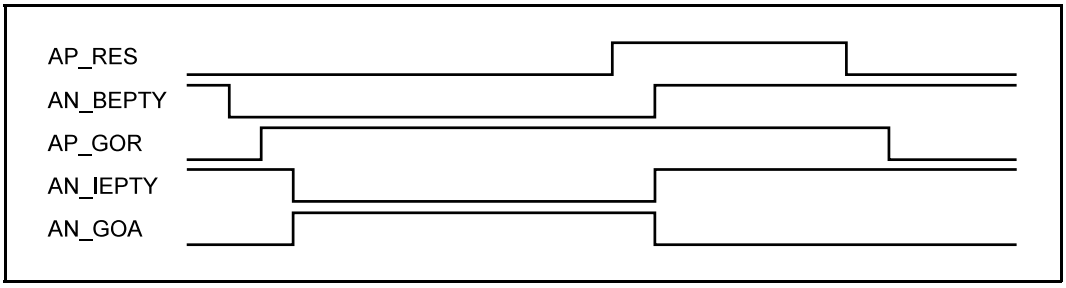
удаляет актуальную команду, находящуюся под выполнением,

удаляет из буфера команду, и установит следующие указатели:

AN_BEPTY=1

AN_GOA=0

AN_IEPTY=1



AP_EFD: Запрет измерения хода

Указатель действителен только в открытом состоянии контура регулирования позиции (AN_OPNA=1).

Если значение указателя 1, оператор оси не обновляет позицию оси об импульсах датчика, блокирует их значение до тех пор, пока указатель не переходит снова на 0. Начиная с этого отсчитываются импульсы, поступившие с датчика от блокированной позиции. Этой функцией можно пользоваться тогда, если несколько осей пользуется тем же датчиком для измерения хода.

Указатель PLC находится в связи с битом #4 EFD параметра N0514 Servo Control.

AP_RPE: Имеется действующая референтная точка

Если регулирование позиции выпоняется не NC, а приводом, тогда и приём референтной точки выполняется приводом. Наличие референтной точки сообщается программой PLC после загрузки контрольного бита привода, включением сигнала AP_RPE для NC.

Этот случай задаваётся установкой #3 ABS=1 и #4 FLO=1 в параметре N0900 Reference Type.

AP_MIRR: Запрос смены направления оси (отражения)

Программа PLC через индикатор AP_MIRR может запросить смену направления на одной из осей. Сигнал AN_MIRA является подтверждением запроса, которого следует всегда дождаться.

При этом в случае перемещения осей из программы деталей направление перемещения данной оси переключается в противоположное.

Направление ручного перемещения (клавиши jog, маховичок) не меняется.

Если значение сигнала:

=0: ось перемещается в первоначальное направление, установленное параметром из программы деталей,

=1: ось перемещается в направление, противоположное установленному параметром из программы деталей.

☞ ***Внимание!*** *Смену направления оси следует выполнить всегда в функции, предотвращающей буферизации кадра (смотри Группу параметров Program).*

7.14 Переменные оператора для шпинделя

Переменными оператора шпинделя являются переменные, идущие от NC в PLC, или из PLC в NC, которые индексируются по шпинделям. Все приведенные здесь символы относятся к первому (с индексом 0) шпинделю. Соответствующие переменные остальных шпинделей доступны с индексированной адресовкой. Управление обращается не более 16 шпинделями.

Переменные

начинающиеся с SN идут от NC в PLC (Вводы), а

начинающиеся с SP от PLC в NC (Выводы).

7.14.1 Бинарные переменные для шпинделя

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
SN_RMPD	Основной сигнал набегал на шпинделе	**SP_SEN**	Выдача сигнала на шпинделе разрешена
SN_NS	Число оборотов, измеренное с датчика шпинделя, достигло команду чисел оборотов N=Ns	**SP_SSTRT**	Старт шпинделя
SN_N0	Число оборотов, измеренное с датчика шпинделя 0 (N=0)	**SP_PAR**	Выдача основного сигнала шпинделя с параметра
SN_FLU	Число оборотов шпинделя колеблится	**SP_NEG**	Направление вращения M4 шпинделя (отрицательный основной сигнал)
SN_FLOFF	Наблюдение за колебанием чисел оборотов шпинделя выключено	**SP_OREQ**	Запрос ориентировки на шпинделе
SN_LPCLSD	Контур регулирования позиции на шпинделе замкнут.	**SP_OSHRT**	Ориентировка шпинделя по кратчайшему пути (если =0: ориентирует по знаку NEG)
SN_ORIP	Контур регулирования позиции на шпинделе замкнут, и в позиции ориентировки	**SP_SSYNCR**	Запрос синхронизации шпинделя
SN_SINPOS	Шпиндель в позиции	**SP_PHSHFTR**	Запрос сдвига по фазе при синхронизации
SN_SSYNA	Шпиндель-аппликат синхронизирован	**SP_POLYR**	Не используется
SN_PHSHFTA	Сдвиг по фазе произошёл на шпинделе-аппликате	**SP_SEND**	Запрет наблюдения за датчиком на шпинделе
SN_SYNCPOS	Не используется	**SP_SMTNDP**	Запрет движения шпинделя в положительном направлении
SN_POLYA	Синхронизация на шпинделе-аппликате под точение многоу-	**SP_SMTNDN**	Запрет движения шпинделя в отрицательном направлении

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
	гольника произошла		
SN_SMTNRP	Запрос движения шпинделя в положительном направлении	**SP_SDETCHR**	Запрос выключения оператора шпинделя
SN_SMTNRN	Запрос движения шпинделя в отрицательном направлении	**SP_SLCLR**	Запрос замыкания контура без ориентировки
SN_SRAPR	Запрос быстрого хода	**SP_SSROFF**	Размыкание контура позиции на шпинделе-аппликате
SN_SDETCHA	Выключение оператора шпинделя подтверждено	**SP_SDISPD**	Запрет индикации шпинделя на экране
SN_SALM	Ошибка на шпинделе	**SP_FEEDD**	Запрет движения подачи шпинделя
SN_RPE	Имеется действительная точка референции на шпинделе	**SP_RAPD**	Запрет движения быстрого хода на шпинделе
SN_SINDP	Шпиндель в позиции индекса	**SP_TLCHIA**	Сигнал подтверждения запроса смены инструмента
SN_TLCHI	Стойкость инструмента в шпинделе закончилась, запрос смены инструмента	**SP_TLCHA**	Сигнал подтверждения запроса смены группы
SN_TLCH	Стойкость полной группы инструментов закончилась, запрос смены группы	**SP_TLSKP**	Инструмент в шпинделе сломан
SN_TLSKPA	Подтверждение заметки поломки инструмента	**SP_TLCD**	Запрет счётчика стойкости
SN_TLNL	Предупреждающая стойкость инструмента в шпинделе закончилась	**SP_OSW**	Шпиндель находится на включателе ориентировки

Бинарные переменные шпинделя, идущие от NC в PLC

SN_RMPD: Основной сигнал набегал на щпинделе

Если надо увеличивать число оборотов в абсолютном значении, на основании параметра N0665+n Rn S Ramp Up оператор шпинделя вызывает набег основного сигнала, выданного приводу, а если надо уменьшить число оборотов в абсолютном значении, на основании параметра N0673+n Rn S Ramp Down вызывает его сбег, где “n”-номер загруженного диапазона.

Как только оператор шпинделя закончил вызывание набега/сбега, установит указатель SN_RMPD в 1.

SN_NS: Число оборотов шпинделя, измеренное с датчика, достигло команду чисел оборотов N=Ns

Если число оборотов шпинделя, измеренное с датчика достигло выданную команду чисел оборотов в пределах допуска, определённого параметрами N0627 S N% и N0628 S NW, оператор шпинделя установит указатель SN_NS в 1.
Если число оборотов в абсолютном значении пригодное, но сигнал SN_NS не приходит, в параметре N0609 Spindle Encoder Config надо изменить направление датчика (в параметре #1 ID, или #2 AD), предпологая, что шпиндель не униполярный: #5 UNI=0.

SN_N0: Число оборотов, измеренное с датчика шпинделя (N=0)
Если число оборотов, измеренное с датчика шпинделя достигло значеине 0 в пределах допуска, определённого параметром N0629 S N0, оператор шпинделя установит в 1 указатель SN_N0 (и указатель SN_NS).

SN_FLU: Число оборотов шпинделя колеблится
Если число оборотов, измеренное с датчика шпинделя выходит из пределов допуска, определённого параметрами N0632 S Fluct% и N0633 S FluctW, оператор шпинделя установит в 1 указатель SN_FLU.

SN_FLOFF: Наблюдение за колебанием чисел оборотов шпинделя выключено
Если из программы деталей командой G25 выключить наблюдение за колебанием чисел оборотов шпинделя, указатель SN_FLOFF примет состояние..

SN_LPCLSD: Контур регулирования позиции на шпинделе замкнут.
Если на шпинлделе контур регулирования позиции замкнут, оператор шпинделя установит указатель в 1. Эти случаи являются следующими:
– после выполнения команды ориентировки SP_OREQ, или замыкания позиции SP_SLCLR,
– если шпиндель после синхронизации двух шпинделей то ли синхронный шпиндель-мастер, то ли синхронный шпиндель-аппликат,
– если шпиндель является то ли шпинделем-мастером, то ли шпинделем-аппликатом точения многоугольника G51.2,
– если шпиндель после включения электронного привода G81.8 является шпинделем-мастером (фрезерование зубчатых колёс способом обкатки).

SN_ORIP: Контур регулирования позиции на шпинделе замкнут, и в позиции ориентировки
Оператор шпинделя установит указатель в 1, если на шпинделе контур регулирования позиции замкнут и шпиндель находится в окресности, заданной параметром N0743+n Rn S Inpos заданного параметром N0684 Spindle Grid Shift расстояния, отсчитывая от нулевого импульса датчика.
Если N0684 Spindle Grid Shift=0, в окресности, заданной параметром N0743+n Rn S Inpos нулевого импульса.

SN_SINPOS: Шпиндель в позиции

Оператор шпинделя установит указатель в 1,

– после выполнения команды ориентировки SP_OREQ или замыкания позиции SP_SLCLR, если абсолютное значение разности между позицией команды, выданной для шпинделя и позицией, измеренной с датчика меньше значения, заданного параметром N0743+n Rn S Inpos.

– если *шпиндель* после синхронизации двух шпинделей является *синхронным шпинделем-аппликатом*, или

– если *шпиндель* является *шпинделем-аппликатом точения многоугольника* G51.2, и абсолютное значение цикличной разности между позициями шпинделя-мастера и шпинделя-аппликата (разность позиции их нулевых импульсов, учитывая и смещения) меньше значения, заданного параметром N0743+n Rn S Inpos.

SN_SSYNA: Шпиндель-аппликат синхронизирован

Это есть указателем подтверждения запроса синхронизации SP_SSYNCR.

Если на шпинделе PLC запросит синхронизацию командой SP_SSYNCR к другому щпинделю, и синхронизация выполнилась, оператор шпинделя установит указатель в 1.

SN_PHSHFTA: Сдвиг по фазе произошёл на шпинделе-аппликате

Это есть указателем подтверждения запроса сдвига по фазе SP_PHSHFTR.

Если PLC запросит синхронизацию со сдвигом по фазе при условии SP_PHSHFTR=1, и синхронизация и сдвиг по фазе выполнились,оператор шпинделя установит указатель SN_PHSHFTA в 1.

SN_SYNCPOS: Не испоьлзуется

SN_POLYA: Синхронизация на шпинделе-аппликате под точение многоугольника произошла

Если шпиндель является шпинделем-аппликатом под точение многоугольника G51.2, и синхронизация к шпинделю-мастеру выполнилась, оператор шпинделя установит указатель SN_POLYA в 1.

SN_SMTNRP: Запрос движения шпинделя в положительное направление

SN_SMTNRN: Запрос движения шпинделя в отрицательное направление

В случае индексирования шпинделя, перед запуском каждого движения интерполятор переключает указатель запроса движения в соответствующее направление в 1.

Если PLC разрешает движение в соответствующее направление, он должен установить указатели SP_SMTNDP=0, или SP_SMTNDN=0.

См. ещё указатели SN_SRAPR, SP_RAPD, SP_FEEDD.

Указатели можно использовать например для фиксирования, отжима шпинделей.

SN_SRAPR: Запрос быстрого хода

Интерполятор обращается этим указателем вместе указателями запроса движения SN_SMTNRP, SN_SMTNRN.

Если SN_SRAPR=0, интерполятор хочет перемещаться с подачей, ожидает состояние и SP_FEEDD=0 (движение подачи разрешено) и SP_RAPD=1 (движение быстрого хода запрещено).

Если SN_SRAPR=1, интерполятор хочет перемещаться быстым ходом, ожидает состояние и SP_FEEDD=1 движение подачи запрещено) и SP_RAPD=0 (движение быстрого хода разрешено).

См. ещё указатели SN_SMTNRP, SN_SMTNRN, SP_RAPD, SP_FEEDD.

Обращение этим указателем можно использовать например для изменения передачи между движениями движения подачи и быстрого хода.

SN_SDETCHA: Подтверждение выключения оператора шпинделем

SN_SDETCHA есть указателем запроса выключения шпинделя SP_SDETCHR.

На запрос выключения шпинделя SP_SDETCHR=1 оператор шпинделя возвращает SN_SDETCHA=1. На запрос включения SP_SDETCHR=0 оператор шпинделя возвращает SN_SDETCHA=0. См.: Описание указателя SP_SDETCHR.

SN_SALM: Ошибка на шпинделе

Если оператор шпинделя обнаруживает ошибку на любом элементе сервоконтура, (датчик, привод, работающий через EtherCAT, контур регулирования позиции) в замкнутом состоянии контура позиции шпинделя, то указатель SN_SALM установит в 1.

☞ ***Внимание!*** *Задачей программы PLC является реагировать на сообщения об ошибке, остановить привод, вызывать аварийное состояние! NC не выключает станок автоматически, только по команде PLC!*

SN_RPE: Имеется действительная точка референции на шпинделе

В состоянии SN_RPE=1 на шпинделе уже произошёл приём референтной точки (положение нулевого импульса известное). В случае абсолютных датчиков указатель всегда 1.

SN_SINDP: Шпиндель в позиции индекса

Значение указателя 1, если позиция шпинделя находится внутри окружности с радиусом, определённым параметром N0743+n Rn S Inpos позиции индекса.

Позицией индекса называются на шпинделе выделённые с какой-то точки зрении позиции, расположенные на дискретное, равное друг от друга расстояние. Например: если для фиксирования шпинделя можно задвигать стержень в диск, прикреплённый к шпинделю по 5 градусам, то шпиндель является индксироваемым по 5 градусам.

Параметром N0822 Basic Angle of Spnd. Pos. можно установить расстояние индексирования. Например: в случае фиксирования указанного выше шпинделя 5.

SN_TLCHI: Стойкость инструмента в шпинделе закончилась, запрос смены инструмента
NC переключает указатель в 1, если параметр N2900 Tool M. Config #0, TMU=1 и у инструмента в данном шпинделе
– закончилась стойкость, или
– PLC включает для него указатель SP_TLSKP: сломан.
Указатель SN_TLCHI остаётся включенным до тех пор, пока PLC не включает сигнал подтверждения SP_TLCHIA. После того, что NC удалил сигнал SN_TLCHI, и PLC должен удалить сигнал подтверждения.
Указатель SN_TLCHI на reset *не удаляется*.

SN_TLCH: Стойкость полной группы инструментов закончилась, запрос смены инструмента
NC переключает указатель в 1, если параметр N2900 Tool M. Config #0, TMU=1 и для полной группы инструментов (инструменты с одинаковым кодом типа) то правда, что
– стойкость“Закончилась”, или
– “Сломан”.
Указатель SN_TLCH остаётся включенным до тех пор, пока PLC не включает сигнал подтверждения SP_TLCHA. После того, что NC удалил сигнал SN_TLCH, и PLC должен удалить сигнал подтверждения.
Указатель SN_TLCH на reset *не удаляется*.

SN_TLSKPA: Подтверждение заметки поломки и нструмента
Если PLC обнаруживает поломку инструмента и запишет указатель SP_TLSKP в 1, NC в таблице регистрирует для инструмента состояние “сломан”, затем запишет сигнал подтверждения SN_TLSKPA в 1, если параметр N2900 Tool M. Config будет #0 TMU=1.
После этого выдаётся указатель запроса смены инструмента SN_TLCHI.

SN_TLNL: Предупреждающая стойкость инструмента в шпинделе закончилась
Если закончилась предупреждающая стойкость инструмента в шпинделе, NC до начала 1 цикла PLC запишет указатель в 1, если параметр N2900 Tool M. Config будет #0 TMU=1.

Бинарные переменные, идущие от PLC в NC

SP_SEN: Выдача сигнала на шпинделе разрешена

Каждый раз, когда PLC выдаёт какую-то команду для шпинделя (вращение, ориентировка, синхронизация, точение многоугольника), с записыванием указателя SP_SEN в 1 должен разрешать выдачу сигнала.

После останова шпинделя указатель надо выключить.

SP_SSTRT: Старт шпинделя

Под действием ***SP_SSTRT=1*** оператор шпинделя вращает шпиндель с установленным числом оборотов.

Под действием ***SP_SSTRT=0*** оператор шпинделя остановит вращение шпинделя.

См. ещё указатели SP_PAR, SP_NEG и регистры SP_PRG.

SP_PAR: Выдача основного сигнала шпинделя с параметра

В состоянии ***SP_PAR=0***, под действием SP_SSTRT=1 оператор шпинделя вращает шпиндель с числом оборотов, заданным в регистре SP_PRG.

В состоянии ***SP_PAR=1***, под действием SP_SSTRT=1 оператор шпинделя вращает шпиндель с числом оборотов, установленным параметром N0657+n Rn S Jog Speed.

SP_NEG: Направление вращения M4 шпинделя (отрицательный основной сигнал)

В состоянии ***SP_NEG=0***, под действием SP_SSTRT=1 оператор шпинделя вращает шпиндель в положительном, M3 направлении.

В состоянии ***SP_NEG=1***, под действием SP_SSTRT=1 оператор шпинделя вращает шпиндель в отрицательном, M4 направлении.

Если направление вращения шпинделя не правильное, в параметре N0609 Spindle Encoder Config надо изменить направление вращения двигателя параметром #0 MD. (Изменить направление вращения не с инвертированием SP_NEG.)

В состоянии ***SP_NEG=0***, под действием SP_SSYNCR=1 запроса синхронизации оператор шпинделя вращает шпиндель в направлении, согласно направлению вращения шпинделя, заданному в регистре SP_MAST.

В состоянии ***SP_NEG=1***, под действием SP_SSYNCR=1 запроса синхронизации оператор шпинделя вращает шпиндель в направлении, противоположно направлению вращения шпинделя, заданному в регистре SP_MAST. (Синхронизация контршпинделя.)

SP_OREQ: Запрос ориентировки на шпинделе

Под действием ***SP_OREQ=1*** оператор шпинделя

– если шпиндель вращается,

замедляет вращение шпинделя до числа оборотов, заданного параметром N0800+n Rn S OrientSpeed,

замыкает контур регулирования позиции (SN_LPCLSD=1),

затем становится в позицию, заданную параметром N0684 Spindle Grid Shift, отсчитанную от нулевого импульса (на нулевой импульс, если параметр 0) включает указатель SN_ORIP.

– если шпиндель стоит и SP_OSHRT=0,
в направлении, определённом указателем SP_NEG, со скоростью, заданной параметром N0800+n Rn S OrientSpeed, становится в позицию, заданную параметром N0684 Spindle Grid Shift, отсчитанную от нулевого импульса (на нулевой импульс, если параметр 0)
включает указатель SN_ORIP.

– если шпиндель стоит SP_OSHRT=1, и положение нулевого импульса уже известное, далее параметр N0607 Spindle Config будет #4 ZOR=1
в ближайшем направлении, со скоростью, заданной параметром N0800+n Rn S OrientSpeed, становится в позицию, заданную параметром N0684 Spindle Grid Shift, отсчитанную от нулевого импульса (на нулевой импульс, если параметр 0)
включает указатель SN_ORIP.

Под действием ***SP_OREQ=0*** оператор шпинделя размыкает контур регулирования позиции и выключает указатель SN_LPCLSD.

SP_OSHRT: Ориентировка шпинделя по кратчайшему пути (если =0: ориентрует согласно знаку NEG)

Если шпиндель стоит и SP_OSHRT=0, ориентрует в направлении, определённом указателем SP_NEG.

Если шпиндель стоит, SP_OSHRT=1, и положение нулевого импульса уже известное, далее параметр N0607 Spindle Config будет #4 ZOR=1, ориентрует по кратчайшему пути.

SP_SSYNCR: Запрос синхронизации шпинделя

Указатель служит для синхронизации двух шпинделей. Синхронизация означает совместный бег нулевого импульса двух шпинделей согласно правилу позиции в состоянии вращения. Эти два нулевых импульсов могут совместно бежать, или бежать на расстояние друг от друга, установленного параметром N0685 Spindle Phase Shift. Усиление синхронного регулирования можно регулировать параметром N0784+n Rn S Synchr K. После останова шпинделя-мастера также шпиндель-аппликат следует за шпинделем-мастером. Синхронизацию к шпинделю-мастеру запрашивает всегда шпиндель-аппликат.

Синхронизацию можно использовать например тогда, когда на станке с контршпинделем в состоянии вращения надо перенимать заготовку контршпинделем, как синхронным шпинделем-аппликатом. Правила позиции означает, что на одной стороне по сравнению к нулевому импульсу шпинделя частично обработали заготовку, а в контршпинделе на другой стороне можно выполнить обработку независимо от этого.

Перед запросом синхронизации в регистр шпинделя-аппликата SP_MAST надо записать индекс шпинделя-мастера.

При желании отодвинуть два нулевых импульсов друг от друга на значение, установленное параметром N0685 Spindle Phase Shift, перед запросом синхронизации надо установить указатель SP_PHSHFTR.

Под действием ***SP_SSYNCR=1*** оператор шпинделя

– если число оборотов шпинделя-мастера больше значения, определённое параметром N0319+n Rn S Rapid n=1...8, замедляет его до значения, определённого параметром,

– замыкает контур регулирования позиции на шпинделе-мастере,

– замыкает контур регулирования позиции на шпинделе-аппликате,

– раскрутит шпиндель-аппликат до числа оборотов шпинделя-мастера по сравнению к шпинделю-мастеру в направление, определённое указателем SP_NEG,

– согласно положению указателя SP_PHSHFTR приведёт нулевые импульсы двух шпинделей на соответствующее расстояние,

– включает регулирование совместного бега, устанавливаемое параметром N0784+n Rn S Synchr K,

– включает на шпинделе-аппликате сигнал подтверждения SN_SSYNA и указатель SN_SINPOS, если абсолютное значение ошибки синхрона двух шпинделей меньше заданного параметром N0743+n Rn S значения.

Под действием ***SP_SSYNCR=0*** оператор шпинделя

– размыкает контур регулирования позиции на шпинделе-мастере,

– устанавливает число оборотов шпинделя-мастера, заданное в регистре SP_PRG, если он вращается,

– размыкает контур регулирования позиции на шпинделе-аппликате,

– выключает указатели SN_SSYNA и SN_SINPOS.

SP_PHSHFTR: Запрос сдига по фазе при синхронизации

В состоянии ***SP_PHSHFTR=1***, под действием запроса синхронизации ***SP_SSYNCR=1,*** устанавливает нулевые импульсы двух шпинделей на расстояние, устанавленное параметром N0685 Spindle Phase Shift, и включает на шпинделе-аппликате указатель подтверждения SN_SSYNA и указатель SN_SINPOS, если абсолютное значение ошибки синхрона двух шпинделей меньше заданного параметром N0743+n Rn S Inpos значения.

В состоянии ***SP_PHSHFTR=0***, ***SP_SSYNCR=1*** под действием запроса синхронизации синхронизирует друг к другу нулевые импульсы двух шпинделей и включает на шпинделе-аппликате сигнал подтверждения SN_SSYNA и указатель SN_SINPOS, если абсолютное значение ошибки синхрона двух шпинделей меньше заданного параметром N0743+n Rn S Inpos значения.

SP_POLYR: Не используется

SP_SEND: Запрет наблюдения за датчиком на шпинделе

Если указатель включить в 1, оператор шпинделя не наблюдает за ошибками оборудованного на шпиндель датчика (не выводит Сообщение об ошибке датчика).

SP_SMTNDP: Запрет движения шпинделя в положительном направлении

SP_SMTNDN: Запрет движения шпинделя в отрицательном направлении

Ответные сигналы, выданные на запрос движения SN_SMTNRP и SN_SMTNRN.

Перед каждым позиционированием шпинделя интерполятор (в состоянии SP_OREQ=1, или SP_SLCLR=1) переключает указатель запроса движения соответствующего направления в 1 (SN_SMTNRP=1, SN_SMTNDN=1).

Шпиндель не движется до тех пор, пока PLC не разрешает движение в соответствующем направлении, положением указателя SP_SMTNDP=0, или SP_SMTNDN=0.

Шпиндель остановится, если PLC переключает указатель запрета движения соответствующего направления в 1.

См. ещё указатели SN_SRAPR, SP_RAPD, SP_FEEDD.

Указатели можно использовать например для фиксирования, отжима шпинделей.

SP_SDETCHR: Запрос выключения оператора шпинделя

Если наобходимо выключить работу шпинделя, это можно запросить от системы записыванием укзателя SP_SDETCHR соответствующего шпинделя в 1. После того, что оператор шпинделя остановил работу шпинделя, возвращает сигнал подтверждения SN_SDETCHA=1. При включении обратно указатель SP_SDETCHR надо записать в 0, и дождаться, пока оператор шпинделя возвращает состояние SN_SDETCHA=0.

Если сигнал SN_SDETCHA находится в состоянии 1, оператор шпинделя не работает:

удаляет на шпинделе сообщение “имеется референтная точка”: SN_RPE=0,
не измеряет и не регистрирует позицию с датчика,
не замыкает контур регулирования позиции,
не выдаёт никакого сигнала в сторону привода шпинделя,
в программе деталей и в PLC нельзя ссылаться на шпиндель,

шпиндель не существует.

И наоборот, если SP_SDETCHR=0 и NC выключает указатель SN_SDETCHA=0, шпиндель снова существует.

Шпиндель существует тогда, если

на параметре N0607 Spindle Config значение #0 SEX будет 1, и
AN_DETCHA=0.

☞ ***Внимание!*** *Включение, затем выключение шпинделя надо выполнять всегда в функции, предотвращающей буферирования кадра.(см. Группа параметров программы)*

Пример:

Токарный шпиндель надо преобразовать в ось C (в поворотный стол), затем обратно в шпиндель. Пусть будет

номер оси C: 3 (N0002 Axis Assign A3=1, N0100 Axis Name1 A3=C) а
номер шпинделя S1: 1. (N0605 Spindle Name2 S1=1)

После включения он установится как шпиндель:

SP_SDETCHR,#0=0, SN_SDETCHA,#0=0
AP_DETCHR,#2=1, AN_DETCHA,#2=1

При желании переключить шпиндель S1 в ось C, надо запрограммировать соответствующую функцию M для подавления буфера и указатели переустановить следующим образом:

SP_SDETCHR,#0=1, SN_SDETCHA,#0=1
AP_DETCHR,#2=0, AN_DETCHA,#2=0

Начиная с этого, сигнали датчика с того же привода шпинделя принимает не оператор шпинделя S1, а оператор оси C. Основной сигнал будет выдавать не оператор шпинделя S1 для привода шпинделя, а оператор оси C. Ось C можно перемещать, на адрес C можно ссылаться в программе деталей.

Преобразование обратно оси C в шпиндель S1 происходит подобным образом.

SP_SLCLR: Запрос замыкания контура без ориентировки

Под действием ***SP_SLCLR=1*** оператор шпинделя

– если шпиндель вращается,

остановит вращение шпинделя,
замыкает контур регулирования позиции (SN_LPCLSD=1),

– если шпиндель стоит, замыкает контур регулирования позиции (SN_LPCLSD=1).

Под действием ***SP_SLCLR=0*** оператор шпинделя размыкает контур регулирования позиции (SN_LPCLSD=0).

Функцию можно использовать например для быстрого выполнения цикла сверления с жёстким стержнем, если не приходится ещё раз попасть в одно и то же отверстие. См. ещё: параметр N0823 M Code for Closing S Loop и параметр N1503 Drilling Cycles Config. будет #1 TSC бит.

SP_SSROFF: Размыкание контура позиции на шпинделе-аппликате

Если шпиндель является **синхронным шпинделем-аппликатом**, то есть на шпинделе SN_SSYNA=1, он размыкает контур регулирования позиции, шпиндель-аппликат перенимает на себя сигнал команды скорости шпинделя-мастера и отправит его для привода шпинделя-аппликата.

Пример:

Контршпиндель надо синхронизировать к шпинделю из-за перенимания заготовки, затем и патрон контршпинделя захватит заготовку. После этого, каждый раз, когда патрон будет зажат и на шпинделе и на контршпинделе, для избежания сверхопределимости, установкой SP_SSROFF=1 надо размыкать контур регулирования позиции на шпинделе-аппликате. При этом шпиндель-аппликат перенимает на себя от шпинделя-мастера сигналы команды скорости и вместе движутся. См. ещё сигнал привода DP_SILCK.

SP_SDISPD: Запрет индикации шпинделя на экране

Еслина шпинделе значение #0 SEX будет 1 на параметре N0607 Spindle Config, шпиндель автоматически появится на индикаторе FST.

Если указатель SP_SDISPD=1, данный шпиндель исчезнет из индикатора FST.

Например, если выключить шпиндель указателем SP_SDETCHR, можно выключить и его индикацию.

SP_FEEDD: Запрет движения подачи шпинделя

Ответный сигнал, выданные на запрос движения быстрого хода SN_SRAPR.

Если SN_SRAPR=0, движения подачи запускается тогда, если PLC установил сигнал SP_FEEDD=0.

Если SP_FEEDD=1, движения подачи не идёт.

Указателем надо обращаться вместе с указателями запрета движения SP_SMTNDP, SP_SMTNDN.

Обращение этим указателем можно использовать например для изменения передачи между движениями подачи и быстрого хода.

SP_RAPD: Запрет движения быстрого хода шпинделя

Ответный сигнал, выданные на запрос движения быстрого хода SN_RAPR.

Если SN_RAPR=1, движение быстрого хода запускается тогда, если PLC установил сигнал SP_RAPD=0.

Если SP_RAPD=1 движение быстрого хода не идёт.

Указателем надо обращаться вместе с указателями запрета движения SP_SMTNDP, SP_SMTNDN.

Обращение этим указателем можно использовать например для изменения передачи между движениями подачи и быстрого хода.

SP_TLCHIA: Сигнал подтверждения запроса смены инструмента

Указатель является сигналом подтверждения указателя SN_TLCHI.

Если PLC переключает сигнал SP_TLCHIA в 1, этим NC удаляет указатель SN_TLCHI. После того, что NC удалил сигнал SN_TLCHI, и PLC должен удалить сигнал подтверждения.

SP_TLCHA: Сигнал подтверждения запроса смены группы

Это есть сигналом подтверждения указателя SN_TLCH.

Если PLC переключает сигнал SP_TLCHA в 1, этим NC удаляет указатель SN_TLCH. После того, что NC удалил сигнал SN_TLCH, и PLC должен удалить сигнал подтверждения.

SP_TLSKP: Инструмент в шпинделе сломан

Если PLC обнаруживает поломку инстумента и записывает в 1 указатель SP_TLSKP, NC регистрирует в таблице для инструмента состояние “сломан”, затем записывает в 1 сигнал подтверждения SN_TLSKPA.

После этого он выдаёт указатель SN_TLCHI запроса смены инструмента.

После того, что NC возвращает сигнал подтверждения SN_TLSKPA, PLC должен выключить указатель SP_TLSKP.

SP_TLCD: Запрет счётчика стойкости

Если PLC переключает сигнал в 1, остановится счётчик стойкости инструмента в данном шпинделе.

☞***Внимание!*** *Указатели*

SP_SSTART,
SP_OREQ,
SP_SSYNCR,
SP_SLCLR

означают исключающие друг друга команды. Из них всегда только один может иметь значение=1!

SP_OSW: Шпиндель находится на включателе ориентировки.

Если бит #4 OSW=1 параметра N0609 Spindle Encoder Config, ориентировка шпинделя выполняется на ввод интерфейса. приём сигнала включателя в любом случае надо выполнять в быстром модуле Int0 программы PLC из-за частоты отбора проб. Состояние включателя надо копировать на укзатель SP_OSW в программе PLC. Оператор шпинделя приступает к ориентировке по состоянию 1 указателя SP_OSW. Точность решения является частным случаем, её определение требует измерения.

7.14.2 Переменные двойного слова для шпинделя

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
SN_1	Биты обращения шпинделем, переданным от NC для PLC (DWORD)	**SP_1**	Биты обращения шпинделем, переданным от PLC для NC (DWORD)
SN_NCOM	Основной сигнал чисел оборотов шпинделя, выданного приводу в об/мин (DWORD)	**SP_PRG**	Запрограммированное число оборотов шпинделя (DWORD)
SN_NACT	Актуальное число оборотов, измеренное с датчика шпинделя (DWORD)	**SP_ROT**	Код состояния вращения шпинделя (3, 4, ...) (DWORD)
		SP_RNG	Код диапазона шпинделя (11, 12, …, 18) (DWORD)
		SP_MAST	Индекс шпинделя-мастера шпинделя (0,1,2...) (DWORD)
		SP_ASSIGN	Назначение шпинделя к номеру канала (1,2,...) (DWORD)
		SP_ACTT	Номер инструмента в шпинделе (DWORD)

Переменные шпинделя с двойным словом, идущие от NC в сторону PLC

SN_NCOM: Основной синал чисел оборотов шпинделя, выданного приводу в об/мин (DWORD)

В состоянии ***1 указателя SP_SSTART*** значение команды чисел оборотов, выданное приводу шпинделя в размерности об/мин.

Значение регистра SN_NCOM,

– если ***SP_PAR=0***, число оборотов, заданное в регистре SP_PRG
принимая во внимание разбег и выбег,
умножая на override шпинделя (SP_SOVER),
ограничивая параметрами N0641+n Rn S Min и N0649+n Rn S Max,
в состоянии G96 ограничивая числом оборотов, заданным командой G92 S и параметром N0688 Min Spindle Speed G96.

– если ***SP_PAR=1***, число оборотов, заданное параметром N0657+n Rn S Jog Speed. На него действует только разбег и выбег, но не действительны ни override шпинделя, ни ограничивающие параметры.

SN_NACT: Актуальное число оборотов, измеренное с датчика шпинделя (DWORD)

Если на шпинделе оборудован датчик, он показывает в любом состоянии шпинделя актуальное число оборотов шпинделя в размерности об/мин.

Переменные шпинделя с двойным словом, идущие от PLC в сторону NC

SP_PRG: Запрограммированное число оборотов шпинделя (DWORD)

Значение кода S, переданное в регистре CN_SC, PLC записывает в соответсвующий регистр SP_PRG. Его размерность об/мин.

В случае постоянной скорости резания, в состоянии G96 (CN_CSURFS=1), число оборотов шпинделя, записанное в регистр CN_CSPN, расчитанное при помощи NC к скорости резания, надо скопировать в соответсвующий регистр SP_PRG.

SP_ROT: Код состояния вращения шпинделя (3, 4, ...) (DWORD)

В регистр SP_ROT надо записать стандартные коды вращения: 3: M3, 4: M4, 5: M5, 19: M19.

Сюда надо записать код замыкания контура без ориентировки, заданный параметром N0821 No. of M Code for Spnd. Pos.

В этот регистр надо записать и коды M, заданные параметрами N0689 Spindle M Low и N0690 Spindle M High.

Код состояния вращения шпинделя выводится на экран FST.

SP_RNG: Код диапазона шпинделя (11, 12, …, 18) (DWORD)

Если на шпинделе происходит смена диапазона на код M, для 8 диапазонов надо использовать коды M11, M12, ..., M18.

Актуальный код диапазона шпинделя надо записать в регистр, даже тогда, если диапазон меняется на код S. В регистр можно записать только 11, 12, ..., 18.

Оператор щпинделя принимает во внимание параметры, зависящие от диапазона шпинделя на основании кода SP_RNG.

Его заполнение обязательно!

SP_MAST: Индекс шпинделя-мастера шпинделя (0,1,2...) (DWORD)

В случае синхронизации шпинделей, шпиндель-аппликат запросит синхронизацию с записыванием указателя SP_SSYNCR. В регистре SP_MAST шпинделя-аппликата, запрашивающего синхронизацию, надо задавать индекс шпинделя-мастера. (Номер шпаинделя - 1.)

SP_ASSIGN: Назначение шпинделя к номеру канала (1,2,...) (DWORD)

В этом регистре можно задавать, что данный шпиндель в каком канале работает. В регистр надо записать всегда номер канала: в случае 1-го канала 1 и т.д. Из программы деталей можно ссылаться на шпиндель только в том канале, к которому регистр SP_ASSIGN выделил шпиндель.

В ходе прогона программы может понадобиться, чтобы шпиндель попал в другой канал, и можно было оттуда выполнить с ним обработку. Переброс может выполнить программа PLC функцией M.

☞ ***Внимание!*** *Изменение регистра SP_ASSIGN разрешено выполнить только в функции, разгружающей буфер кадра!*
Его заполнение обязательно, после включение необходимо инициирование!

SP_ACTT: Номер инструмента в шпинделе (DWORD)

Если бит #7 TSP параметра N2901 Search Config будет 1, индикация актуального номера инструмента совершается в окошке FST из регистра SP_ACTT, индексированного по шпинделям.

Программа PLC сюда записывает номер инструмента, имеющегося в шпинделе. Он используется обычно на многошпидельных фрезерных станках.

7.14.3 Переменные с плавающей запятой для шпинделя

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
		SP_SOVER	Override шпинделя: если =1: 100% (double)

Переменные шпинделя с плавающей запятой, идущие от PLC в сторону NC

SP_SOVER: Override шпинделя: если =1: 100% (double)

По шпинделям стоит на распоряжение регистр override. В регистр надо записывать значение override с плавающей запятой. Если например

значение SP_SOVER=0.32 - 32%

значение SP_SOVER=1.0 - 1007%

В случае использования станочной панели оператора NCT, положение включателя override шпинделя можно брать из регистра MKSOVER, или считать их происхождение с клавиш MB_SMAX, MB_S100, MB_SMAX.

Нижний и верхний предел override шпинделя надо установить в программе PLC!

☞ ***Внимание!*** *MKSOVER является целым числом типа DWORD, а SP_SOVER - double с плавающей запятой, значит, установка override требует конверсию (команду FLT) из числа с фиксированной запятой в число с плавающей запятой. Индексирование SP_SOVER выполняется по 2!*

7.15 Переменные оператора для канала

Переменные оператора канала являются такими переменными, идущими от NC в PLC, или от PLC в NC, которые индексируются по каналам. Все перечисленные здесь символи относятся к первому(с индексом 0) каналу. Соответствующие переменные остальных каналов доступны с индексированной адресовкой. Управление может обращаться не более 8 каналами.

Переменные,

начинающиеся CN идут от NC в PLC (Вводы), а

начинающиеся CP идут от PLC в NC (Выводы).

7.15.1 Бинарные переменные для канала

Вводы		**Выводы**	
Символ	**Описание**	**Символ**	**Описание**
CN_M1STB	Сигнал вписывания функции 1.M	**CP_START**	Запрос старта
CN_M2STB	Сигнал вписывания функции 2.M	**CP_STOP**	Запрос стопа
CN_M3STB	Сигнал вписывания функции 3.M	**CP_JOG**	Запрос режима перемещения
CN_M4STB	Сигнал вписывания функции 4.M	**CP_INCR**	Запрос режима сдвига по шагам
CN_M5STB	Сигнал вписывания функции 5.M	**CP_HNDL**	Запрос режима маховичка
CN_M6STB	Сигнал вписывания функции 6.M	**CP_REFP**	Запрос режима приёма референтной точки
CN_M7STB	Сигнал вписывания функции 7.M	**CP_EDIT**	Запрос режима редактирования
CN_M8STB	Сигнал вписывания функции 8.M	**CP_AUTO**	Запрос автоматического режима
CN_SSTB	Сигнал вписывания функции S	**CP_MDI**	Запрос режима ручного ввода данных
CN_TSTB	Сигнал вписывания функции T	**CP_JOGRAP**	Перемещение Jog быстрым ходом
CN_AUX1STB	Сигнал вписывания 1-й вспомогательной функции	**CP_TAXF**	Запрос ручного перемещения в направлении инструмента
CN_AUX2STB	Сигнал вписывания 2-й вспомогательной функции	**CP_TRGAF**	Запрос ручного перемещения пе рпендикулярно к инструменту
CN_AUX3STB	Сигнал вписывания 3-й вспомогательной функции	**CP_TTCRF**	Запрос ручного перемещения вокруг точки центра инструмента

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
CN_GSTB	Не используется	**CP_TBLB**	Запрос ручного перемещения по столу
CN_BKBUF	Обрабатываемый кадр в буфере	**CP_INTDREQ**	Не используется
CN_STPREQ	Не используется	**CP_TLCM**	Режим замера коррекции по длине вкл.
CN_ALRM	Не используется	**CP_S2TS**	Выбор замера по длине, относящего к S2
CN_OPMES	Не используется	**CP_WPCM**	Режим замера нулевой точки заготовки вкл.
CN_INTD	Состояние блокировки	**CP_S2WS**	Выбор замера нулевой точки, относящегося к S2
CN_WTNG	Ожидает сборный код M других каналов	**CP_AHND**	Запрос смещения нулевой точки с маховичком
CN_ITFCHK	Не используется	**CP_WMCAXF**	Запрос подачи, параллельной направлению поворота системы координат заготовки
CN_ITFALM	Не используется		
CN_CSURFS	Постоянная скорость резания (G96)		
CN_POLYT	Режим точения многоугольника (G51.2)		
CN_INCH	Данные размера запрограммированы в дюймах (G20)		
CN_HSHP	Высокоскоростный высокоточный режим (G5.1)		
CN_CSACK	Не используется		
CN_WPCNT	Обработанная заготовка = изготавливаемая хаготовка		
CN_EGBMD	Режим EGB (G81.8)		
CN_RTRFIN	Выхватка закончена		
CN_IPSTP	Интерполятор стоит	**CP_SGLBK**	Режим по кадрам вкл.
CN_IPEPTY	Интерполятор срабатывал	**CP_CNDSP**	Условный стоп вкл.
CN_CBFR	Запрос подачи кадра точения	**CP_TEST**	Запрос режима теста
CN_OVDIS	Override запрещено (G63)	**CP_MLCK**	Запрос режима станок закрыт
CN_THRD	Нарезание резьбы (G33, G34)	**CP_DRRUN**	Запрос сухого бега
CN_THRDC	Цикл нарезания резьбы (G76, G78)	**CP_BKRST**	Запрос кадр снова
CN_TAP	Нарезание резьбы метчиком (G84)	**CP_BKRET**	Запрос кадр назад

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
CN_RTAP	Нарезаине резьбы метчиком с жёстким стержнем (G84.2...)	**CP_FLCK**	Запрос функция закрыта
CN_REFPG	Запрограммированный приём референтной точки (G28)	**CP_ABSOFF**	Не используется
CN_DWELL	Ожидание (G04)	**CP_CNDBK_1**	Условный кадр 1 вкл.
CN_SKIP	Набег на щуп (G31)	**CP_CNDBK_2**	Условный кадр 2 вкл.
CN_FREV	Подача за оборот (G95)	**CP_CNDBK_3**	Условный кадр 3 вкл.
CN_POSCHK	Ожидает сигнала в позиции	**CP_CNDBK_4**	Условный кадр 4 вкл.
CN_CHOP	Не используется	**CP_CNDBK_5**	Условный кадр 5 вкл.
CN_CHARP	Не используется	**CP_CNDBK_6**	Условный кадр 6 вкл.
CN_TLLE	Стойкость ссылочного инструмента кодом T закончилась	**CP_CNDBK_7**	Условный кадр 7 вкл.
		CP_CNDBK_8	Условный кадр 8 вкл.
CN_START	Состояние старта	**CP_FIN**	Сигнал готов, все функции выполнены
CN_STOP	Состояние стопа	**CP_RST**	Не используется
CN_JOG	Режим перемещения	**CP_RSTREW**	Не используется
CN_INCR	Режим сдвига по шагам	**CP_CSREQ**	Не используется
CN_HNDL	Режим мазовичка	**CP_NOWT**	Не надо дождаться кода сбора M от другого канала
CN_REFP	Режим приёма референтной точки	**CP_MINT**	Вызов макрокоманды прерывания
CN_EDIT	Режим редактирования	**CP_TMREN**	Разрешение таймера свободного использования
CN_AUTO	В автоматическом режиме	**CP_TSBD**	Не используется
CN_MDI	Режим ручного ввода данных	**CP_EGBRRQ**	Запрос выхватки инструмента в режиме EGB
CN_FLCK	Режим функция закрыта	**CP_M1ACK**	Функция 1.M выполнена
CN_TAXF	Режим ручного перемещения в направлении инструмента	**CP_M2ACK**	Функция 2.M выполнена
CN_TRGAF	Режим ручного перемещения перпендикулярно к инструменту	**CP_M3ACK**	Функция 3.M выполнена
CN_TTCRF	Режим ручного перемещения вокруг точки центра инструмента	**CP_M4ACK**	Функция 4.M выполнена
CN_TBLB	Режим ручного перемещения по столу	**CP_M5ACK**	Функция 5.M выполнена
CN_ABSOFF	Не используется	**CP_M6ACK**	Функция 6.M выполнена

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
CN_TEST	Режим теста	**CP_M7ACK**	Функция 7.M выполнена
CN_MLCK	Режим станок закрыт	**CP_M8ACK**	Функция 8.M выполнена
CN_DRRUN	Режим сухого бега	**CP_SACK**	Функция S выполнена
CN_BKRST	Состояние кадр снова	**CP_TACK**	Функция T выполнена
CN_BKRET	Состояние кадр назад	**CP_AUX1ACK**	Вспомогательная функция 1 выполнена
CN_AHND	Смещение нулевой точки с маховичком завершено	**CP_AUX2ACK**	Вспомогательная функция 2 выполнена
CN_WMCAXF	Включение подачи, параллельной направлению поворота системы координат заготовки	**CP_AUX3ACK**	Вспомогательная функция 3 выполнена
		CP_GACK	Не используется
CN_1100	Значение макропеременной #1100 (бит)	**CP_HOLD**	Стоп движения для каждой оси в канале
CN_1101	Значение макропеременной #1101 (бит)	**CP_CBFEN**	Разрешение подачи кадра точения
CN_1102	Значение макропеременной #1102 (бит)	**CP_FHNDL**	Режим подача с маховичка вкл.
CN_1103	Значение макропеременной #1103 (бит)	**CP_OVC**	Не используется
CN_1104	Значение макропеременной #1104 (бит)	**CP_HNDLS1**	Подача с маховичка 1.
CN_1105	Значение макропеременной #1105 (бит)	**CP_HNDLS2**	Подача с маховичка 2.
CN_1106	Значение макропеременной #1106 (бит)	**CP_HNDLS3**	Подача с маховичка 3.
CN_1107	Значение макропеременной #1107 (бит)	**CP_HNDLS4**	Подача с маховичка 4.
CN_1108	Значение макропеременной #1108 (бит)	**CP_LIM1DIS**	Запрет диапазона конечнего положения 1 с параметром
CN_1109	Значение макропеременной #1109 (бит)	**CP_LIM2DIS**	Запрет диапазона конечнего положения 2 с параметром
CN_1110	Значение макропеременной #1110 (бит))	**CP_LIM3DIS**	Запрет диапазона конечнего положения 3 с параметром
CN_1111	Значение макропеременной #1111 (бит)	**CP_LIMSEL**	Выбор диапазона 1В конечнего положения для каждой оси с параметром
CN_1112	Значение макропеременной #1112 (бит)	**CP_CHOPON**	Не используется

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
CN_1113	Значение макропеременной #1113 (бит)	**CP_SGOEN**	Разрешение второй таблицы геометрической коррекции
CN_1114	Значение макропеременной #1114 (бит)	**CP_SGOX**	Учёт второй геометрической коррекции на оси X
CN_1115	Значение макропеременной #1115 (бит)	**CP_SGOY**	Учёт второй геометрической коррекции на оси Y
CN_1116	Значение макропеременной #1116 (бит)	**CP_SGOZ**	Учёт второй геометрической коррекции на оси Z
CN_1117	Значение макропеременной #1117 (бит)	**CP_OSGNX**	Коррекция X с противиположным знаком (Xкоррекция=-Xкорреция)
CN_1118	Значение макропеременной #1118 (бит)	**CP_OSGNY**	Коррекция Y с противиположным знаком (Yкоррекция=-Yкоррекция)
CN_1119	Значение макропеременной #1119 (бит)	**CP_OSGNZ**	Коррекция Z с противиполож-ным знаком (Zкоррецция=-Zкоррекция)
CN_1120	Значение макропеременной #1120 (бит)	**CP_ROVLD**	Выключение перекрывания кадров быстрого хода
CN_1121	Значение макропеременной #1121 (бит)	**CP_RLSOT3**	Не используется
CN_1122	Значение макропеременной #1122 (бит)		
CN_1123	Значение макропеременной #1123 (бит)		
CN_1124	Значение макропеременной #1124 (бит)		
CN_1125	Значение макропеременной #1125 (бит)		
CN_1126	Значение макропеременной #1126 (бит)		
CN_1127	Значение макропеременной #1127 (бит)		
CN_1128	Значение макропеременной #1128 (бит)		
CN_1129	Значение макропеременной #1129 (бит)		
CN_1130	Значение макропеременной #1130 (бит)		
CN_1131	Значение макропеременной #1131 (бит)		

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
		CP_1000	Макропеременная #1000 (бит)
		CP_1001	Макропеременная #1001 (бит)
		CP_1002	Макропеременная #1002 (бит)
		CP_1003	Макропеременная #1003 (бит)
		CP_1004	Макропеременная #1004 (бит)
		CP_1005	Макропеременная #1005 (бит)
		CP_1006	Макропеременная #1006 (бит)
		CP_1007	Макропеременная #1007 (бит)
		CP_1008	Макропеременная #1008 (бит)
		CP_1009	Макропеременная #1009 (бит)
		CP_1010	Макропеременная #1010 (бит)
		CP_1011	Макропеременная #1011 (бит)
		CP_1012	Макропеременная #1012 (бит)
		CP_1013	Макропеременная #1013 (бит)
		CP_1014	Макропеременная #1014 (бит)
		CP_1015	Макропеременная #1015 (бит)
		CP_1016	Макропеременная #1016 (бит)
		CP_1017	Макропеременная #1017 (бит)
		CP_1018	Макропеременная #1018 (бит)
		CP_1019	Макропеременная #1019 (бит)
		CP_1020	Макропеременная #1020 (бит)
		CP_1021	Макропеременная #1021 (бит)
		CP_1022	Макропеременная #1022 (бит)
		CP_1023	Макропеременная #1023 (бит)
		CP_1024	Макропеременная #1024 (бит)
		CP_1025	Макропеременная #1025 (бит)
		CP_1026	Макропеременная #1026 (бит)
		CP_1027	Макропеременная #1027 (бит)
		CP_1028	Макропеременная #1028 (бит)
		CP_1029	Макропеременная #1029 (бит)
		CP_1030	Макропеременная #1030 (бит)
		CP_1031	Макропеременная #1031 (бит)

Бинарные переменные канала, идущие от NC в PLC

CN_M1STB: Сигнал вписывания функции 1. M
CN_M2STB: Сигнал вписывания функции 2. M
CN_M3STB: Сигнал вписывания функции 3. M
CN_M4STB: Сигнал вписывания функции 4. M
CN_M5STB: Сигнал вписывания функции 5. M
CN_M6STB: Сигнал вписывания функции 6. M
CN_M7STB: Сигнал вписывания функции 7. M
CN_M8STB: Сигнал вписывания функции 8. M

В программе деталей в один кадр можно писать не более 8 различных кодов M, то есть NC в одном кадре 8 различных кодов M может передать для PLC.

Если в программу деталей запрограммирована функция M, оператор канала

– записывает значение кода M в регистр CN_MnC (n=1, 2, ..., 8)

– затем сигнал вписывания ***CN_MnSTB*** NC записывает в 1 ***в период цикла 1 PLC***.

Управление записывает значение 8 различных кодов M в 8 различные регистры CN_MnC (n=1, 2, ..., 8). К 8 регистрам передачи принадлежат 8 сигналов вписывания CN_MnSTB (n=1, 2, ...8). Под действием сигналов вписывания, на основании полученного кода PLC декодирует функцию.

Если полученный код прикрывает существующую на станке функцию, PLC установит в 0 сигнал подтверждения CP_MnACK, согласно сигналу вписывания. Сигнал подтверждения CP_MnACK разрешено включить в 1 только после полного выполнения функции.

CN_SSTB: Сигнал вписывания функции S

Если в программу деталей запрограммирована функция S, оператор канала

– записывает в регистр CN_SC значение чисел оборотов, запрограммированное в функции S,

– записывает в регистр CN_SSEL номер ссылочного шпинделя (1...16)

– записывает в регистр CN_RNGREQ код диапазона, относящегося к запрограммированному числу оборотов (11...18),

– затем записывает указатель ***CN_SSTB*** в 1 ***в период цикла 1 PLC***.

Под действием сигналов вписывания PLC декодирует записанные в указанные выше регистры коды и установит в 0 сигнал подтверждения CP_SACK. Сигнал подтверждения CP_SACK разрешено включить в 1 только после полного выполнения функции.

CN_TSTB: Сигнал вписывания функции T

Если в программу деталей запрограммирована функция T, оператор канала.

– записывает в регистр CN_TC номер инструмента, запрограммированный в функции T, ***отрезая номер коррекции***, вызванный в токарном канале по адресу T,
– записывает в регистр CN_MGZNO, что ссылочный инструмент в каком по номеру магазине находится,
– записывает в регистр CN_POTNO, что ссылочный инструмент в каком по номеру кармане находится в данном магазине,
– просмотрит, что кончилась ли стойкость ссылочного инструмента, или группы инструментов, и согласно этому установит указатель CN_TLLE,
– затем записывает указатель ***CN_TSTB*** в 1 ***в период цикла 1 PLC***.

Под действием сигналов вписывания PLC декодирует записанные в указанные выше регистры коды и установит в 0 сигнал подтверждения CP_TACK. Сигнал подтверждения CP_TACK разрешено включить в 1 только после полного выполнения функции.

CN_AUX1STB: Сигнал вписывания 1-й вспомогателной функции
CN_AUX2STB: Сигнал вписывания 2-й вспомогателной функции
CN_AUX3STB: Сигнал вписывания 3-й вспомогателной функции

На параметре N1332+n Aux Fu Addrn (n=1..3) можно выделить три различных адресов из A, B, C, U, V, W. Если в программе деталей ссылаться на адрес заданной здесь трёх вспомогателных функций, оператор канала
– записывает в регистр CN_AUXnC (n=1...3) значение, запрограммированное по адресу вспомогателной функции (всегда целое число DWORD),
– затем записывает указатель ***CN_AUXnSTB*** (n=1...3) в 1 ***в период цикла 1 PLC***.

Под действием сигналов вписывания PLC декодирует записанные в указанные выше регистры коды и установит в 0 сигнал подтверждения CP_AUXnACK (n=1...3). Сигнал подтверждения CP_AUXnACK разрешено включить в 1 только после полного выполнения функции.

CN_GSTB: Не используется

CN_BKBUF: Обрабатываемый кадр в буфере

NC переключает указатель в 1, если
– в автоматическом режиме (CN_AUTO=1) выделена одна программа для прогона по автомату,
– находится в режиме MDI (CN_MDI=1),
– в режиме jog, jog с приращением, или маховичка (CN_JOG=1, CN_INCR=1, или CN_HNDL=1) закрыть ввод одного кадра до тех пор, пока кадр не выполнен.

CN_STPREQ: Не используется

CN_ALRM: Не используется

CN_OPMES: Не используется

CN_INTD: Состояние блокировки

Оператор канала записывает указатель в 1, если в автоматическом режиме прогон программы прерван. Прерывание будет в следующих случаях:

– во время прогона программы выходили из автоматического режима,

– в автоматическом режиме возникает аварийное состояние.

В процессе прерывания оператор канала

– сохраняет состояние выполнения программы,

– позицию прерывания

– состояние сигналов подтверждения функции(CP_MnACK, CP_SACK, CP_TACK, CP_AUXnACK),

Возвращаясь в автоматический режим, прерванную программу можно продолжать в зависимости от условий CN_BKRST, CN_BKRET.

Те функции, сигнал подтверждения которых был 0, при прерывании снова выдаются.

CN_WTNG: Ожидает сборный код M других каналов

В многоканальных управлениях понадобится, что выполнение программы деталей, работающей в одном канале прерывать до тех пор, пока один или несколько каналов не достигает определённую точку выполнения программы. Это называется синхронизацией, или дожижанием друг друга каналов. Синхронизацию каналов можно выполнить кодами M. На параметрах N2201 Waiting M Codes Min и N2202 Waiting M Codes Max можно выделить диапазон кода M.

Дожидание полностью NC выполняет, этим для PLC не приходится заниматься. В том случае, если в данном канале имеется дожидание, NC записывает указатель CN_WTNG в 1.

CN_ITFCHK: Не используется

CN_ITFALM: Не используется

CN_CSURFS: Постоянная скорость резания (G96)

Если канал находится в состоянии G96 (расчёт постоянной скорости резания),

– NC записывает указатель CN_CSURF в 1,

– PLC расчитывает по циклам и записывает в регистр CN_CTSPN число оборотов шпинделя, относящееся к мгновенным координатам,

– записывает в регистр CN_NMAX максимальное число оборотов, запрограммированное командой G92 S.

Задачей программы PLC является, что в состоянии CN_CSURF=1 скопировать значение, полученное в регистре CN_CTSPN, в регистр SP_PRG соответствующего шпинделя (определённого регистром CP_SINP).

CN_POLYT: Режим точения многоугольника (G51.2)

Если оператор канала выполняет команду G51.2 точения многоугольника, указатель переключает в 1. Указатель выключается командой G50.2.

Если указатель находится в состоянии 1, PLC должен предотвратить, чтобы выделённый в регистре CP_POLYSL шпиндель выполнил любую функцию.

CN_INCH: Данные размеры запрограммированы в дюймах (G20)

При вводе метрических данных указатель =0 G21.

При вводе дюймовых данных указатель =1 G20.

В зависимости от указателя и бита #0 IND параметра N0104 Unit of Measure, PLC должен записывать величину шага в регистр с приращением CP_INC в режиме jog с приращением. Пример:

– Если IND=0, CN_INCH=0 и нажато MB_I100: CP_INC=0.1

– Если IND=0, CN_INCH=1 и нажато MB_I100: CP_INC=0.254

CN_HSHP: Высокоскоростный высокоточный режим (G5.1)

CN_CSACK: Не используется

CN_WPCNT: Обработанная заготовка = изготавливаемая заготовка

Оператор может установить в управлении счётчик заготовки в окошке Время/Счётчики. Если число обработанных заготовок достигло число изготавливаемых заготовок, управление запишет на период 1 цикла PLC указатель CN_WPCNT в 1.

Счётчик изготавливаемых заготовок управление увеличивает до номера функции M, заданной параметром N2305 Part Count M. Например: если параметр =30, счётчик увеличивает до каждой функции M30.

CN_EGBMD: Режим EGB (G81.8)

Если оператор канала выполняет функцию нарезания зубьев G81.8 (электронный привод), указатель CN_EGBMD переключает в 1. Указатель удаляется функцией G80.8 (электронный привод выкл.). (См.: Группа параметров N1800 EGB Contr)

CN_RTRFIN: Выхватка закончена

Если оператор канала выполняет функцию нарезания зубьев G81.8 (электронный привод), (CN_EGBMD=1) и PLC запросит выхватку зуборезного инструмента указателем CP_EGBRRQ=1, NC отводит инструмент в направление и на расстояние, определённое параметром N1804 Retr. Dist. В конце отвода оператор канала

записывает указатель CN_RTRFIN в 1. После этого PLC должен выключить указатель CP_EGBRRQ=0.

CN_IPSTP: Интерполятор стоит

Указатель принимает состояние CN_IPSTP=1 в следующих случаях:

– в канале нет прогона программы,

– в канале идёт прогон программы, но значение override 0 CP_FOVER=0. Это правда и в кадрах позиционирования G0.

– канал попадает в состояние стопа CN_STOP=1 (Например: под действием клавиши стопа, в режиме по кадрам, и т.д.).

CN_IPEPTY: Интерполятор срабатывал

Указатель принимает состояние CN_IPEPTY=1 в следующих случаях:

– в канале нет прогона программы,

– в канале идёт прогон программы, но интерполятор стоит, например потому, что канал выполняет кадр функции.

CN_CBFR: Запрос подачи кадра точения

Оператор канала запрашивает разрешение движения подачи от PLC включением указателя CN_CBFR=1 в следующих случаях:

– линейная интерполяция G01,

– круговая интерполяция G02, G03,

– нарезание резьбы G33,

– в каждом цикле, в котором выполняется любой из перечисленных выше кадров.

Подача (движение) не запускается до тех пор, пока PLC не разрешает это на указателе CP_CBFEN=1, например, пока шпиндель не вращается.

CN_OVDIS: Override запрещено (G63)

Оператор канала сообщает о состоянии override и запрета стопа для PLC включением указателя CN_OVDIS=1 в следующих случаях:

– состояние запрета override вкл. G63 (выключает: G61, G62, G64),

– в кадре нарезания резьбы G33 (его выключает прочая интерполяция, например G0, G1),

– в цикле нарезания резьбы метчиком G74, G84, пока работает в отверстии,

– в простом цикле нарезания резьбы G78, когда нарезает резьбу,

– в сложном цикле нарезания резьбы G76, когда нарезает резьбу.

CN_THRD: Нарезание резьбы (G33, G34)

Оператор канала переключает указатель в 1 CN_THRD=1, когда

– выполняет кадр нарезания резьбы G33 (его выключает прочая интерполяция, например G0, G1),

– в простом цикле нарезания резьбы G78 нарезает резьбу,

– в сложном цикле нарезания резьбы G76 нарезает резьбу.

CN_THRDC: Цикл нарезания резьбы (G76, G78)

Оператор канала переключает указатель в 1 CN_THRDC=1, когда

– в простом цикле нарезания резьбы G78 нарезает резьбу,

– в сложном цикле нарезания резьбы G76 нарезает резьбу.

CN_TAP: Нарезание резьбы метчиком (G84)

Оператор канала переключает указатель в 1 CN_TAP=1, когда

– в цикле нарезания резьбы метчиком G74, G84 работает в отверстии.

CN_RTAP: Нарезание резьбы метчиком с жёстким стержнем (G84.2...)

Оператор канала переключает указатель в 1 CN_RTAP=1, когда

– в цикле нарезания резьбы метчиком с жёстким стержнем G84.2, G84.3 работает в отверстии.

CN_REFPG: Запрограммированный приём референтной точки (G28)

Оператор канала переключает указатель в 1 CN_REFPG=1, когда

– в команде G28 принимает референтную точку.

CN_DWELL: Ожидание (G04)

Оператор канала переключает указатель в 1 CN_DWELL=1, когда

– в кадре G4 выполняет ожидание,

– выполняет ожидание, запрограммированное в цикле сверления.

CN_SKIP: Прогон щупа (G31)

Оператор канала переключает указатель в 1 CN_SKIP=1, когда

– выполняет функцию G31.

CN_FREV: Подача за оборот(G95)

Оператор канала переключает указатель в 1 CN_FREV=1, когда

– перемещается подачей за оборот (G95).

В случае G94 (подача за минуту) указатель CN_FREV=0. В кадре позиционирования (G0, G53) указатель выключает даже тогда, когда имеется состояние G95.

CN_POSCHK: Ожидает сигнала в позиции

Оператор канала переключает указатель в 1 CN_POSCHK=1,

– в конце каждого движения быстрого хода (G0, G53), если параметр N1337 Execution Config будет #0 PCH=1,

– в конце каждого кадра движения, в котором запрограммирована G9,

– в конце каждого кадра движения, в состоянии точного останова G61,

и дождётся, чтобы значение отставания на каждой оси, учавствующей в движении, попал внутрь окна, определённого параметром N0516 Inpos. После этого удаляет указатель CN_POSCHK.

CN_CHOP: Не используется

CN_CHARP: Не используется

CN_TLLE: Стойкость инструмента, ссылочного кодом T закончилась

При каждом вызове инструмента (код T) оператор канала просмотрит, что стойкость ссылочного инструмента, или группы инструментов закончилась ли, предпологая, что параметр N2900 Tool M. Config будет #0 TMU=1.

Если стойкость закончилась, NC параллельно с сигналом CN_TSTB записывает сигнал CN_TLLE в 1.

CN_START: Состояние старта

Это есть сигнал подтверждения запроса старта CP_START.

Когда PLC включает сигнал CP_START, оператор канала просмотрит, что

– в данном режиме (такие: авто, MDI, jog, jog с приращением, маховичок) можно ли выполнить программу, или индивидуальный кадр,

– имеется ли выделённая программа, или закрытый индивидуальный кадр для выполнения,

– нет ли прочего препятствия запуску, например, сообщение об ошибке.

Если перечисленные выше условия удовлетворяются, оператор канала вернёт состояние CN_START=1, по которому можно включить лампу клавиши старта (например ML_START).

Оператор канала удаляет состояние CN_START, если NC

– попадает в состояние стопа (CN_STOP=1)

– выполнил выделённую на исполнение программу, или закрытый индивидуальный кадр.

CN_STOP: Состояние стопа

Это есть сигнал подтверждения запроса стопа CP_STOP.

Когда PLC включает сигнал CP_STOP, оператор канала просмотрит, что

– в данном режиме (такие: авто, MDI, jog, jog с приращением, маховичок) имеется ли программа, или индивидуальный кадр под выполнением,

– нет ли прочего препятствия останову, например состояние override и запрет стопа.

Если перечисленные выше условия удовлетворяются, оператор канала остановит интерполяцию и вернёт состояние CN_STOP=1, по которому можно включить лампу клавиши стопа (например ML_STOP).

Оператор канала удаляет состояние CN_STOP, если NC

– попадает в состояние старта (CN_START=1)
– выполнил выделённую на исполнение программу, или закрытый индивидуальный кадр.

CN_JOG: Режим перемещения
CN_INCR: Режим сдвига по шагам
CN_HNDL: Режим маховичка
CN_REFP: Режим приёма референтной точки
CN_EDIT: Режим редактирования
CN_AUTO: Автоматический режим
CN_MDI: Режим ручного ввода данных

Сигналы подтверждения запроса режима CP_JOG, CP_INCR, CP_HNDL, CP_REFP, CP_EDIT, CP_AUTO и CP_MDI.
Когда PLC запросит смену режима на одном из указателей CP_xxx, оператор канала просмотрит, что
– в том режиме, в котором канал находится, имеется ли программа, или индивидуальный кадр под выполнением,
– если имеется выполнение программы, сразу остановит интерполятор, или в состоянии override и запрета стопа дождётся, чтобы прекратилось это состояние,
– если идёт выполнение индивидуального кадра в режиме jog, jog с приращением, маховичка, или идёт выполнение программы в режиме MDI, удаляет полное выполнение,
– если идёт выполнение программы в автоматическом режиме, установит состояние блокировки CN_INTD=1,
– переходит указателем CP_xxx в желаемый режим и включает указатель CN_xxx.
В состоянии 1 указателя CN_xxx PLC может включить лампу клавиши соответствующего режима (например ML_xxx).

CN_FLCK: Режим функция закрыта

Это есть сигнал подтверждения запроса функция закрыта CP_FLCK.
Когда PLC включает сигнал запроса функция закрыта CP_FLCK=1, оператор канала просмотрит, что
– можно ли изменить состояние указателя CN_FLCK, то есть не находится ли управление в автоматическом режиме (CN_AUTO=1), или в одном из режимов MDI (CN_MDI=1),
– если нет, изменяет состояние указателя CN_FLCK в противоположное.
В состоянии 1 указателя CN_FLCK, PLC может включить лампу клавиши станок закрыт (например ML_FLCK).

В состоянии 1 указателя CN_FLCK а оператор канала не передаёт никакую функцию для PLC: не выдаёт сигналы вписывания CN_MnSTB, CN_SSTB, CN_TSTB, CN_AUXnSTB в сторону PLC.

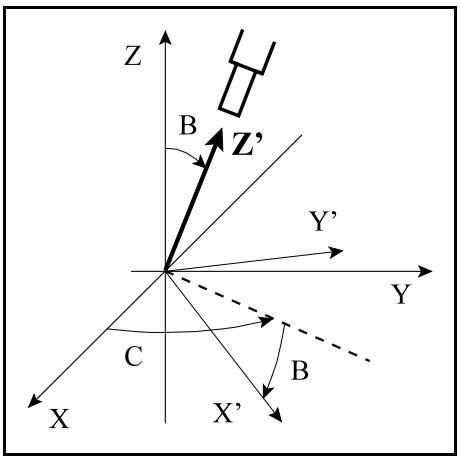
CN_TAXF: Режим ручного перемещения в направлении инструмента

CP_TAXF является сигналом подтверждения запроса ручного перемещения в направлении инструмента. Указатель NC включает его на 5-координатном станке типа ***головка-головка, или головка-стол***, ***в режиме jog, инкрементный jog и маховичка***. На станках типа стол-стол он не работает.

В состоянии CN_TAXF=1, ***выбирав ось в направлении инструмента, заданную параметром*** N3201 Tool Axis Direction, ***инструмент перемещается в направлении инструмента в зависимости от углового положения вращающихся осей,*** определённого параметром N3204 No. of the First Rot. Ax.и N3208 No. of the Second Rot. Ax.

Например:

Пусть будет наш станок типа головка-головка, то есть N3200 Mechanical Type=Head-Head, N3201 Tool Axis Direction=Z, то есть инструмент в исходном положении вращающихся осей параллельный направлению Z, N3204 No. of the First Rot. Ax.=C и N3208 No. of the Second Rot. Ax.=B. При этом ***нажав клавишу Z jog, или выбирав ось Z для перемещения маховичка***, учитывая позицию осей B и C ***перемещается в направлении инструмента*** хоть вдоль всех трёх осей (X, Y, Z).



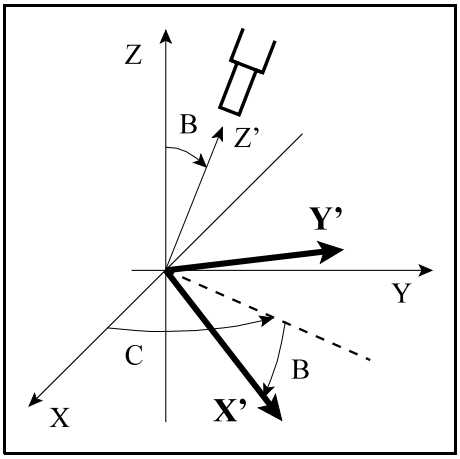
CN_TRGAF: Режим ручного перемещения перпендикулярно к инструменту

CP_TRGAF является сигналом подтверждения запроса ручного перемещения перпендикулярно к инструменту. Указатель NC включает его на 5-координатном станке типа ***головка-головка, или головка-стол***,***в режиме jog, инкрементный jog и маховичка***. На станках типа стол-стол он не работает.

В состоянии CN_TRGAF=1, ***выбирав ось в одном из направлении, перпендикулярном к заданной оси, заданную параметром*** N3201 Tool Axis Direction, ***инструмент перемещается в перпендикулярной к оси инструмента плоскости, в зависимости от углового положения вращающихся осей, определённого параметром*** N3204 No. of the First Rot. Ax. и N3208 No. of the Second Rot. Ax.

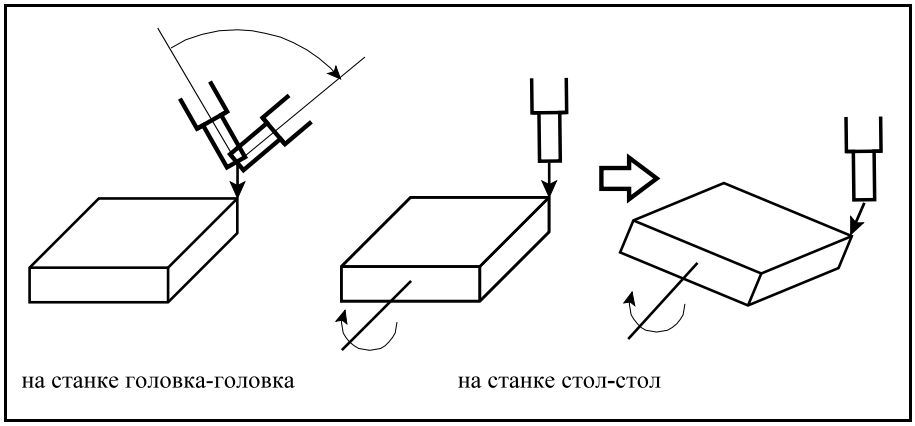
Например:

Пусть будет наш станок типа головка-головка, то есть N3200 Mechanical Type=Head-Head, N3201 Tool Axis Direction=Z, то есть инструмент в исходном положении вращающихся осей параллельный направлению Z, N3204 No. of the First Rot. Ax.=C и N3208 No. of the Second Rot. Ax.=B. При этом ***нажав клавишу X, илиY jog, а также выбирав ось X, или Y, для перемещения маховичка***, учитывая позицию осей B и C ***перемещается в перпендикулярной к инструменту плоскоти*** хоть вдоль всех трёх осей (X, Y, Z).



CN_TTCRF: Режим ручного перемещения вокруг точки центра инструмента

CP_TTCRF является сигналом подтверждения запроса ручного перемещения вокруг точки центра инструмента. Указатель NC включает его ***в случае всех трёх типов 5-координатного станка типа***, ***в режиме jog, инкрементный jog и маховичка***. В состоянии CN_TTCRF=1, ***перемещая вращающиеся оси,*** выделённые параметрами N3204 No. of the First Rot. Ax., или N3208 No. of the Second Rot. Ax., ***перемещается так, чтобы положение точки центра инструмента осталось неизменным по сравнению соответствующей точки заготовки***.

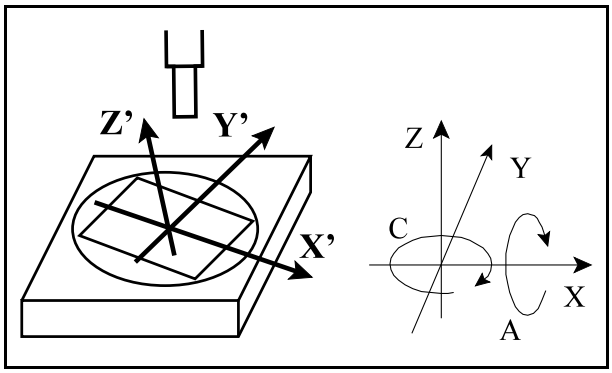


CN_TBLB: Режим ручного перемещения по столу

CP_TBLB является сигналом подтверждения запроса ручного перемещения по столу. Указатель NC включает его на 5-координатном станке типа ***стол-стол, или головка-стол***, ***в режиме jog, инкрементный jog и маховичка***. На станках головка-головка он не работает.

В состоянии CN_TBLB=1, ***в зависимости от углового положения вращающихся осей, определённых параметром*** N3204 No. of the First Rot. Ax., и N3208 No. of the Second Rot. Ax., ***перемещает линейные оси*** (X, Y, Z) так, что ***направлением, пер-***

пендикулярным к столу считает ось, заданную параметром N3201 Tool Axis Direction, ***а оставшиеся две оси - направлением, лежащим в плоскости стола***. ***Например***:



Пусть будет наш станок типа стол-стол, то есть N3200 Mechanical Type=Table-Table, N3201 Tool Axis Direction=Z, то есть в исходном положении вращающихся осей инструмент параллельный направлению Z, N3204 No. of the First Rot. Ax.=A и N3208 No. of the Second Rot. Ax.=C. При этом ***нажав клавишу X, или Y jog, а также выбирав ось X, или Y, для перемещения маховичка***, принимая во внимание позиции осей A и C, ***перемещается в плоскости стола***. ***Нажав клавишу Z jog, или выбирав ось Z для перемещения маховичка***, принимая во внимание позиции осей A и C ***перемешается в направлении, перпендикулярно к плоскости стола***.

CN_ABSOFF: Не используется

CN_TEST: Режим теста

Это есть сигнал подтверждения запроса режима теста CP_TEST.

Когда PLC включает сигнал запроса режима теста CP_TEST=1, оператор канала просмотрит, что

– можно ли изменить состояние указателя CN_TEST, то есть не находится ли управление в автоматическом режиме (CN_AUTO=1), или в одном из режимов MDI (CN_MDI=1),

– если нет, изменяет состояние указателя CN_TEST в противоположное.

В состоянии 1 указателя CN_TEST, PLC может включить лампу клавиши режима теста (например ML_TEST).

В состоянии 1 указателя CN_TEST оператор канала

– выполняет интерполяцию, каждый кадр, и кадры с подачей (G1, G2, G3, G33), с повышенной подачей, в зависимости от положения включателя override увеличивая скорость экспоненциально,

– не передаёт никакую команду движения контуру регулирования позиции, таким образом оси не движутся,

– не передаёт никакую функцию для PLC: не выдаёт сигналы вписывания CN_MnSTB, CN_SSTB, CN_TSTB, CN_AUXnSTB в сторону PLC.

CN_MLCK: Режим Станок закрыт

Это есть сигнал подтверждения запроса режима станок закрыт CP_MLCK.

Когда PLC включает сигнал запроса режима станок закрыт CP_MLCK=1, оператор канала просмотрит, что

– можно ли изменить состояние указателя CN_MLCK, то есть не находится ли управление в автоматическом режиме (CN_AUTO=1), или в одном из режимов MDI (CN_MDI=1),

– если нет, изменяет состояние указателя CN_MLCK в противоположное.

В состоянии 1 указателя CN_MLCK, PLC может включить лампу клавиши режима станок закрыт (например ML_MLCK).

В состоянии 1 указателя CN_MLCK оператор канала

– выполняет интерполяцию, каждый кадр с запрограммированной подачей, с учётом включателей override (подача, быстрый ход),

– если включен сухой бег CN_DRRUN=1, кадры с подачей (G1, G2, G3, G33) выполняет с повышенной подачей, определённой параметром N0305 Max Feed,

– не передаёт никакую команду движения контуру регулирования позиции, таким образом оси не движутся,

– не передаёт никакую функцию для PLC: не выдаёт сигналы вписывания CN_MnSTB, CN_SSTB, CN_TSTB, CN_AUXnSTB в сторону PLC.

CN_DRRUN: Режим Сухой бега

Это есть сигнал подтверждения запроса режима сухого бега CP_DRRUN.

Когда PLC включает сигнал запроса режима сухого бега CP_DRRUN=1, оператор канала просмотрит, что

– можно ли изменить состояние указателя CN_DRRUN то есть не находится ли управление в автоматическом режиме (CN_AUTO=1),или в одном из режимов MDI (CN_MDI=1),

– если нет, изменяет состояние указателя CN_DRRUN в противоположное.

В состоянии 1 указателя CN_DRRUN, PLC может включить лампу клавиши режима сухого бега (например ML_DRRUN).

В состоянии 1 указателя CN_DRRUN оператор канала

– кадры с подачей (G1, G2, G3, G33) выполняет с повышенной подачей, определённой параметром N0305 Max Feed,

– все команды движения выдаёт контуру регулирования позиции, таким образом оси движутся, при условии, что станок не закрыт (CN_MLCK=0),

– все функции передаёт для PLC, при условии, что станок не закрыт (CN_MLCK=0), и функция не закрыта (CN_MLCK=0).

CN_BKRST: Состояние кадр снова

Это есть сигнал подтверждения запроса кадр снова CP_BKRST.

Когда PLC включает сигнал запроса кадр снова CP_BKRST=1, оператор канала просмотрит, что

– имеется ли прерванная в автоматическом выполнении программа, то есть CN_INTD=1,

– если да, включает указатель CN_BKRST,

В состоянии 1 указателя CN_BKRST, PLC может включить лампу клавиши кадр снова (например ML_BKRST).

В автоматическом режиме под действием старта оператор канала

– выдаёт для PLC - сигналами вписывания через регистры передачи - те функции, которые до момента прерывания ещё не выполнил,

– вернётся в начальную точку прерванного кадра,

– затем отсюда продолжает обработку.

CN_BKRET: Состояние кадр назад

Это есть сигнал подтверждения запроса кадр назад CP_BKRET.

Когда PLC включает сигнал запроса кадр назад CP_BKRET=1, оператор канала просмотрит, что

– имеется ли прерванная в автоматическом выполнении программа, то есть CN_INTD=1,

– если да, включает указатель CN_BKRET,

В состоянии 1 указателя CN_BKRET, PLC может включить лампу клавиши кадр назад (например: ML_BKRET).

В автоматическом режиме под действием старта оператор канала

– выдаёт для PLC - сигналами вписывания через регистры передачи - те функции, которые до моменте прерывания ещё не выполнил,

– вернётся в прерванную позицию прерванного кадра,

– затем отсюда продолжает обработку.

CN_AHND: Смешение нулевой точки маховичком завершено

CP_AHND является сигналом подтверждения запроса смещения нулевой точки маховичком. В Автоматическом режиме и в режиме Ручного ввода данных (MDI) можно его завершить, хоть и в состоянии старта.

При этом выбор оси и величину шага надо разрешить в PLC соответственно как в режиме маховичка. Целесообразно по причине безопасности ограничивать величину шага до 0.001 мм, а может быть до 0.01 мм.

Под действием маховичка перемещается выбранная ось, перемещение учитывается не в абсолютной позиции, а в строке Маховичка таблицы Нулевой точки. Введённое маховичком смещение нулевой точки удаляется по M30 и под действием reset.

CN_WMCAXF: Включение подачи, параллельной направлению поворота системы координат заготовки

В режиме Jog, Инкрементного jog, или Маховичка PLC может запросить индикатором CP_WMCAXF, чтобы ручное перемещение осей происходило согласно первоначальному направлению, или согласно повороту системы координат заготовки. Сигнал CN_WMCAXF является знаком подтверждения запроса. Если сигнал =0: ручное перемещение происходит по первоначальному направлению осей,

=1: ручное перемещение происходит по повернутому направлению осей.

CN_1100: значение макропеременной #1100 (бит)
CN_1101: значение макропеременной #1101 (бит)
CN_1102: значение макропеременной #1102 (бит)
CN_1103: значение макропеременной #1103 (бит)
CN_1104: значение макропеременной #1104 (бит)
CN_1105: значение макропеременной #1105 (бит)
CN_1106: значение макропеременной #1106 (бит)
CN_1107: значение макропеременной #1107 (бит)
CN_1108: значение макропеременной #1108 (бит)
CN_1109: значение макропеременной #1109 (бит)
CN_1110: значение макропеременной #1110 (бит)
CN_1111: значение макропеременной #1111 (бит)
CN_1112: значение макропеременной #1112 (бит)
CN_1113: значение макропеременной #1113 (бит)
CN_1114: значение макропеременной #1114 (бит)
CN_1115: значение макропеременной #1115 (бит)
CN_1116:значение макропеременной #1116 (бит)
CN_1117: значение макропеременной #1117 (бит)
CN_1118: значение макропеременной #1118 (бит)
CN_1119: значение макропеременной #1119 (бит)
CN_1120: значение макропеременной #1120 (бит)
CN_1121: значение макропеременной #1121 (бит)
CN_1122: значение макропеременной #1122 (бит)
CN_1123: значение макропеременной #1123 (бит)
CN_1124: значение макропеременной #1124 (бит)
CN_1125: значение макропеременной #1125 (бит)
CN_1126: значение макропеременной #1126 (бит)
CN_1127: значение макропеременной #1127 (бит)
CN_1128: значение макропеременной #1128 (бит)
CN_1129: значение макропеременной #1129 (бит)
CN_1130: значение макропеременной #1130 (бит)
CN_1131: значение макропеременной #1131 (бит)

Пользователь из программы деталей может вставить бинарные указатели PLC с передачей значения для макропеременных, далее может их удалить. По каналам 32 указательных битов стоят на распоряжение пользователя для коммуникации с PLC: #1100, #1101, ..., #1131.

Например, записанная в программу деталей команда

#1109=1

установит указатель CN_1109=1 PLC. Команда

#1109=0

удаляет указатель CN_1109=0 PLC.

Бинарные переменные канала, идущие от PLC в NC

CP_START: Запрос старта

Когда оператор нажимает клавишу старта, программа PLC должна просмотреть, что со стороны станка нет ли препятствия вызыванию старта. Если нет, то записывает в 1 указатель запроса старта CP_START, тем самым запросит у оператора канала состояние старта.

В состоянии CP_START=1 оператор канала просмотрит, что можно ли запускать обработку в канале. Если да, состоянием CN_START=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_START даст состояние клавиши старта.

CP_STOP: Запрос стопа

Когда оператор нажимает клавишу стопа, программа PLC должна просмотреть, что со стороны станка нет ли препятствия вызыванию стопа. Если нет, то записывает в 1 указатель запроса стопа CP_STOP, тем самым запросит у оператора канала состояние стопа.

В состоянии CP_STOP=1 оператор канала просмотрит, что можно ли остановить обработку в канале. Если да, состоянием CN_STOP=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_STOP даст состояние клавиши стопа.

CP_JOG: Запрос режима перемещения
CP_INCR: Запрос режима сдвига по шагам
CP_HNDL: Запрос режима маховичка
CP_REFP: Запрос режима приёма референтной точки
CP_EDIT: Запрос режима редактирования
CP_AUTO: Запрос автоматического режима
CP_MDI: Запрос режима ручного ввода данных

Когда оператор нажимает какую-то клавишу изменения режима, программа PLC должна просмотреть, что со стороны станка нет ли препятствия изменению режима. Если нет, то записывает в 1 указатель запроса режима CP_xxx, принадлежащий к нажатой клавише режима.

В состоянии CP_xxx=1 оператор канала просмотрит, что можно ли загрузить желаемый режим. Если да, состоянием указателя соответствующего режима CN_xxx=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатели MB_JOG, MB_INCR, MB_HNDL, MB_REFP, MB_EDIT, MB_AUTO и MB_MDI дают состояние клавиш изменения режима.

CP_JOGRAP: Перемещение быстрым ходом Jog

Когда оператор нажимает клавишу быстый ход jog, программа PLC должна просмотреть, что со стороны станка нет ли препятствия перемещению осей станка быстрым ходом. Если нет, то записывает в 1 указатель запроса перемещения быстрым ходом Jog CP_JOGRAP.

В состоянии CP_JOGRAP=1 оператор канала те оси, которые принадлежат к каналу

– перемещает в режиме (jog) быстрым ходом, если она нажата вместе с клавишей выбора соответствующего направления,

– во время выполнения программы он увеличивает запрограммированную подачу F с помощью умножителя, определённого параметром N0313 Feed Mult.

В случае использования станочной панели оператора NCT указатель MB_JOGRAP даст состояние клавиш jog быстрым ходом. Лампой клавиши (ML_JOGRAP) обращается указатель CP_JOGRAP.

CP_TAXF: Запрос ручного перемещения в направлении инструмента

Под действием нажатия кнопки PLC запросит режим посредством включения указателя CP_TAXF. Включением указателя CN_TAXF, NC подтверждает запрос. NC включает указатель CN_TAXF только в каком-то ручном режиме. Подробное описание ручного перемещения инструмента см. при описании указателя CN_TAXF.

CP_TRGAF: Запрос ручного перемещения перпендикулярно к инструменту

Под действием нажатия кнопки PLC запросит режим посредством включения указателя CP_TRGAF. Включением указателя CN_TRGAF, NC подтверждает запрос. NC включает указатель CN_TRGAF только в каком-то ручном режиме. Подробное описание ручного перемещения, перпендикулярного к инструменту см. при описании указателя CN_TRGAF.

CP_TTCRF: Запрос ручного перемещения вокруг точки центра инструмента

Под действием нажатия кнопки PLC запросит режим посредством включения указателя CP_TTCRF. Включением указателя CN_TTCRF, NC подтверждает запрос. NC включает указатель CN_TTCRF только в каком-то ручном режиме. Подробное описание ручного перемещения вокруг точки центра инструмента см. при описании указателя CN_TTCRF.

CP_TBLB: Запрос ручного перемещения по столу

Под действием нажатия кнопки PLC запросит режим посредством включения указателя CP_TBLB. Включением указателя CN_TBLB, NC подтверждает запрос. NC включает указатель CN_TBLB только в каком-то ручном режиме. Подробное описа-

ние ручного перемещения вокруг точки центра инструмента см. при описании указателя CN_TBLB.

CP_TLCM: Режим замера коррекции по длине вкл.

Режим замера коррекции по длине можно использовать в таких токарных каналах, на которые оборудованы замеры инструмента Режим замера коррекции по длине PLC соединяет с записыванием указателя CP_TLCM в 1.

Включение указателя можно совершиться **при отгибе рычага замера**, или под действием нажатия произвольной, определённой программой PLC **клавиши**.

Когда PLC включает бит CP_TLCM

– канал автоматически переключается в режим Jog: CN_JOG=1. Этот режим нельзя покинуть, пока состояние указателя CP_TLCM будет ПРАВДА.

– при этом движение подачи клавишей jog берётся интерполятором с параметра N0319 T Meas Feed. Одновременно может перемещаться только одна ось, то есть исключено одновременное перемещение нескольких осей.

– сторона указателя загружает экран Смещение, принадлежащий к каналу указателя, со включенным состоянием Замера коррекции.

CP_S2TS: Выбор замера по длине, относящегося к S2

Один канал может обращаться сигналами не более 2 замера инструмента. Два замера инструмента можно использовать например на токарных станках с контршпинделем.

Измерительная система на основании положения бита CP_S2TS PLC выбирает, какой из щупов должен использовать. Если значение CP_S2TS

=0: используется щуп, выбранный параметром N3012 Sensor Input of Tool Setter S1,

=1: используется щуп, выбранный параметром N3013 Sensor Input of Tool Setter S2.

CP_WPCM: Режим замера нулевой точки заготовки вкл.

Режим замера нулевой точки заготовки можно использовать только в токарных каналах и он обычно замеряет смещение лобовой поверхности заготовки в направлении Z. Режим замера нулевой точки заготовки PLC включает с записыванием указателья CP_WPCM в 1

Включение указателя может происходить при включении щупа, или под действием нажатия произвольной, определённой программой PLC клавиши.

Когда PLC включает бит CP_WPCM

– сторона NC автоматически переключается в режим Jog: CN_JOG=1. Этот режим нельзя покинуть, пока состояние указателя CP_WPCM будет ПРАВДА.

– при этом движение подачи с клавишей jog берётся интерполятором с параметра N0319 T Meas Feed. Одновременно может перемещаться только одна ось, то есть исключено одновременное перемещение нескольких осей.

– сторона указателя загружает экран Смещение, принадлежащий к каналу указателя, со включенным состоянием Замера нулевой точки.

CP_S2WS: Выбор замера нулевой точки, относящегося к S2

Один канал может обращаться сигналами не более 2 замера нулевой точки заготовки. Два замера нулевой точки заготовки можно использовать например на токарных станках с контршпинделем.

Измерительная система на основании положения бита CP_S2WS PLC выбирает, какой из щупов должен использовать. Если значение CP_S2WS

=0: используется щуп, выбранный параметром N3014 Sensor Input of Workpiece Setter S1,

=1: используется щуп, выбранный параметром N3015 Sensor Input of Workpiece Setter S2.

Последним пользуется тогда, если на станок с одной головкой с контршпинделем к каждому шпинделю оборудуют щуп замера нулевой точки.

CP_AHND: Запрос смещения нулевой точки маховичком

Под действием нажатия кнопки PLC запросит режим посредством включения указателя CP_AHND. Включением указателя NC a CN_AHND, NC подтверждает запрос. NC включает указатель CN_AHND только в Автоматическом режиме, или в режиме Ручного ввода данных (MDI). Подробное описание перемещения нулевой точки маховичком см. при описании указателя CN_AHND.

CP_WMCAXF: Запрос подачи, параллельной направлению поворота системы координат заготовки

В режиме Jog, Инкрементного jog, или Маховичка PLC может запросить, чтобы ручное перемещение осей происходило согласно повороту основания:

=0: запрос ручного перемещения по первоначальному направлению осей,

=1: запрос ручного перемещения по направлению повернутых осей.

CP_SGLBK: Режим по кадрам вкл.

Программа PLC при нажатии клавиши режима по кадрам, должна изменить значение указателя CP_SGLBK на противлположное.

В состоянии CP_SGLBK=1 оператор канала после выполнения каждого кадра принимает состояние стопа CN_STOP=1, то есть остановит обработку.

В случае использования станочной панели оператора NCT указатель MB_SGLBK даст состояние клавиши режима по кадрам. Лампой клавиши (ML_SGLBK) обращается указатель CP_SGLBK.

CP_CNDSP: Условный стоп вкл.

Программа PLC при нажатии клавиши условного стопа, должна изменить значение указателя CP_CNDSP на противлположное.

Если выполнение программы набегает на код M01, оператор канала просмотрит состояние указателя CP_CNDSP. В состоянии CP_CNDSP=1 он остановит обработ-

ку и принимает состояние стопа CN_STOP=1. В противом случае нет останова CP_CNDSP=0.

В случае использования станочной панели оператора NCT указатель MB_CNDSP даст состояние клавиши условного стопа. Лампой клавиши (ML_CNDSP) обращается указатель CP_CNDSP.

CP_TEST: Запрос режима теста

Когда оператор нажимает клавишу режима теста, программа PLC записывает указатель запроса режим теста CP_TEST в 1, тем самым запросит у оператора канала режима теста.

В состоянии CP_TEST=1 оператор канала просмотрит, что можно ли включить режима теста в канале. Если да, состоянием CN_TEST=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_TEST даст состояние клавиши теста.

CP_MLCK: Запрос режима станок закрыт

Когда оператор нажимает клавишу станок закрыт, программа PLC записывает указатель запроса режима станок закрыт CP_MLCK в 1, тем самым запросит у оператора канала режим станок закрыт.

В состоянии CP_MLCK=1 оператор канала просмотрит, что можно ли включить режим станок закрыт в канале. Если да, состоянием CN_MLCK=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_MLCK даст состояние клавиши станок закрыт.

CP_DRRUN: Запрос сухого бега

Когда оператор нажимает клавишу сухого бега, программа PLC записывает указатель запроса сухого бега CP_DRRUN в 1, тем самым запросит у оператора канала режим сухого бега.

В состоянии CP_DRRUN=1 оператор канала просмотрит, что можно ли включить режим сухой бег в канале. Если да, состоянием CN_DRRUN=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_DRRUN даст состояние клавиши сухого бега.

CP_BKRST: Запрос кадр снова

Когда оператор нажимает клавишу кадр снова, программа PLC записывает указатель запроса кадр снова CP_BKRST в 1, тем самым запросит у оператора канала повторный запуск кадра.

В состоянии CP_BKRST=1 оператор канала просмотрит, что можно ли включить повторный запуск кадра в канале. Если да, состоянием CN_BKRST=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_BKRST даст состояние клавиши кадр снова.

CP_BKRET: Запрос кадр назад

Когда оператор нажимает клавишу кадр назад, программа PLC записывает указатель запроса кадр назад CP_BKRET в 1, тем самым запросит у оператора канала возвращение к точке прерывания.

В состоянии CP_BKRET=1 оператор канала просмотрит, что можно ли включить в канале обратное позиционирование в кадр. Если да, состоянием CN_BKRET=1 CN_BKRST=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_BKRET даст состояние клавиши кадр назад.

CP_FLCK: Запрос функция закрыта

Когда оператор нажимает клавишу функция закрыта, программа PLC записывает указатель запроса функция закрыта CP_FLCK в 1, тем самым запросит у оператора канала закрытие выдачи функции.

В состоянии CP_FLCK=1 оператор канала просмотрит, что можно ли включить в канале режим функция закрыта. Если да, состоянием CN_FLCK=1 подтверждает запрос.

В случае использования станочной панели оператора NCT указатель MB_FLCK даст состояние клавиши функция закрыта.

CP_ABSOFF: Не используется

CP_CNDBK_1: Условный кадр 1 вкл.
CP_CNDBK_2: Условный кадр 2 вкл
CP_CNDBK_3: Условный кадр 3 вкл
CP_CNDBK_4: Условный кадр 4 вкл
CP_CNDBK_5: Условный кадр 5 вкл
CP_CNDBK_6: Условный кадр 6 вкл
CP_CNDBK_7: Условный кадр 7 вкл
CP_CNDBK_8: Условный кадр 8 вкл

По каналам можно установить 8 различных условий для пропускания кадра. Если в программе деталей кадр начинается с командой /n (n=1, ..., 8), оператор канала просмотрит перед выполнением кадра, что удовлетворяется ли n-ое условие, то есть будет ли 1 состояние указателя CP_CNDBK_n. Если

– CP_CNDBK_n=0 выполняет кадр,
– CP_CNDBK_n=1 не выполняет кадр, а переходит на следующий.

В случае использования станочной панели оператора NCT указатель MB_CNDBK даст состояние клавиши условного кадра. Лампой клавиши (ML_CNDBK) обращается указатель CP_CNDBK_n, выбранный к клавише програмистом PLC.

CP_FIN: Сигнал готов, все функции выполнены

Программа PLC состоянием CP_FIN=1 даст сообщение для оператора канала, что все функции выполнены. В состоянии CP_FIN=1 оператор канала

– в состоянии старта возьмёт следующий кадр из буфера и выполняет его,

– в случае выполнения по кадрам принимает состояние стопа, и ожидает старта,

– в конце программы, то есть при пустом буфере, удаляет состояние старта.

В PLC сигнал CP_FIN можно включить тогда, если сигнал подтверждения всех функций (CP_MnACK, CP_SACK, CP_TACK, CP_AUXnACK) будет 1 и со стороны программы PLC нет прочих причин блокировать выполнения программы деталей.

CP_RST: Не используется

CP_RSTREW: Не используется

CP_CSREQ: Не используется

CP_NOWT: Не надо дождаться кода сбора M от другого канала

В многоканальных управлениях понадобится, что выполнение программы деталей, имеющегося в одном канале, блокировать до тех пор, пока выполнение программы в одном или нескольких других каналах не достигает одну определённую точку. Это называется синхронизацией каналов, или дожиданием друг друга. Синхронизацию каналов можно выполнить кодами M. Параметрами N2201 Waiting M Codes Min и N2202 Waiting M Codes Max можно выделить диапазон для кода M.

Дожидание выполняется полностью с помощью NC, этим для PLC не нужно заниматься.

Если какую-то программу записали таким образом, чтобы ожидала кода M синхронизации другого канала, или каналов, однако программу желаем пускать в прогон сама по себе, без синхронизации к программам, работающим в других каналах, оператор канала

– в состоянии CP_NOWT=1 перешагает коды M синхронизации, то есть не дождётся других каналов.

CP_MINT: Вызов макрокоманды прерывания

CP_TMREN: Разрешение таймера свободного использования

В строке Таймера в окошке Время/счётчики управления можно вычитать время, показывающее значение таймера свободного использования в форме день/час/минута/секунда/мсек.

Таймер свободного использования PLC
– состоянием CP_TMREN=1 запускает,
– состоянием CP_TMREN=0 остановит.
Значение таймера можно изменить/вычитать
– с панели оператора,
– из программыы деталей,через макропеременную #3001.

CP_TSBD: Не используется

CP_EGBRRQ: Запрос выхвата инструмента в режиме EGB
Если оператор канала выполняет функцию нарезания зубьев G81.8 (электронный привод), указатель CN_EGBMD переключает в 1.
В состоянии 1 указателя CN_EGBMD, PLC может запросить выхватку инструмента с записыванием указателя CP_EGBRRQ в 1. При этом NC
– отводит инструмент в направлении и на расстояние, определённое параметром N1804 Retr. Dist.,
– затем в конце отвода оператор канала записшет в 1 указатель CN_RTRFIN.
После этого PLC должен выключить указатель CP_EGBRRQ=0.

CP_M1ACK: Функция 1. M выполнена
CP_M2ACK: Функция 2. M выполнена
CP_M3ACK: Функция 3. M выполнена
CP_M4ACK: Функция 4. M выполнена
CP_M5ACK: Функция 5. M выполнена
CP_M6ACK: Функция 6. M выполнена
CP_M7ACK: Функция 7. M выполнена
CP_M8ACK: Функция 8. M выполнена
В программе деталей в одном кадре можно писать не более 8 различных кодов M, то есть в одном кадре 8 различных кодов M может передать NC для PLC.
Если в программе деталей запрограммирована функция M, оператор канала запишет сигнал вписывания CN_MnSTB в 1 на период 1 цикла PLC. Под действием сигнала вписывания, на основании полученного кода PLC декодирует функцию.
Если полученный код покрывает существующую на станке функцию, то PLC
– CP_MnACK=0 удаляет сигнал подтверждения, соответствующий сигналу вписывания.
После выполнения функции PLC
– CP_MnACK=1 запишет сигнал подтверждения.
Оператор канала ведёт учёт на основании состояния указателей CP_MnACK, что какие функции M выполнены из кадра, находящегося под выполнением. Прерывая программу, затем снова запуская, оператор канала снова выдаёт для PLC не выполненные функции.

После включения программа PLC должна инициировать состояние сигналов подтверждения (CP_MnACK=1).

CP_SACK: Функция S выполнена

Если в программе деталей запрограммирована функция S, оператор канала запишет указатель CN_SSTB в 1 на период 1 цикла PLC. Под действием сигнала вписывания PLC

– CP_SACK=0 удаляет сигнал подтверждения функции S

После выполнения функции PLC

– CP_SACK=1 запишет сигнал подтверждения.

Оператор канала ведёт учёт на основании состояния указателей CP_SACK, что выполнена ли функция S в кадре, находящегося под выполнением. Прерывая программу, затем снова запуская, оператор канала снова выдаёт для PLC не выполненную функцию S.

После включения программа PLC должна инициировать состояние сигналов подтверждения (CP_SACK=1).

CP_TACK: Функция T выполнена

Если в программе деталей запрограммирована функция T, оператор канала запишет указатель CN_TSTB в 1 на период 1 цикла PLC. Под действием сигнала вписывания PLC

– CP_TACK=0 удаляет сигнал подтверждения функции T.

После выполнения функции PLC

– CP_TACK=1 запишет сигнал подтверждения.

Оператор канала ведёт учёт на основании состояния указателей CP_TACK что выполнена ли функция T в кадре, находящегося под выполнением. Прерывая программу, затем снова запуская, оператор канала снова выдаёт для PLC не выполненную функцию T.

После включения программа PLC должна инициировать состояние сигналов подтверждения (CP_TACK=1).

CP_AUX1ACK: Вспомогательная функция 1 выполнена

CP_AUX2ACK: Вспомогательная функция 2 выполнена

CP_AUX3ACK: Вспомогательная функция 3 выполнена

Если в программе деталей запрограммирована вспомогательная функция, оператор канала запишет указатель CN_AUXnSTB, относящий к вспомогательной функции в 1 на период 1 цикла PLC. Под действием сигнала вписывания PLC

– CP_AUXnACK=0 удаляет сигнал подтверждения соответствующей вспомогательной функции.

После выполнения функции PLC

– CP_AUXnACK=1 запишет соответствующей сигнал подтверждения.

Оператор канала ведёт учёт на основании состояния указателей CP_AUXnACK что выполнена ли данную спомогательную функция в кадре, находящегося под выполнением. Прерывая программу, затем снова запуская, оператор канала снова выдаёт для PLC не выполненные вспомогательные функции.
После включения программа PLC должна инициировать состояние сигналов подтверждения (CP_AUXnACK=1).

CP_GACK: Не используется

CP_HOLD: Стоп движения для каждой оси в канале

Если PLC переключает указатель в 1, оператор канала безусловно остановит движение всех осей, относящихся к каналу.
Он отличается от указателя CP_STOP в том, что пока в состоянии override и запрета стопа (G63) стоп является недействительным, CP_HOLD и при этом является действительным.
Из-за изложенных выше, при нарезании резьбы, или нарезании резьбы метчиком (G74, G84) при останове шпинделя надо использовать указатель CP_HOLD для останова подачи, далее подачу можно остановить остановом шпинделя через указатель CP_HOLD.

CP_CBFEN: Разрешение подачи кадра точения

Оператор канала запросит включение движения подачи от PLC включением указателя CN_CBFR=1.
Подача (движение) не запускается до тех пор, пока PLC не разрешает этого на указателе CP_CBFEN=1. Разрешение подачи можно обусловливать по разному со стороны программы PLC, например, вращением шпинделя на металлорежущих станках, включением лазера или пламени на машинах для обрезки и т.д.

CP_FHNDL: Режим подачи с маховичка вкл.

Если указатель CP_FHNDL включён под выполнением программы, интерполятор движется в данном канале не соответственно запрограммированной подаче (G94, или G95 F), а по импульсам, поступающим с маховичка. Скорость движения зависит
– от величины выбора приращения
– от скорости вращения маховичка.
С помощью указателей CP_HNDLSn надо выбирать, что каким маховичком пользовался канал.
Функцию можно использовать например на многоканальных станках для настройки программы для целей испытания соударения.

CP_OVC: Не используется

CP_HNDLS1: Подача с маховичка 1
CP_HNDLS2: Подача с маховичка 2
CP_HNDLS3: Подача с маховичка 3
CP_HNDLS4: Подача с маховичка 4

Если включен режим подачи с маховичка (CP_FHNDL=1), на приведенных выше указателях может оператор канала выделить, что из возможных 4-х маховичка каким лучше пользоваться для функции, с записыванием соответствующего указателя CP_HNDLSn.

CP_LIM1DIS: Запрет диапазона конечнего положения 1 с параметром
CP_LIM2DIS: Запрет диапазона конечнего положения 2 с параметром
CP_LIM3DIS: Запрет диапазона конечнего положения 3 с параметром

В управлении можно разрешать по осям 3 различных диапазонов конечнего положения, устанавливаемых параметром на битах #0 RE1, #1 RE2, #2 RE3 параметра N1000 Range Enable.

Программа PLC может выключить любой из 3-х различных устанавливаемых диапазонов конечнего положения на всех осях, относящихся к данному каналу, с записыванием в 1 соответствующего указателя CP_LIMnDIS.

☞ ***Внимание!*** *При этом управление не обращается конечными положениями с параметром!*

CP_LIMSEL: Выбор диапазона конечнего положения 1B с параметром для всех осей

Выбор конечнего положения с параметром по каналам из PLC для всех осей канала. Если

параметр N1001 StrkCont будет #4 ABA=1

в данном канале действительный указатель CP_LIMSEL PLC. Указатель CP_LIMSEL подскажет, что в 1-ом диапазоне какая группа параметров конечнего положения

CP_LIMSEL=0: 1A,
CP_LIMSEL=1: 1B

будет действительная для всех осей канала, в обоих направлениях.

Изменение выбора конечнего положения надо выполнять в стоячем положении осей.

☞ ***См. ещё:*** *указатели AP_LIMSELP и AP_LIMSELN.*

CP_CHOPON: Не используется

CP_SGOEN: Разрешение таблицы второй геометрической коррекции

Указатель дайствительен исключительно в токарных каналах..

На токарных станках, использующих развёрнутую раскладку инструментов, целесообразно вводить магазин со второй геометрической коррекцией. Магазин со второй геометрической коррекцией такой же длинный, как первый, и в ходе ссылки на геометрическую коррекцию, в случае удовлетворения определённых условий, значения, хранившиеся в магазине второй геометрической коррекции надо прибавить к значениям, хранившим в первом магазине.

В магазине второй геометрической коррекции положение держателей инструментов, занятое ими в станочной системе координат, можно задавать как X, Y, Z. Этим станет возможным задавать в магазине первой геометрической коррекции настоящую длину X, Y, Z инструмента, то есть в магазине первой геометрической коррекции можно задавать непосредственно значение вылета, измеренного на внешнем замере инструмента. Значит, значение коррекции, принятое во внимание:

коррекция = 1-я геометрическая коррекция + 2-я геометрическая коррекция + коррекция по износу

Если бит #4 SGC параметра N1414 Comp. Config on Lathes:

=0: магазин второй геометрической коррекции не существует, не высвечивается,

=1: магазин второй геометрической коррекции существует, высвечивается, и указатели PLC регулируют учёт значения коррекции.

Указатель CP_SGOEN разрешения таблицы второй геометрической коррекции в состоянии

=0 оператор канала не учитывает вторую геометрическую коррекцию,

=1 оператор канала учитывает вторую геометрическую коррекцию

при вызове коррекции кодом T.

См. ещё указатели CP_SGOX, CP_SGOY, CP_SGOZ.

CP_SGOEN можно использовать, если например, внутри токарного канала имеется и револьверная головка, и развёрнутая раскладка инструментов, и для инструментов в револьверной головке не надо вызывать вторую геометрическую коррекцию, а для инструментов развёрнутой раскладки - надо.

CP_SGOX: Учёт второй геометрической коррекции на оси X

CP_SGOY: Учёт второй геометрической коррекции на оси Y

CP_SGOZ: Учёт второй геометрической коррекции на оси Z

Если учёт второй геометрической коррекции разрешён в канале CP_SGOEN=1, на осях X, Y, Z канала можно разрешить их учёт отдельно:

CP_SGOX=1 на оси X,

CP_SGOY=1 на оси Y,

CP_SGOZ=1 на оси Z

разрешает учёт второй геометрической коррекции.

См. ещё указатели CP_OSGNX, CP_OSGNY, CP_OSGNZ.

CP_OSGNX: коррекция X с потивопложным знаком (xкоррекция=-xкоррекция)
CP_OSGNY: коррекция Y с потивопложным знаком (yкоррекция=-yкоррекция)
CP_OSGNZ: коррекция Z с потивопложным знаком (zкоррекция=-zкоррекция)

Если учёт второй геометрической коррекции разрешён CP_SGOEN=1 в канале, и на данной оси указателями CP_SGOX, CP_SGOY, CP_SGOZ, тогда направление учёта коррекции (знака) также можно регулировать по осям:
CP_OSGNX=1 на оси X с потивопложным знаком,
CP_OSGNY=1 на оси Y с потивопложным знаком,
CP_OSGNZ=1 на оси Z с потивопложным знаком
учитывается коррекция:
коррекция = −(1-я геометрическая коррекция + 2-я геометрическая коррекция + коррекция по износу)
Это можно использовать, если направление оборудованного на какой-то оси инструмента совпадает с положительным направлением оси.

CP_ROVLD: Выключение перекрытия кадров быстрого хода

Указатель можно использоватьтогда, когда на бите #0 ROL параметра N0407 Acc Contr включено перекрытие движений быстрого хода.
Если перекрытие надо выключить в некоторых случаях, программа PLC это может сделать со включением указателя CP_ROVLD=1.

CP_RLSOT3: Не используется

CP_1000: макропеременная #1000 (бит)
CP_1001: макропеременная #1001 (бит)
CP_1002: макропеременная #1002 (бит)
CP_1003: макропеременная #1003 (бит)
CP_1004: макропеременная #1004 (бит)
CP_1005: макропеременная #1005 (бит)
CP_1006: макропеременная #1006 (бит)
CP_1007: макропеременная #1007 (бит)
CP_1008: макропеременная #1008 (бит)
CP_1009: макропеременная #1009 (бит)
CP_1010: макропеременная #1010 (бит)
CP_1011: макропеременная #1011 (бит)
CP_1012: макропеременная #1012 (бит)
CP_1013: макропеременная #1013 (бит)
CP_1014: макропеременная #1014 (бит)
CP_1015: макропеременная #1015 (бит)
CP_1016: макропеременная #1016 (бит)
CP_1017: макропеременная #1017 (бит)

CP_1018: макропеременная #1018 (бит)
CP_1019: макропеременная #1019 (бит)
CP_1020: макропеременная #1020 (бит)
CP_1021: макропеременная #1021 (бит)
CP_1022: макропеременная #1022 (бит)
CP_1023: макропеременная #1023 (бит)
CP_1024: макропеременная #1024 (бит)
CP_1025: макропеременная #1025 (бит)
CP_1026: макропеременная #1026 (бит)
CP_1027: макропеременная #1027 (бит)
CP_1028: макропеременная #1028 (бит)
CP_1029: макропеременная #1029 (бит)
CP_1030: макропеременная #1030 (бит)
CP_1031: макропеременная #1031 (бит)

Пользователь может запросить из программы деталей бинарные указатели PLC через макропеременные. По каналам 32 указательных битов стоят на распоряжение пользователя для коммуникации с PLC: #1000, #1001, ..., #1031.

Например, записанная в программу деталей команда

```
IF #1025EQ1 GOTO30
```

переходит на кадр N30, когда указатель CP_1025 будет 1.

7.15.2 Переменные с двойным словом для канала

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
CN_1	Биты оператора канала DWORD1 переданные из NC в PLC	**CP_1**	Биты оператора канала DWORD1 переданные из PLC в NC
CN_2	Биты оператора канала DWORD2 переданные из NC в PLC	**CP_2**	Биты оператора канала DWORD2 переданные из PLC в NC
CN_3	Биты оператора канала DWORD3 переданные из NC в PLC	**CP_3**	Биты оператора канала DWORD3 переданные из PLC в NC
CN_1132	значение макропеременной #1132 (DWORD)	**CP_4**	Биты оператора канала DWORD4 переданные из PLC в NC
CN_M1C	Код функции 1 M (DWORD)	**CP_JOGFD**	Выбор подачи Jog из таблицы (DWORD)
CN_M2C	Код функции 2 M (DWORD)	**CP_SINP**	Номер шпинделя, выполняюще-го обработку (1,2...) (DWORD)
CN_M3C	Код функции 3 M (DWORD)	**CP_CSAX**	Не используется
CN_M4C	Код функции 4 M (DWORD)	**CP_POLYSL**	Номер шпинделя-аппликата то-чения многоугольника (1,2...) (DWORD)
CN_M5C	Код функции 5 M (DWORD)	**CP_ACTT**	Номер актуального инструмента (DWORD)
CN_M6C	Код функции 6 M (DWORD)	**CP_MGR1**	Код 1 M, желаемый высвечивать из PLC (DWORD)
CN_M7C	Код функции 7 M (DWORD)	**CP_MGR2**	Код 2 M, желаемый высвечивать из PLC (DWORD)
CN_M8C	Код функции 8 M (DWORD)	**CP_MGR3**	Код 3 M, желаемый высвечивать из PLC (DWORD)
CN_SC	Код функции S (DWORD)	**CP_MGR4**	Код 4 M, желаемый высвечивать из PLC (DWORD)
CN_SSEL	Номер шпинделя, к которому код S относится (DWORD)	**CP_MGR5**	Код 5 M, желаемый высвечивать из PLC (DWORD)
CN_RNGREQ	Номер диапазона, к которому код S относится (DWORD)	**CP_MGR6**	Код 6 M, желаемый высвечивать из PLC (DWORD)
CN_TC	Код функции T (DWORD)	**CP_MGR7**	Код 7 M, желаемый высвечивать из PLC (DWORD)
CN_AUX1C	Код вспомогательной функции 1 (DWORD)	**CP_MGR8**	Код 8 M, желаемый высвечивать из PLC (DWORD)
CN_AUX2C	Код вспомогательной функции 2	**CP_MGR9**	Код 9 M, желаемый высвечивать

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
	(DWORD)		из PLC (DWORD)
CN_AUX3C	Код вспомогательной функции 3 (DWORD)	**CP_MGR10**	Код 10 M, желаемый высвечивать из PLC (DWORD)
CN_CTSPN	Число оборотов шпинделя в случае G96 (DWORD)	**CP_MGR11**	Код 11 M, желаемый высвечивать из PLC (DWORD)
CN_NMAX	Значение максимального числа оборотов в случае G96 (G92 S_) (DWORD)	**CP_MGR12**	Код 12 M, желаемый высвечивать из PLC (DWORD)
CN_MGZNO	Номер магазина, относящегося к коду T (DWORD)	**CP_MGR13**	Код 13 M, желаемый высвечивать из PLC (DWORD)
CN_POTNO	Номер кармана, относящегося к коду T (DWORD)	**CP_MGR14**	Код 14 M, желаемый высвечивать из PLC (DWORD)
		CP_MGR15	Код 15 M, желаемый высвечивать из PLC (DWORD)
		CP_MGR16	Код 16 M, желаемый высвечивать из PLC (DWORD)
		CP_1032	Макропеременная #1032 (DWORD)
		CP_OFFSNO	Номер коррекции, используемой при замере инструмент (DWORD)

Переменные канала с двойным словом, идущие от NC в сторону PLC

CN_1132: значение макропеременной #1132 (DWORD)

Пользователь из программы деталей может вставить, далее удалить бинарные указатели PLC с присвоением значения для макропеременных. По каналам 32 указательный битов стоят на распоряжение пользователя для коммуникации с PLC: #1100, #1101, ..., #1131. Эти указатели PLC можно писать и с двойным словом через макропеременные #1132.

Команда

#1132=128

записшет указатель CN_1107=1.

PLC может таким же образом, с двойным словом запросить биты CN_1100, CN_1101, ... CN_1131 через переменные CN_1132.

CN_M1C: Код функции 1 M(DWORD)

CN_M2C: Код функции 2 M (DWORD)

CN_M3C: Код функции 3 M (DWORD)

CN_M4C: Код функции 4 M (DWORD)
CN_M5C: Код функции 5 M (DWORD)
CN_M6C: Код функции 6 M (DWORD)
CN_M7C: Код функции 7 M (DWORD)
CN_M8C: Код функции 8 M (DWORD)

В программе деталей в один кадр можно писать не более 8 различных кодов M, то есть в одном кадре 8 различных кодов M может передать NC для PLC.

Если в программу деталей запрограммирована и функция M, оператор канала

– записшет в регистр CN_MnC (n=1, 2, ..., 8) значение кода M

– затем сигнал вписывания CN_MnSTB запишет NC в 1 на период 1 цикла PLC.

Управление записывает 8 различных кодов M в 8 различных регистров ***CN_MnC*** (n=1, 2, ..., 8). К 8 регистрам передачи принадлежит 8 сигналов вписывания CN_MnSTB (n=1, 2, ...8). Под действием сигнала вписывания, на основании полученного кода PLC декодирует функцию.

Если полученный код покрывает существующую на станке функцию, то согласно сигналу вписывания PLC он установит в 0 сигнал подтверждения CP_MnACK. Сигнал подтверждения CP_MnACK разрешено включить в 1 только после полного выполнения функции.

CN_SC: Код функции S(DWORD)

Если в программу деталей запрограммирована функция S, оператор канала

– запишет в регистр CN_SC значение чисел оборотов, запрограммированное в функции S,

– запишет в регистр CN_SSEL номер ссылочного шпинделя (1...16)

– запишет в регистр CN_RNGREQ код диапазона, относящегося к запрограммированному числу оборотов (11...18),

– затем указатель CN_SSTB запишет в 1 на период 1 цикла PLC.

Под действием сигнала вписывания PLC декодирует коды, записанные в перечисленные выше регистры и установит в 0 сигнал подтверждения CP_SACK. Сигнал подтверждения CP_SACK разрешено включить в 1 только после полного выполнения функции.

Код, полученный в регистре CN_SC в ходе выполнения функции надо записать в регистр шпинделя SP_PRG, выделённого регистром CN_SSEL.

CN_SSEL: Номер шпинделя, на котоый относится код S (DWORD)

Если в программу деталей запрограммирована функция S, оператор канала

– запишет в регистр CN_SC значение чисел оборотов, запрограммированное в функции S,

– запишет в регистр CN_SC значение чисел оборотов, запрограммированное в функции S(1...16)

– запишет в регистр CN_RNGREQ код диапазона, относящегося к запрограммированному числу оборотов (11...18),

– затем указатель CN_SSTB запишет в 1 на период 1 цикла PLC.

Под действием сигнала вписывания PLC декодирует коды, записанные в перечисленные выше регистры и установит в 0 сигнал подтверждения CP_SACK. Сигнал подтверждения CP_SACK разрешено включить в 1 только после полного выполнения функции.

Оператор канала установит номер шпинделя, установленного в регистре CN_SSEL, согласно ссылке в программе деталей.

На шпидели из программы деталей можно ссылаться по адресу S, или в сулчае расширенного названия по адресу с символом S+2. В случае ссылки по расширенным адресам 2-й и 3-й символ адреса можно задавать параметрами N0605 Spindle Name2 и N0606 Spindle Name3. Первым символом является обязательно всегда S. Если последный символ является числом, при задании данных надо использовать знак =.

Если на станке имеется несколько шпинделей и не желаем пользоваться расширенными адресами, можно ссылаться на один данный шпиндель по адресу S и P, где по S задаётся число оборотов, а по P - номер шпинделя, к которому код S относится.

Оператор канала на основании запрограммированного кода Sxx, или Ss Pp установит, что на какой по номеру шпиндель была ссылка в программе деталей, и запишет в регистр CN_SSEL номер ссылочного шпинделя.

Пусть будет в случае использования расширенных адресов адрес 3-го шпинделя S3.

В случае запрограммирования

S3=1000

регистры передачи соответствующего канала получат следующие данные:

CN_SC=1000

CN_SSEL=3

В случае запрограммирования адреса S и P приведенная выше ссылка:

S1000 P3

В регистры передачи попадают те же значения.

Параметром N0604 Default Spindle можно выделить один шпиндель по каналам, на который можно ссылаться по адресу S. Если например, значение параметра 2, то на 2-й шпиндель можно ссылаться ипо адресу S2 и S. В обоих случаях оператор канала передаёт CN_SSEL=2.

Программа PLC выдаёт все команды на шпиндель, определённый в регистре ***CN_SSEL***

– команду M3, M4, M5, M19,

– команду замыкания контура позиции, установленную параметром N0823 M Code for Closing S Loop,

– команду для синхронизацию, относящуюся к шпинделю-аппликату (код M),

– код M, служащий для выбора шпинделя-аппликата точения многоугольника.

Одновременно можно вращать несколько шпинделей из программы в одном канале. При этом надо писать:

N10 S1=1000 M3

N20 S2=1500 M4

Если один шпиндель надо остановить из программы, надо писать:

N100 S1=0 M5

N110 S2=0 M5

CN_RNGREQ: Номер диапазона, относящегося к коду S (DWORD)

Если в программу деталей запрограммирована функция S, оператор канала

– запишет в регистр CN_SC значение чисел оборотов, запрограмимированное в функции S,

– запишет в регистр CN_SC значение чисел оборотов, запрограммированное в функции S (1...16)

– запишет в регистр CN_RNGREQ код диапазона, относящегося к запрограммированному числу оборотов (11...18),

– затем указатель CN_SSTB запишет в 1 на период 1 цикла PLC.

Под действием сигнала вписывания PLC декодирует коды, записанные в перечисленные выше регистры и установит в 0 сигнал подтверждения CP_SACK. Сигнал подтверждения CP_SACK разрешено включить в 1 только после полного выполнения функции.

Оператор канала декодирует, что в какой диапазон данного шпинделя попадает запрограммированное число оборотов:

– Если границы диапазонов перекрывают друг друга, как обычно бывает на токарных станках, регистр CN_RNGREQ нельзя использовать (и при этом производится декодирование).При этом изменение диапазона надо ставить на код M (обязательно на M11, M12, ..., M18) в программе PLC, чтобы пользователь мог решить, какой из диапазонов будетоптимальным с точки зрения точения.

– Если границы диапазонов не перекрывают друг друга, как это обычно бывает на фрезерных станках, можно использовать регистр CN_RNGREQ.

В этом случае изменение диапазона можно провести в программе PLC и без программирования отдельных функций M, не надо вводить функции M.

Оператор канала декодирует регистр ***CN_RNGREQ*** следующим образом:

Если значение параметра ***S≤N0650 R1 S Max***: тогда ***CN_RNGREQ=11***,

если значение параметра ***S≤N0651 R2 S Max***: тогда ***CN_RNGREQ=12***,

и так далее.

CN_TC: Код функции T (DWORD)

Если в программу деталей запрограммирована функция T, оператор канала

– запишет в регистр CN_TC номер инструмента, запрограммированного на функции T, в токарном канале ***отрезая номер коррекции***,вызванного по адресу T,

– запишет в регистр CN_MGZNO, что ссылочный инструмент в каком по номеру магазине находится,

– запишет в регистр CN_POTNO, что ссылочный инструмент в каком по номеру кармана данного магазина находится,
– просмотрит, что закончилась ли стойкость ссылочного инструмента, или группы инструментов, и согласно этому установит указатель CN_TLLE,
– затем указатель CN_TSTB запишет в 1 на период 1 цикла PLC.

Под действием сигнала вписывания PLC декодирует коды, записанные в перечисленные выше регистры и установит в 0 сигнал подтверждения CP_TACK. Сигнал подтверждения CP_TACK разрешено включить в 1 только после полного выполнения функции.

В регистр CN_TC попадает всегда номер ссылочного в программе деталей инструмента (номер типа), даже при использования таблицы оператора инструментов, когда в магазине может быть несколько инструментов тем же номером типа. При этом из регистров CN_MGZNO и CN_POTNO можно вычитать местонахождения загружаемого инструмента.

CN_AUX1C: Код вспомогательной функции 1(DWORD)
CN_AUX2C: Код вспомогательной функции 2 (DWORD)
CN_AUX3C: Код вспомогательной функции 3 (DWORD)

Параметром N1332+n Aux Fu Addrn (n=1..3) можно выделить три различных адресов из A, B, C, U, V, W. Если в программе деталей ссылаться на адрес заданных здесь трёх вспомогательных функций, оператор канала

– запишет в регистр ***CN_AUXnC*** (n=1...3) значение, запрограммированое по адресу вспомогательной функции (всегда целое число DWORD),
– затем указатель CN_AUXnSTB (n=1...3) запишет в 1 на период 1 цикла PLC.

Под действием сигнала вписывания PLC декодирует коды, записанные в перечисленные выше регистры и установит в 0 сигнал подтверждения CP_AUXnACK (n=1...3). Сигнал подтверждения CP_AUXnACK разрешено включить в 1 только после полного выполнения функции.

CN_CTSPN: Число оборотов шпинделя в случае G96 (DWORD)

Если канал находится в состоянии G96 (расчёт постоянной скорости резания),
– NC запишет в 1 указатель CN_CSURF,
– PLC расчитает по циклам и запишет в регистр CN_CTSPN число оборотов шпинделя, относящегося к мгновенным координатам,
– запишет максимальное число оборотов, запрограммированное командой G92 S в регистр CN_NMAX.

Задачей программы PLC является скопировать значение, полученное в состоянии CN_CSURF=1 в регистре ***CN_CTSPN***, в соответствующий регистр ***SP_PRG*** шпинделя (определённый регистром CP_SINP).

CN_NMAX: Максимальное число оборотов в случае G96 (G92 S_) (DWORD)

Если канал находится в состоянии G96 (расчёт постоянной скорости резания),

– NC запишет в 1 указатель CN_CSURF,
– PLC расчитает по циклам и запишет в регистр CN_CTSPN число оборотов шпинделя, относящегося к мгновенным координатам,
– запишет максимальное число оборотов, запрограммированное командой G92 S в регистр ***CN_NMAX***.

CN_MGZNO: Номер магазина, относящегося к коду T (DWORD)
CN_POTNO: Номер кармана, относящегося к коду T (DWORD)

Если в программу деталей запрограммирована функция T, оператор канала
– запишет в регистр CN_TC номер инструмента, запрограммированного на функции T, в токарном канале ***отрезая номер коррекции***, вызванного по адресу T,
– запишет в регистр CN_MGZNO, что ссылочный инструмент в каком по номеру магазине находится,
– запишет в регистр CN_POTNO, что ссылочный инструмент в каком по номеру кармана данного магазина находится,
– просмотрит, что закончилась ли стойкость ссылочного инструмента, или группы инструментов, и согласно этому установит указатель CN_TLLE,
– затем указатель CN_TSTB запишет в 1 на период 1 цикла PLC.

Под действием сигнала вписывания PLC декодирует коды, записанные в перечисленные выше регистры и установит в 0 сигнал подтверждения CP_TACK. Сигнал подтверждения CP_TACK разрешено включить в 1 только после полного выполнения функции.

Если установлен бит #0 TMU=1 параметра N2900 Tool M. Config, на станке работает таблица оператора инструментов. При этом надо обращаться регистрами CN_MGZNO и CN_POTNO!

Функцию обращения инструментами надо задействовать в управлении в следующих случаях:
– для инструментов желаем проводить просмотр стойкости,
– для инструментов, хранившихся в магазине желаем ссылаться из программы деталей не по коду места, а на основании кода инструмента,
– если использованная на станке смена инструментов требует random обращения магазином.

В таблицу оператора инструментов можно записать код T, то есть номер типа использованных в управлении инструментов, на который ссылаемся в программе деталей.

Если в обращении стойкостью учавствует несколько инструментов, обеспечивающих одинакову операцию, то инструменты, выполняющие одинакову операцию, надо записать в таблицу с тем же номером типа. В программе деталей по адресу T надо ссылаться на номер типа, и оператор инструментов решает, что какой по номеру типа инструмент возьмёт. Обычно берётся инструмент с наименьшей, но ещё не

законченной стойкостью. Таблица оператора инструментов является глобальной, то есть общей для всех каналов.

Оператор инструментов обращается не более 4 магазинами. Помимо этого, параметром N2922 Spindle Magazines можно определить каждый шпиндель, определённый на станке, в который может попасть инструмент, в качестве магазина шпинделя. Рядом с магазином шпинделя, по предыдущему параметру можно определить резервный магазин для хранения инструментов, находящихся в рычаге.

В регистр CN_TC попадает всегда номер ссылочного в программе дателей инструмента (номер типа), даже в случае использования таблицы оператора инструментов, когда в магазине может быть несколько инструментов с тем же номером типа.

При использовании таблицы оператора инструментов, оператор инструментов запишет рядом со ссылочным номером типа (регистр CN_TC)

в регистр ***CN_MGZNO***, что ссылочный инструмент в каком по номеру магазине находится, далее

в регистр ***CN_POTNO***, что инструмент в каком по номеру кармана данного магазина находится.

Исстолкование регистра ***CN_MGZN***:

CN_MGZN=0 означает: ссылочного инструмента нет ни в одном магазине,

CN_MGZN=1, 2, 3, 4 означает:инструмент находится в магазине 1., 2., 3., 4.,

CN_MGZN=10, 20, 30, ... означает: инструмент находится в шпинделе 1., 2., 3., ...,

CN_MGZN=11, 21, 31, ... означает: инструмент находится в резервном магазине (в рычаге) 1., 2., 3., ... относящемся к шпинделю.

Исстолкование регистра ***CN_POTNO***:

если CN_MGZN=0, не имеет значения,

если CN_MGZN=1, 2, 3, 4, тогда CN_POTNO=1, 2, 3, ... это есть значение, установленное по длине данного магазина в группе параметров 30 Tool Management,

если CN_MGZN=10, 20, 30, ...,тогда CN_POTNO=1, (1., 2., 3., ... находится в шпинделе)

если CN_MGZN=11, 21, 31, ..., тогда CN_POTNO=1. (1, 2, 3, ... находится в рычаге, относящемся к шпинделю).

Переменные канала с двойным словом, идущие от PLC в сторону NC

CP_JOGFD: Выбор подачи Jog из таблицы (DWORD)

Если параметром N0316 Jog F Contr выбрана опция JFT (значение параметра 1), в режиме jog оператор канала устанавливает значение подачи на основании значения регистра CP_JOGFD PLC, по экспоненциальной зависимости. Подача выдаётся всегда в размерности 1/мин.

Значение регистра CP_JOGFD надо привязать к положению включателя override. В случае станочной панели оператора NCT к значению регистра MKFOVER.

Таблица ниже показывает от 0 до 15 значение подачи, в зависимости от значения регистра CP_JOGFD, если установлен параметр JFT. Ряд продолжается при больших значениях CP_JOGFD.

CP_JOGFD	G21 мм/мин	G20 дюйм/мин	ось круга °/мин
0	0	0	0
1	2	0.08	0.4
2	3.2	0.12	0.64
3	5	0.2	1
4	7.9	0.3	1.58
5	12.6	0.5	2.52
6	20	0.8	4
7	32	1.2	6.4
8	50	2	10
9	79	3	15.8
10	126	5	25.2
11	200	8	40
12	320	12	64
13	500	20	100
14	790	30	158
15	1260	50	252

CP_SINP: Номер шпинделя, выполняющего обработку (1,2...) (DWORD)

Программа PLC должна выделить по каналам номер активных шпинделей в регистре CP_SINP: ***CP_SINP***=1, 2, 3, ...

Оператор канала берёт импульсы датчика с датчика активного шпинделя

– при подаче за оборот G95,

– при нарезании резьбы G33,

– в случае нарезании резьбы метчиком с жёстким стержнем G84.2, G84.3 ожидает сигнала контур замкнут SN_LPCLSD с активного шпинделя, и сверлит с активным шпинделем,

– в случае точения многоугольника G51.2 шпинделем-мастером является всегда активный шпиндель, далее

Программа PLC всегда на основании вращения активного шпинделя, заданного в регистре CP_SINP

– разрешает подачу в кадре точения с указателем CP_CBFEN,

– при расчёте постоянной скорости резания G96 в состоянии CN_CSURF=1 скопирует полученное в регистре CN_CTSPN значение, в регистр ***SP_PRG*** активного шпинделя, определённого соответствующим регистром ***CP_SINP***.

Тот же самый шпиндель можно назначить к нескольким каналам. Например, на токарном станке с осью 2x2 к обоим каналам назначен шпиндель левой стороны, на той же заготовке может работать обе системы суппортов с подачей за оборот. Про-

грамист программы деталей может решить, что управление на основании какого канала оси X принимало во внимание расчёт постоянной скорости резания (код G96 запрограммирует в нужном канале).

Строитель станка решает, что какой шпиндель будет активным и как надо этого выбирать, Выделение активного шпинделя можно задавать кодами M. Например:

Код M31 активный S1,

Код M32 активный S2 и т.д.

☞ ***Внимание!*** *Функцию M для выделения активного щпинделя надо выделить для функции дожидания (для разгрузки буфера), так как выполнение кода влияет на подготовку кадра.*

CP_CSAX: Не используется

CP_POLYSL: Номер щпинделя-аппликата точения многоугольника (1,2...) (DWORD)

В регистр CP_POLYSL надо записать по каналам номер того шпинделя, который будет щпинделем-аппликатом точения многоугольника, то есть в котором инструмент вращается.

Шпиндель-мастер точения многоугольника выделяется регистром CP_SINP.

Перед выдачей команды точения многоугольника G51.2 функцией M должна программа PLC

– выбирать щпиндель-аппликат, заданием регистра CP_POLYSL

– надо разрешать оператора канала SP_SEN=1

– надо включить привод шпинделя.

CP_ACTT: Номер актуального инструмента (DWORD)

Если бит #7 параметра TSP N2901 будет 0, индикация номера актуального инструмента в окошке FST происходит из регистра CP_ACTT, индексированного по каналам.

Программа PLC запишет сюда номер загружаемого инструмента.

CP_MGR1: Код 1 M, желаемый высвечивать из PLC (DWORD)
CP_MGR2: Код 2 M, желаемый высвечивать из PLC (DWORD)
CP_MGR3: Код 3 M, желаемый высвечивать из PLC (DWORD)
CP_MGR4: Код 4 M, желаемый высвечивать из PLC (DWORD)
CP_MGR5: Код 5 M, желаемый высвечивать из PLC (DWORD)
CP_MGR6: Код 6 M, желаемый высвечивать из PLC (DWORD)
CP_MGR7: Код 7 M, желаемый высвечивать из PLC (DWORD)
CP_MGR8: Код 8 M, желаемый высвечивать из PLC (DWORD)
CP_MGR9: Код 9 M, желаемый высвечивать из PLC (DWORD)
CP_MGR10: Код 10 M, желаемый высвечивать из PLC (DWORD)
CP_MGR11: Код 11 M, желаемый высвечивать из PLC (DWORD)

CP_MGR12: Код 12 M, желаемый высвечивать из PLC (DWORD)
CP_MGR13: Код 13 M, желаемый высвечивать из PLC (DWORD)
CP_MGR14: Код 14 M, желаемый высвечивать из PLC (DWORD)
CP_MGR15: Код 15 M, желаемый высвечивать из PLC (DWORD)
CP_MGR16: Код 16 M, желаемый высвечивать из PLC (DWORD)

Параметрами N1341 M GR Low 1, ..., N1356 M GR Low 16 и N1357 M GR High 1, ..., N1372 M GR High 16 можно выделить 16 различных групп кодов M. На параметр, с обозначением Low надо записать самый меньший номер кода группы, а на параметр, с обозначением High - самый больший.

Параметры надо задавать таким образом, чтобы выделённые полоса кодов M были кодами состояния станка, исключающими друг друга.

Блок зачитания кадров фильтрует коды M таким образом, чтобы из кодов M, принадлежащих в одну группу, в данном кадре может быть только один, иначе выводится сообщение *Противоречивые коды M*.

Значения, установленные параметром, принимаются во внимание управлением и в ходе поиска кадров, при сборе кодов M. Из кодов M, относящихся в одну группу, он собирёт только последним заданный код M. Пересмотрим следующие коды M:

M51: зажим патрона на шпинделе S1
M52: разжим патрона на шпинделе S1
M53: разжим патрона при вращении шпинделя S1

Установим параметры следующим образом:

N1341 M GR Low 1=51
N1357 M GR High 1=53

Состояние зажима:

M54: внутренний зажим S1
M55: внешний зажим S1

Параметры:

N1342 M GR Low 2=54
N1358 M GR High 2=55

В ходе поиска кадра будут собраны и выполнены оператором из кодов M51, M52, M53 только запрограммированные последними. То же самое относится и к группе M54, M55. Из соответствующих состояний станка выписывается в регистрах CP_MGR1, 2 всегда только один, например, содержание регистров после зажима патрона:

CP_MGR1=51 – патрон зажат на шпинделе S1 и
CP_MGR2=54 – внутренний зажим на шпинделе S1.

Пара M GR Low n, M GR High n соответствует регистру CP_MGRn. На параметры надо записать начальный и конечный номера тех групп кодов M, которые программа PLC

– запишет в регистры ***CP_MGR1, ..., CP_MGR16***, и которые высвечиваются в окошке кодов M.

CP_1032: Макропеременная #1032 (DWORD)

Пользователь из программы деталей через макропеременные может запросить бинарные указатели PLC. По каналам 32 указательных битов стоят на распоряжение пользователя для коммуникации с PLC: #1000, #1001, ..., #1031. Эти переменные можно запросить и с двойным словом через макропеременную #1032.

Например, команда

IF #1032EQ16 GOTO30,

записанная в программу деталей, переходит на кадр N30 тогда, когда #1004=1 (указатель CP_1004 будет 1), а остальные 0.

CP_OFFSNO: Номер коррекции, используемая при замере инструмента (DWORD)

В ходе ручного замера щупом нулевой точки коррекции по длине/заготовки, регистр коррекции по длине, куда попадает результат измерения щупом/откуда берётся коррекция для расчёта смещения, если значение #0 ONS бита параметра N3016 Tool/WP Setter Config:

=0: оператор вручную выбирает на панели оператора,

=1: PLC выбирает в регистре CP_OFFSNO.

Если например оператор хочет замерить инструменты таким образом, чтобы номер коррекции совпадал с номером инструмента (например T1212), программа PLC запишет в регистр CP_OFFSNO номер актуального инструмента. Благодаря этому после смены инструмента окошко замера перескочит на коррекцию, совпадающую с номером актуального инструмента, и не нужно вручную выбирать замеряемую коррекцию.

7.15.3 Переменные с плавающей запятой для канала

Вводы		Выводы	
Символ	**Описание**	**Символ**	**Описание**
CN_1133	Значение макропеременной #1133 (double)	**CP_INC**	Величина jog с приращением и шага маховичка (double)
CN_1134	Значение макропеременной #1134 (double)	**CP_FOVER**	Override подачи: если =1: 100% (double)
CN_1135	Значение макропеременной #1135 (double)	**CP_ROVER**	Override быстрого хода: если =1: 100% (double)
CN_GC	Неиспользуется	**CP_COV**	Неиспользуется
CN_G1DAT	Неиспользуется	**CP_1033**	Макропеременная #1033 (double)
CN_G2DAT	Неиспользуется	**CP_1034**	Макропеременная #1034 (double)
CN_G3DAT	Неиспользуется	**CP_1035**	Макропеременная #1035 (double)

Переменные канала с плавающей запятой от NC в сторону PLC

CN_1133: Значение макропеременной #1133(double)
CN_1134: Значение макропеременной #1134(double)
CN_1135: Значение макропеременной #1135(double)

Пользователь из программы деталей передачей значений для макропеременных #1133, #1134, #1135 может передавать данные с плавающей запятой по каналам для PLC. Эти данные программ PLC может непосредственно вычитать из регистров CN_1133, CN_1134, CN_1135.

Записанная в программу деталей команда

#1134=167.832

запишет в регистр CN_1134 PLC число 167.832.

CN_GC: Не используется

CN_G1DAT: Не используется

CN_G2DAT: Не используется

CN_G3DAT: Не используется

Переменные канала с плавающей запятой от PLC в сторону NC

CP_INC: Величина jog с приращением и шага маховичка (double)

В режимах jog с приращением (CN_INCR=1), маховичка (CN_HNDL=1), или в случае управляемой маховичком подачи (CP_FHNDL=1) в регистр CP_INC надо записать по каналам величину шага с плавающей запятой.

Выбор величины шага в случае использования станочной панели оператора NCT происходит с клавиши

MB_I1, MB_I10, MB_I100, MB_I1000.

Выбор величины шага в случае вынесенного маховичка NCT происходит с указателей

HB_I1, HB_I10, HB_I100, HB_I1000.

В регистр CP_INC надо записать всегда число с плавающей запятой. Размерность записаемого числа определяется битом #0 IND параметра N0104 Unit of Measure, показывающий систему мер вывода. (Ту систему мер, в которой происходит измерение позиции.)

Если CP_INC=0.01 и

IND=0: CP_INC=0.01 означает 0.01 мм

IND=1: CP_INC=0.01 означает 0.01 дюйм

В зависимости от применённой в управлении системы мер ввода, данные нужно конвертировать. К конверсии можно использовать указатель CN_INCH. Если указатель

CN_INCH= 0: действует метрический ввод G21

CN_INCH=1: действует дюймовый ввод G20.

Например, нажата клавиша MB_I10 (0.01):

– в случае ***IND=0*** (метрического измерения) ***CN_INCH=0*** (G21 метрического ввода данных)

CP_INC=0.01 (величина шага ***0.01 мм***) надо писать

– в случае ***IND=0*** (метрического измерения) ***CN_INCH=1*** (G20 дюймого ввода данных)

CP_INC=0.0254 (величина шага 0.0254 мм = 0.0254/25.4 = ***0.01 дюйм***) надо писать.

CP_FOVER: Override подачи: если =1: 100% (double)

Значение override подачи по каналам. Число с плавающей запятой. Если например

CP_FOVER=1.0 означает 100%

CP_FOVER=1.427 означает 142.7%

В случае использования станочной панели оператора NCT положение включателя override надо взять из регистра MKFOVER.

Верхний предел override подачи надо установить в программе PLC!

☞ ***Внимание!*** *MKFOVER целое число типа DWORD, а CP_FOVER double с плавающей запятой, значит, установка override требует конверсию из числа с фиксированной*

запятой в число с плавающей запятой (команда FLT). CP_FOVER индексируется по 2!

CP_ROVER: Override быстрого хода: если =1: 100% (double)

Значение override быстрого хода по каналам. Число с плавающей запятой. Если например

CP_ROVER=0.272 означает 27.2%

CP_ROVER=1.0 означает 100%

В случае использования станочной панели оператора NCT значение override быстрого хода можно взять и из положение включателя override подачи и из регистра MKFOVER.

Верхний предел override быстрого хода 100%, выше этого оператор канала не допускает!

☞ ***Внимание!*** *MKFOVER целое число типа DWORD, а CP_ROVER double с плавающей запятой, значит, установка override требует конверсию из числа с фиксированной запятой в число с плавающей запятой (команда FLT). CP_ROVER индексируется по 2!*

CP_COV: Не используется

CP_1033: Макропеременная #1033 (double)
CP_1034: Макропеременная #1034 (double)
CP_1035: Макропеременная #1035 (double)

Программа PLC писанием в регистры CP_1033, CP_1034, CP_1035 может передавать данные с плавающей запятой непосредственно для программы деталей. Программа деталей может использовать данные через макропеременные #1033, #1034, #1035.

Записанная в программу деталей команда

#100=#1134

записанное в регистр CP_1134 PLC число с плавающей запятой запишет в макропеременную #100.

Алфавитный указатель: