



Kawasaki Robot Controller D Series

AS ЯЗЫК ПРОГРАММИРОВАНИЯ РУКОВОДСТВО

Robot

Kawasaki Heavy Industries, Ltd.

ПРЕДИСЛОВИЕ

Это руководство описывает AS* язык, используемый в контроллере робота Kawasaki D серии. Цель этого руководства состоит в том, чтобы обеспечить детальную информацию относительно иерархической структуры AS системы, основных использований, типов данных, управления траекторией робота и всех команд/инструкций, для того чтобы позволить эффективное использование AS системы. Процедуры работы с роботом здесь не включены, так что обратитесь к «Руководство Оператора» для информации. Это руководство должно читаться с внимательным обзором, связанных с ним, нижеупомянутых руководств. Только когда содержание всех руководств полностью прочитано и понято, робот может использоваться.

1. Руководство по обеспечению мер безопасности
2. Руководство по установке и подключению робота
3. Руководство по установке и подключению контроллера
4. Руководство по управлению внешними сигналами ввода - вывода (для того, чтобы соединяться с периферийными устройствами)
5. Руководство по ремонту и обслуживанию

Содержание этого руководства описано при условии, что установка и подключение робота сделаны в соответствии с вышеупомянутыми, внесенными в список, руководствами.

Данное руководство обеспечивает объем информации, в соответствии со стандартами, определяющими операционные методы для робота Kawasaki. Однако, не каждое возможное действие, условие или ситуация, которую нужно избежать, могут быть описаны полностью. Поэтому, любые необъясненные вопросы или проблемы, возникающие в течение работы робота или операционной системы, необходимо задать, войдя в контакт с ф. Kawasaki группа Робототехнические Системы. Обратитесь, для более быстрого контакта, в ближайший филиал Kawasaki, адреса филиалов внесены в список в конце этого руководства.

Это руководство не устанавливает гарантию для систем, в которых используется робот. Соответственно, Kawasaki не ответственен за любые несчастные случаи, убытки, и/или проблемы, в части, касающейся промышленных прав собственности в результате использования системы.

1. Рекомендуется, чтобы весь персонал, назначенный для запуска, рабочего использования и обслуживания системы прошел необходимый курс обучения, проводимый Kawasaki, перед принятием ими производственных обязанностей.
2. Kawasaki резервирует права изменить (заменить), пересматривать, или модернизировать это руководство без предшествующего уведомления.
3. Это руководство не может, полностью или частично, быть переиздано или скопировано без предшествующего письменного согласия Kawasaki.
4. Бережно храните это руководство и держите его в месте доступном для использования в любое время. В случае если руководство потеряно или повреждено, входите в контакт с вашим агентом Kawasaki.
5. Хотя это руководство было подготовлено, как можно более полным и точным, авторы приносят извинения, если какая-либо информация будет найдена неполной или ошибочной.

Все права зарегистрированы. Охраняется авторским правом 2002 Kawasaki Heavy Industries Ltd.

Пункты, которые требуют специального внимания в этом руководстве, определяются следующими символами.

Гарантией корректной и безопасной работы робота и предотвращения физических травм или материального ущерба будет соблюдение правил безопасности, в том числе предупреждений, данных в полях с этими символами.



Отказ следовать обозначенным предупреждениям может привести к неизбежной травме или смерти.



Отказ следовать обозначенным предупреждениям возможно может привести к травме или смерти.



Отказ следовать обозначенным предупреждениям может привести к физической травме или механическому повреждению

[Примечание]

Обозначает предосторожности относительно технических характеристик робота, управления, обучения, оперирования и обслуживания.

СОДЕРЖАНИЕ

Предисловие	2
1.0 Общее представление об AS системе	6
1.1 Краткий обзор	7
1.2 Характеристики AS системы	7
1.3 Конфигурация AS системы	8
2.0 AS система	10
2.1 Состояния AS системы	11
2.2 Системные переключатели AS системы	12
2.3 AS системные установки	13
2.4 Операции ввода и вывода	14
2.5 Установка программного обеспечения терминала	15
2.6 Работы с персональным компьютером	15
3.0 Информационные выражения в AS языке	23
3.1 Система обозначений и соглашений	24
3.2 Информация о позиции, числовая информация, символьная информация	25
3.3 Переменные	31
3.4 Имена переменных	33
3.5 Задание переменных позиции	33
3.6 Задание реальных переменных	38
3.7 Задание строковых переменных	39
3.8 Числовые выражения	39
3.9 Строковые выражения	41
4.0 AS программы	43
4.1 Типы AS программ	44
4.2 Создание и редактирование программ	45
4.3 Выполнение программ	49
4.4 Процесс выполнения программы	51
4.5 Движение робота	53
5.0 Мониторные команды	62
5.1 Команды редактирования	63
5.2 Команды управления программой и данными	72
5.3 Команды сохранения программ и данных	80
5.4 Команды управления программой	86
5.5 Команды позиционной информации	93
5.6 Команды управления системой	98
5.7 Команды двоичных сигналов	127
5.8 Команды отображения сообщений	135
6.0 Программные инструкции	140
6.1 Инструкции движения	141
6.2 Инструкции управления скоростью и точностью	151
6.3 Инструкции управления фиксаторами	158
6.4 Инструкции конфигурации робота	163

6.5 Инструкции управления программой	166
6.6 Инструкции построения программных структур	175
6.7 Инструкции двоичных сигналов	184
6.8 Инструкции управления сообщениями	198
6.9 Инструкции позиционной информации	203
6.10 Инструкции управления программой и данными	214
7.0 AS системные переключатели	216
8.0 Операторы	229
8.1 Арифметические операторы	230
8.2 Относительные операторы	231
8.3 Логические операторы	232
8.4 Бинарные операторы	233
8.5 Операторы векторной алгебры	235
8.6 Строковые операторы	237
9.0 Функции	238
9.1 Функции реальных значений	239
9.2 Функции значений позиции	253
9.3 Математические функции	264
9.4 Строковые функции	266
10.0 Программы управления процессом	274
11.0 Примеры программ	280
11.1 Первоначальные установки для программ	281
11.2 Паллетирование	282
11.3 Внешнее взаимодействие	284
11.4 Преобразование инструментальной системы координат	287
11.5 Относительные позиции	290
11.6 Использование относительных позиций с функцией FRAME	294
11.7 Установка конфигурации робота	296
Приложение 1 Коды ошибок	299
Приложение 2 Листинг AS языка	317
Приложение 3 ASCII коды	331
Приложение 4 Ограничение сигналов	334
Приложение 5 Углы Эйлера O,A,T	335

1.0 ОБЩЕЕ ПРЕДСТАВЛЕНИЕ ОБ AS СИСТЕМЕ

Роботы KAWASAKI управляются программной системой, называемой AS. Эта глава описывает общие сведения об AS системе.

- 1.1 Краткий обзор
- 1.2 Характеристики AS системы
- 1.3 Конфигурация AS системы

1.1 КРАТКИЙ ОБЗОР

В AS системе вы можете задавать команды или выполнять программы, используя AS язык программирования. AS система записана в энергонезависимой памяти контроллера (ОЗУ). Когда управляющее питание включается, AS система стартует и ожидает ввода команд для выполнения.

AS система управляет роботом в соответствии с данными командами и программами. Во время выполнения программы могут использоваться некоторые типы функций такие как, отображение статус состояния системы, позиционное текущее положение робота, записанные данные во внешних устройствах памяти, написание и редактирование программ.

1.2 ХАРАКТЕРИСТИКИ AS СИСТЕМЫ

В AS системе роботы управляются и функционируют, основываясь на программе, которая создается до выполнения действий и описывает последовательность действий, необходимых для решения поставленной задачи. (Обучение методом воспроизведения).

AS язык программирования разделяется на два вида: мониторные команды и программные инструкции

Мониторные команды используются для написания, редактирования, выполнения программ и единичных команд. Они вводятся после знака (>), появляющегося в начале строки и выполняются после нажатия клавиши (Enter) немедленно. Некоторые мониторные команды используются внутри программы, как программные инструкции.

Программные инструкции используются для создания последовательности движения робота, для контроля и управления внешними сигналами и т.д. в программах. Программа есть совокупность программных инструкций. В данном руководстве мониторные команды называются команды, программные инструкции называются инструкциями.

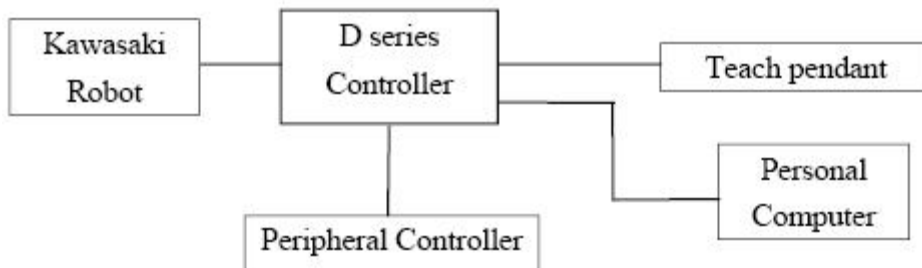
AS уникальна по следующим причинам:

1. Робот может двигаться непрерывно вдоль заданной траектории (СР движение: непрерывное движение)
2. Две координатные системы, базовая система координат и инструментальная система координат, обеспечивающие более точное управление перемещениями робота
3. Координаты могут быть смещены и повернуты в соответствии с изменением положения детали
4. При обучении позициям или повторения действий робот может идти по линейной траектории, сохраняя положение инструмента неизменным
5. Программы могут наименовываться свободно и запоминаться без ограничений
6. Каждая операционная единица может быть задана как программа, и эти программы могут быть скомбинированы в один комплекс (подпрограмма)
7. При помощи контролирования сигналов программы могут быть прерваны, и перейти к выполнению других программ с временной приостановкой текущего движения, когда происходит вход внешнего сигнала (прерывание)

8. Программы управления производственным процессом (РС программы) могут выполняться одновременно с управляющими программами робота
9. Программы и данные позиций могут отображаться на дисплее и записываться на внешний носитель памяти, например на PC карту
10. Программирование может быть осуществлено при помощи персонального компьютера, подключенного к контроллеру (автономное программирование)

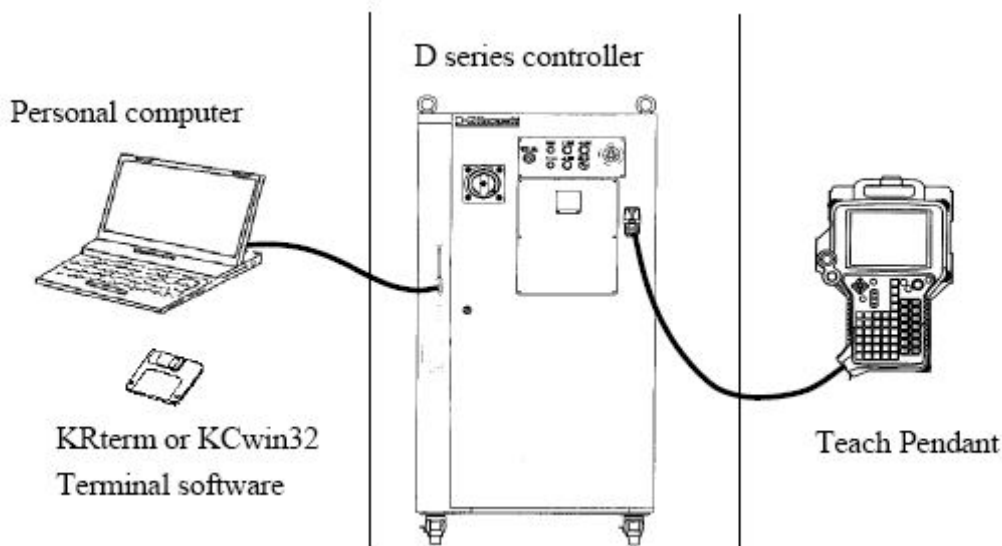
1.3 КОНФИГУРАЦИЯ AS СИСТЕМЫ

Kawasaki контроллер робота D серии состоит из следующих компонентов:



При подключении персонального компьютера, с загруженным программным обеспечением терминала KRterm или KCwin32 в контроллер D серии, следующие операции могут быть сделаны:

- запись AS команд
- сохранение и загрузка в из персонального компьютера



Персональный компьютер

- ввод AS команд
- создание и редактирование AS программ
- сохранение/загрузка программ

Контроллер

- ежедневные операции

Пульт ручного управления

- **выбор программы**
- **отображение программных имен и шагов**
- **ручное управление роботом**
- **управление сигналами**
- **установка условий повторения**
- **обучение данных позиции (координат)**
- **обучение вспомогательных данных (блочное программирование)**

(ПРИМЕЧАНИЕ)

Управляющее программное обеспечение для персонального компьютера работает с 95/98/Me/2000/XP.

Пожалуйста, подготовьте соответствующую OS (операционную систему)

.

2.0 AS СИСТЕМА

Эта глава описывает состояние AS системы, системные переключатели, системные установки.

2.1 Состояние AS системы

2.2 Системные переключатели AS системы

2.3 AS системные установки

2.4 Операции ввода, вывода

2.5 Установка программного обеспечения терминала

2.6 Работа с персональным компьютером

2.1 СОСТОЯНИЕ AS СИСТЕМЫ

AS система совмещает следующие три режима

1. Мониторный режим

Это основной режим в AS системе. В данном режиме выполняются команды монитора. Редактирование или режим воспроизведения доступны только в мониторном режиме.

2. Режим редактирования

Этот режим позволяет вам создавать новые программы или редактировать уже созданные программы. Команды редактора выполняются только в этом режиме.

3. Режим воспроизведения

Система находится в режиме воспроизведения в течение программного выполнения. Вычисления для управления движением робота и команды, вводимые с терминала, обрабатываются и выполняются в течение свободного таймерного цикла. Некоторые мониторные команды не могут быть выполнены в данном режиме.

2.2 СИСТЕМНЫЕ ПЕРЕКЛЮЧАТЕЛИ AS СИСТЕМЫ

Следующие системные переключатели могут быть установлены в AS системе, используя мониторинговую команду SWITCH. Состояние и условия установки для каждого переключателя могут быть проверены и изменены с терминала.

1. CHECK.HOLD

Определяет действенность команд EXECUTE, DO, STEP, MSTEP, CONTINUE, когда переключатель **HOLD/RUN** на операционной панели находится в положение HOLD.

2. CP

Разрешает или запрещает непрерывное движение от точки к точке. Когда системный переключатель в положение ON, робот производит плавные переходы между сегментами движения. Когда переключатель в положение OFF, робот замедляется и останавливается в конце каждого сегмента движения.

3. CYCLE.STOP

Задаёт включение или выключение **CYCLE.START** на операционной панели, когда вводится внешний сигнал для прерывания движения робота.

4. MESSAGES

Разрешает или запрещает вывод сообщений на дисплей в соответствии с командами PRINT или TYPE.

5. OX.PREOUT

Устанавливает синхронизацию для установки ON/OFF OX сигналов в инструкциях блочного программирования, допускающую более раннее появление выходного сигнала без изменения параметра точности выхода в позицию.

6. PREFETCH.SIGINS

Устанавливает синхронизацию для выходных сигналов в AS программировании. Действие аналогично OX.PREOUT.

7. QTOOL

Разрешает или запрещает в режиме обучения изменять коррекцию инструмента в соответствии с номером коррекции, заданным в инструкциях блочного программирования.

8. REP_ONCE (Repeat Once)

Когда переключатель в состоянии ON, программа выполняется один раз, когда в состоянии OFF, программа выполняется непрерывно.

9. RPS (Random Program Selection)

Разрешает или запрещает выбор программ, задающийся двоичным состоянием внешних сигналов.

10. SCREEN

Разрешает или запрещает просмотр страниц, когда информация не помещается на одном листе (строке).

11. STP_ONCE

Устанавливает пошаговый или непрерывный режим выполнения программы.

2.3 AS СИСТЕМНЫЕ УСТАНОВКИ

Следующие системные параметры настройки могут быть изменены в зависимости от потребности, используя мониторные команды.

1. Обнуление (команда ZZERO)

ZZERO команда используется, чтобы установить значение кодера в соответствии с известной механической позицией робота. При замене двигателя серводвигателя или выполнения обслуживания на кодере, значение кодера будет нуждаться в настройке, для этого используют эту команду. (Эта команда – только для целей обслуживания.)

2. Установка фиксатора (команда HSETCLAMP)

Эта установка сделана по отгрузке с фабрики. Параметры настройки, единичные/двойные и спецификации выхода (ON когда открыто / OFF когда закрыто), могут быть изменены, используя HSETCLAMP команду. Однако, изменение при этом затронет программное обеспечение, так что убедитесь, что проверили, является ли оно совместимым с аппаратными средствами.

3. Максимальное количество сигналов входа и выхода (команда ZSIGSPEC)

ZSIGSPEC команда устанавливает максимальное количество сигналов входа и выхода, которые могут использоваться.

Они устанавливаются при фабричной отгрузке. (Это является настройкой по умолчанию, которые функционируют как проверка ошибки программного обеспечения, таким образом убедитесь, что они совместимы с аппаратными средствами.)

4. Программные приоритетные сигналы (команда DEFSIG)

В дополнение к аппаратным приоритетным сигналам, есть сигналы входа - выхода в программном обеспечении, которые могут использоваться, как приоритетные сигналы (Программно созданные приоритетные сигналы). Сигналы в таблице ниже могут использоваться, как программно созданные сигналы. Обратите внимание, что количество I/O сигналов в программном обеспечении есть сумма программно созданных приоритетных сигналов и сигналов общего назначения, количество сигналов общего назначения уменьшается на количество используемых программно созданных приоритетных сигналов.

Входы	Выходы
EXT. MOTOR ON	MOTOR_ON
EXT. ERROR RESET	ERROR
EXT. CYCLE START	AUTOMATIC
EXT. PROGRAM RESET	CYCLE START
Ext. prog. select (JUMP_ON, JUMP_OFF RPS_ON, RPSxx)	TEACH MODE
EXT_IT	HOME1, HOME2
EXT. SLOW REPEAT MODE	POWER ON
	RGSO
	Ext. prog. select (JMP_ST, RPS_ST)

2.4 ОПЕРАЦИИ ВВОДА И ВЫВОДА

2.4.1 УПРАВЛЕНИЕ ТЕРМИНАЛОМ

Данные и команды, вводимые в терминал, поступают через системный буфер. Затем они читаются монитором или программой, повторяются и отображаются на экране. Максимальное количество символов, которые могут быть введены в терминал 128, остальные символы игнорируются.

Вывод данных на терминал можно осуществить, используя инструкции PRINT или TYPE. 8 битов отображается на терминальном экране. Если формат не определен, использование спецификации сода “/S” с инструкциями PRINT или TYPE позволит отображать данные с новой строки после каждой команды.

Терминальный ввод и вывод может быть управляем, используя следующие команды. Эти команды называются команды управления терминалом. Клавиша **Ctrl** нажимается с каким-либо алфавитным символом. В отличие от других AS команд, данные команды выполняются без нажатия клавиши **Enter**, как это происходит в других случаях.

Команды	Назначение
Ctrl + S	Останавливает прокрутку информации на экране
Ctrl + Q	Продолжает вывод данных, остановленных предыдущей командой
Ctrl + C	Отменяет последнюю линию ввода
Ctrl + H	Удаляет последний введенный символ
Ctrl + M	Заканчивает ввод текущей строки
Ctrl + L	Отображает содержимое строки, введенной ранее текущей. Возможно отобразить до семи строк
Ctrl + N	Отображает содержимое строки, которое было до вывода строки при помощи Ctrl + L
Backspace	Удаляет последний введенный символ

Ctrl + **I** или **TAB** воспринимается как пробел.

2.4.2 ВНЕШНИЕ УСТРОЙСТВА ПАМЯТИ

Команды, показанные ниже, используются для сохранения программ, переменных, позиционной информации в памяти контроллера, PC карте, дискете или жестком диске компьютера.

1. Инициализация памяти (CARD_FORMAT, FD_FORMAT)
2. Отображение содержимого (CARD_FDIR, FD_FDIR)
3. Сохранение информации во внешней памяти (SAVE, CARD_SAVE, FD_SAVE)

4. Загрузка информации из внешней памяти (LOAD, CARD_LOAD, FD_LOAD)
5. Удаление файлов (DELETE, CARD_FDEL, FD_FDEL)

Команды с CARD_ работают с PC картой, FD_ с дискетой.

ПРИМЕЧАНИЕ* SAVE, используется при подключенном к системе персональном компьютере.

2.5 УСТАНОВКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ТЕРМИНАЛА

Робот может управляться от персонального компьютера, используя AS язык. Чтобы сделать так, загрузите KCwin32 или KRterm программное обеспечение терминала на PC и подключите PC с контроллером D серии.

KCwin32 и KRterm может быть установлен на компьютерах, работающих в Windows 95/98/Me/2000/XP.

При подсоединении компьютера и контроллера, используют кабель RS-232C, который позволяет отдельному компьютеру управлять отдельным роботом. Подключение Ethernet дает возможность множественным компьютерам управлять множественными роботами. Программное обеспечение терминала KCwin и KCwin32 может также использоваться путем подключения в порт RS-232C.

Следуйте ниже указанным процедурам, чтобы установить программное обеспечение терминала на PC.

1. Установите программное обеспечение на ваш жесткий диск

Копируйте следующие файлы с KRterm дискеты в любой каталог жесткого диска вашего компьютера.

KRTERM.EXE

KRTERMJ.HLP (файл справки японской версии)

KRTERME.HLP (файл справки английской версии)

KRTERM.INI

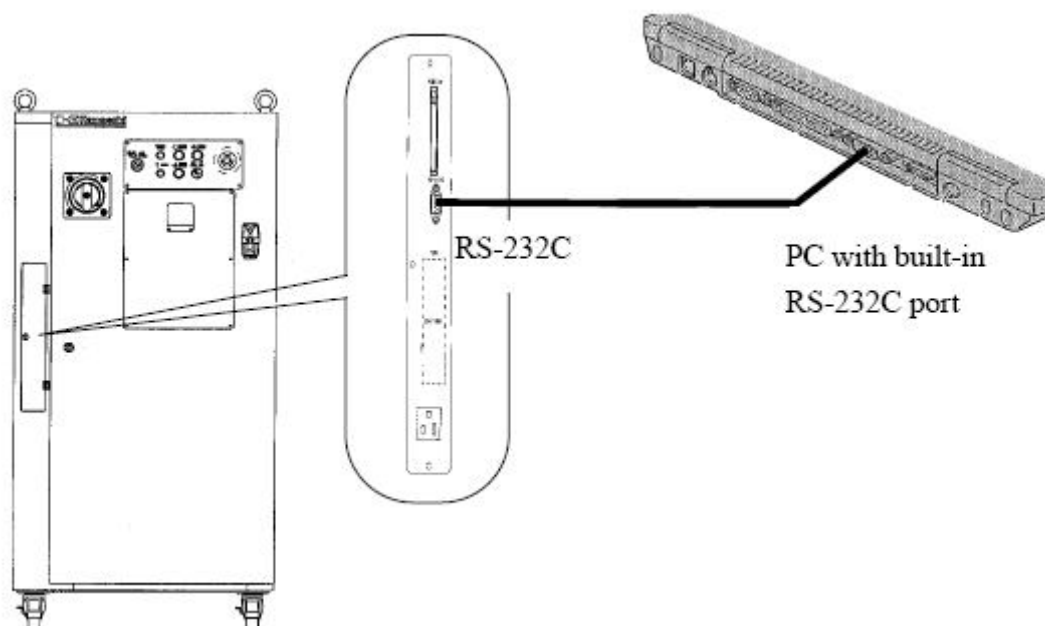
2. Рекомендуются, чтобы Вы сделали ярлык на рабочем столе или в стартовом меню для более простого запуска KRterm программного обеспечения.

2.6 РАБОТА С ПЕРСОНАЛЬНЫМ КОМПЬЮТЕРОМ

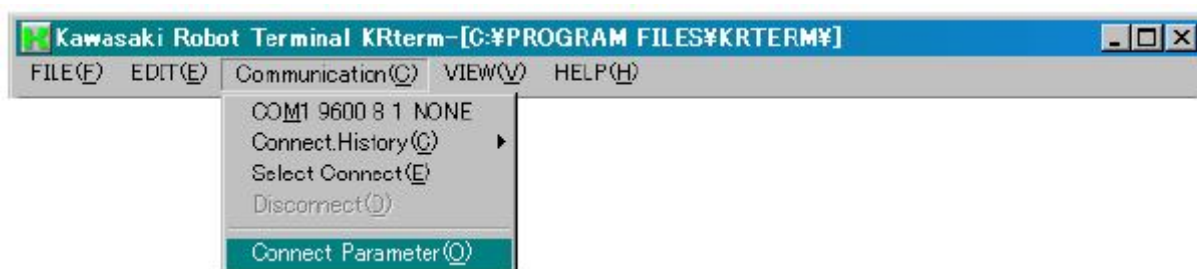
2.6.1 ЗАПУСК СИСТЕМЫ

2.6.1.1 ПОДКЛЮЧЕНИЕ К ПОРТУ RS -232C

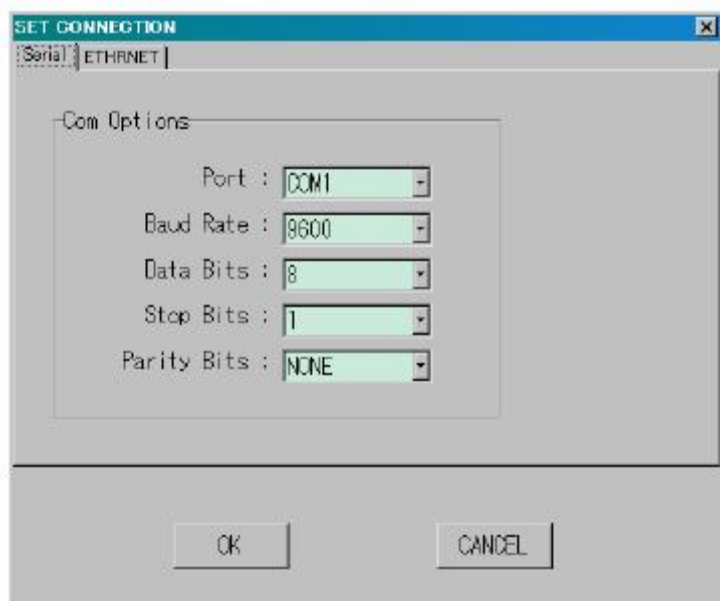
1. Подключите персональный компьютер к контроллеру, используя кабель RS-232C. Удостоверьтесь, что силовое питание контроллера и компьютера выключено.



2. Включите компьютер, и запустите программное обеспечение терминала (KRterm или KCwin32, диаграммы ниже касаются KRterm).
3. Когда программное обеспечение открывается, выберите тип подключения для использования. Выберите из строки меню, [Communication (C)] → [Connect Parameter (O)].



4. Щелкните “Serial” строку, проверьте содержимое, и если все нормально, щелкните <OK>.



5. Включите силовое питание контроллера.

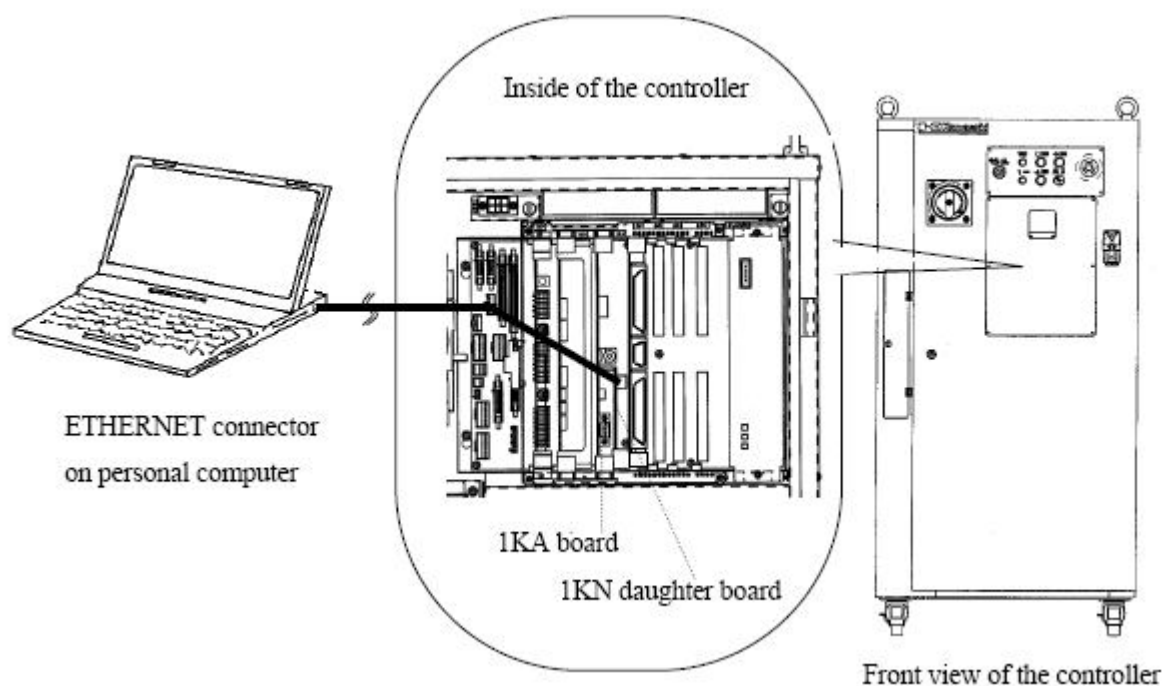
(См. “Руководство оператора” 3.1 Процедура включения силового питания.

6. Первоначальный экран KRterm следует за подсказкой “>”, которая появится на экране. Когда силовое питание включено до подключения РС и контроллера, появится только подсказка “>” без первоначального экрана. Тем не менее, KRterm работает одинаково.

2.6.1.2 ПОДКЛЮЧЕНИЕ РОБОТОВ, ИСПОЛЬЗУЯ СЕТЬ ETHERNET

1. Подключение кабелей.

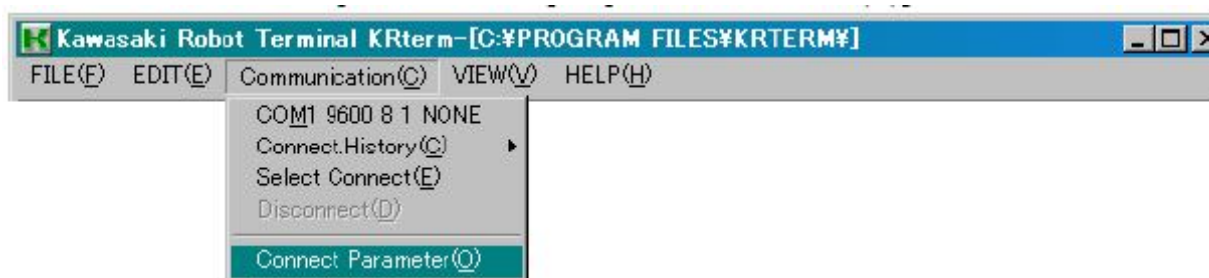
Подключите ETHERNET разъем на вашем персональном компьютере и разъем на дополнительной 1 KN дочерней плате на 1 KA плате в контроллере, используя кабель локальной сети.



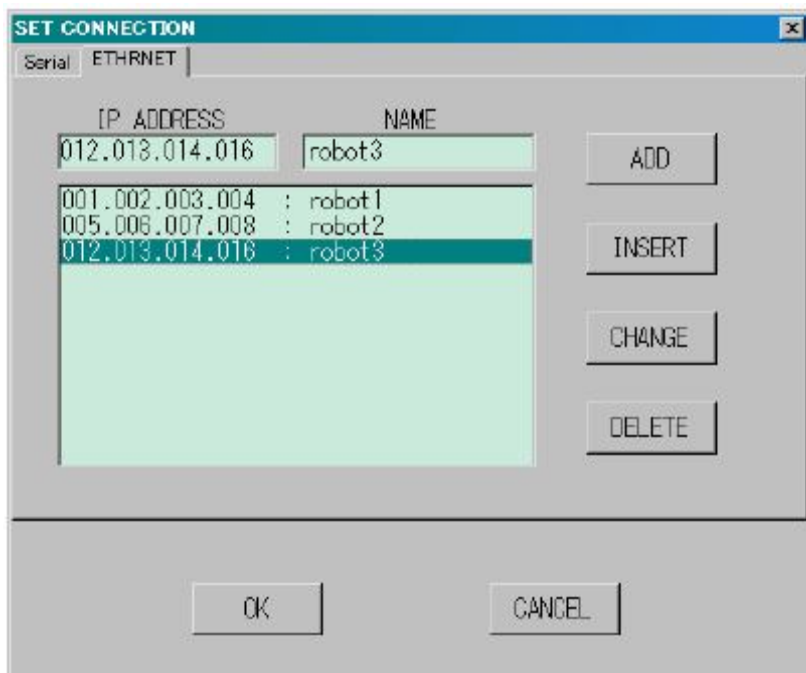
2. Следуя шагам 2 и 3, описанных выше, запустите KRterm и робот.

3. Установите IP адрес робота.

(1) Перейдите из строки меню [Communication] → [Connect Parameter (O)].



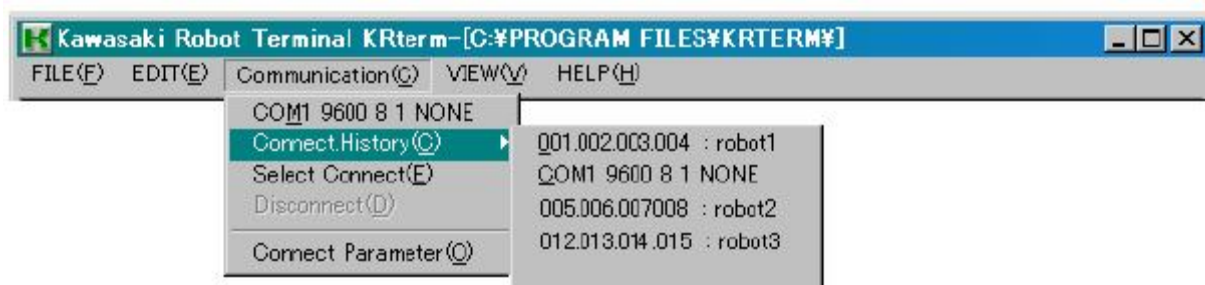
(2) Введите IP адрес и имя робота, который вы хотите подключить к сети и нажмите <ADD>.



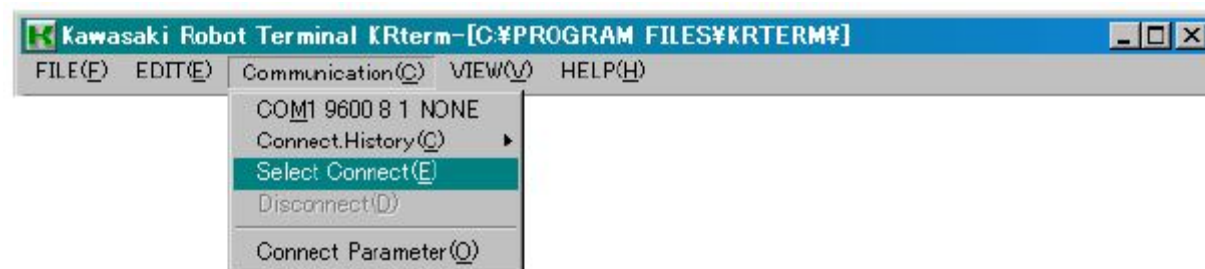
4. Для того чтобы подключиться с роботом, следуйте за процедурами, описанными ниже.

(1) Робот, подключенный последним, отображен наверху спускающегося меню, когда [Communication (C)] выбрана от строки меню. Нажмите имя робота, для того чтобы выбрать робот.

(2) Выберите из строки меню [Communication(C)] → [Connect.History(C)], чтобы отобразить список используемых роботов. Выберите робот для подключения из списка.



3) Выберите из строки меню [Communication(C)] → [Select Connect(E)], если желательный робот не появляется при пользовании вышеупомянутых способов.



Выберите желательный робот, и нажмите <CONNECT>.



Если подключение установлено, информация робота типа его имени, сопровождаемого подсказкой “>”, появляется на KRterm экране. AS команды могут быть введены, как только подсказка появляется.

2.6.2 ЗАГРУЗКА И ВЫГРУЗКА ДАННЫХ

(1) SAVE команда

Чтобы сохранять данные относительно компьютера, используйте команду SAVE (См. 5.3 команда SAVE).

Пример> SAVE test.pg, сохраняет данные в том же самом каталог, в котором находится KRterm на компьютерном жестком диске.

> SAVE test.pg ¥ My Documents сохраняет данные в указанном файле.

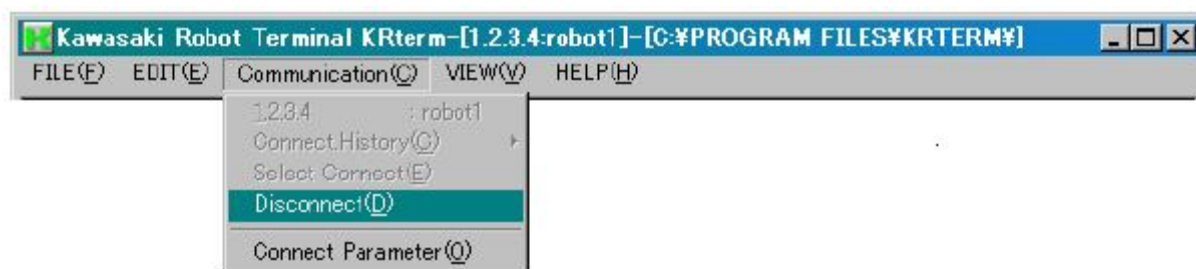
(2) LOAD команда

Чтобы загружать данные из компьютера в память робота, используйте команду LOAD.

Пример> LOAD data01.as

2.6.3 ЗАВЕРШЕНИЕ РАБОТЫ СИСТЕМЫ

1. Когда робот подключен, выберите из строки меню [Communication(C)] → [Disconnect(D)], чтобы отключить робот.

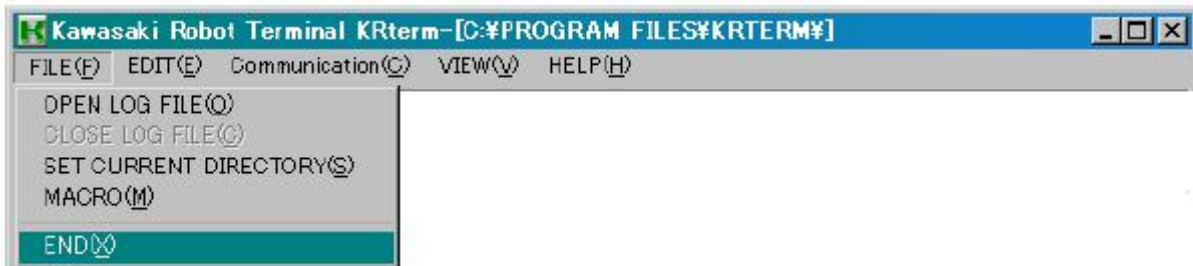


2. Выключите контроллер робота.

- (1) Поверните переключатель **HOLD/RUN** в состояние HOLD
- (2) Выключите силовое питание привода, нажимая кнопку **EMERGENCY STOP**.
- (3) Выключите силовое питание.

3. Закройте KRterm.

- (1) Выберите из строки меню **[FILE(F)] → [END(X)]**.



- (2) нажмите <Yes>

4. Выключите компьютер.

5. Если нет никакой необходимости сохранять подключение компьютера с контроллером, отсоедините кабель.

Удостоверьтесь, что контроллер и компьютер выключены перед разъединением.

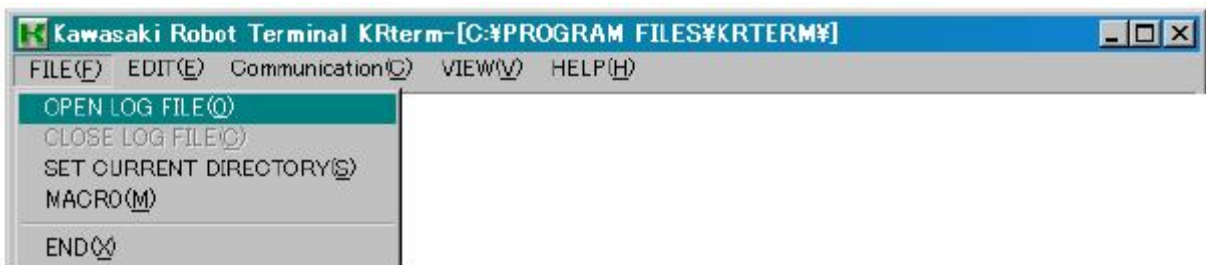
2.6.4 ПОЛЕЗНЫЕ ФУНКЦИИ KRTERM

2.6.4.1 СОЗДАНИЕ ЖУРНАЛА РЕГИСТРАЦИИ

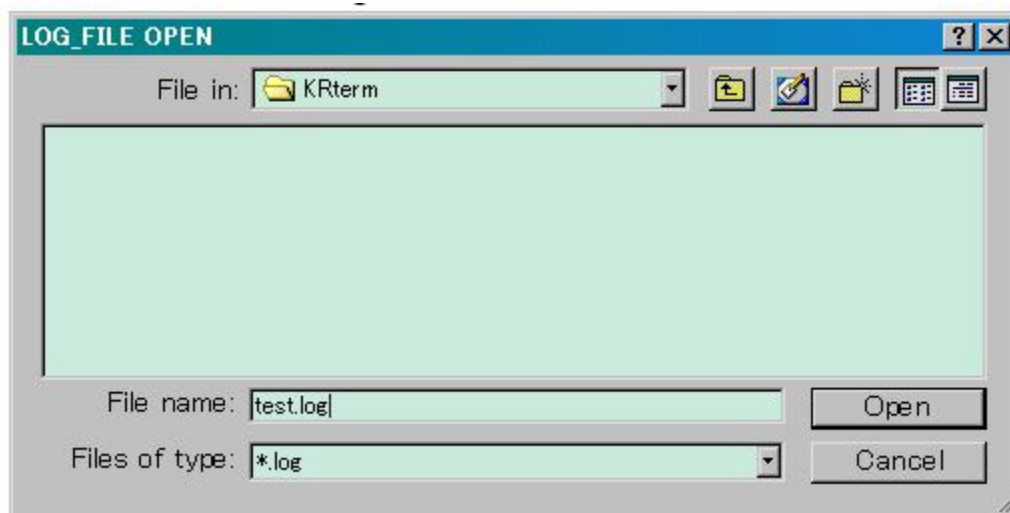
Содержание, отображенное на KRterm экране, может быть сохранено как журнал. Это полезно для создания распечатки рабочих процедур робота.

1. Начало регистрации.

- (1) Выберите из строки меню **[FILE(F)] → [OPEN LOG FILE(O)]**.



- (2) Выберите папку, чтобы сохранить журнал, и назовите файл.



(3) Сообщение [Logging Now] появится в области заголовка. Содержание дисплея регистрируется, пока журнал не закрыт.

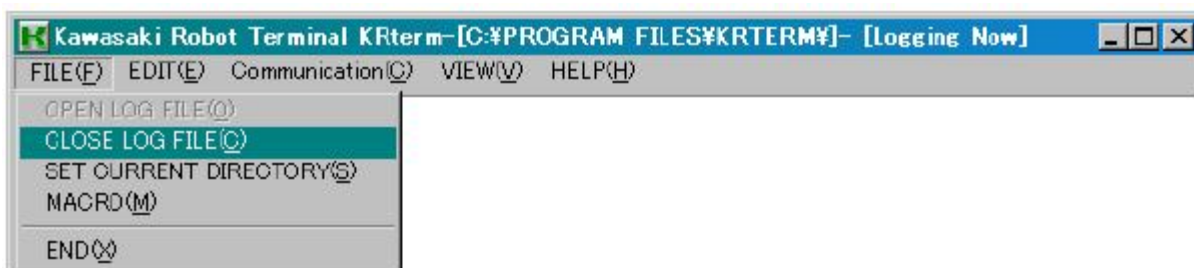


Содержание дисплея регистрируется, пока есть это сообщение.

2. Завершение регистрации

Т.к. регистрация начата, все содержание дисплея KRterm будет зарегистрировано, пока журнал не закрыт.

Чтобы закрыть журнал и завершить регистрацию, выберите из строки меню [FILE(F)] → [CLOSE LOG FILE(C)].



2.6.4.2 МАКРО ФУНКЦИИ

Макро функции обеспечены для KRterm и KCwin32 системы. Если задача должна быть выполняема неоднократно, запись серии команд/инструкций для той задачи в макрокомандах может быть очень полезна и увеличит эффективность.

Чтобы сделать запись макрокоманды, выберите из строки меню [FILE (F)] → [MACRO (M)], и введите имя файла, чтобы сохранить эту макрокоманду. Чтобы выполнить макрокоманду, используйте команду SEND на KRterm экране.

См. Справку в KRterm или KCwin32 для большего количества подробностей.

3.0 ИНФОРМАЦИОННЫЕ ВЫРАЖЕНИЯ В AS ЯЗЫКЕ

Эта глава описывает виды информации и переменных, используемых в AS языке программирования

3.0 Информационные выражения в AS языке

3.1 Система обозначений и соглашений

3.2 Информация о позиции, числовая информация, символьная информация

3.3 Переменные

3.4 Имена переменных

3.5 Задание переменных позиции

3.6 Задание реальных переменных

3.7 Задание строковых переменных

3.8 Числовые выражения

3.9 Строковые выражения

3.1 СИСТЕМА ОБОЗНАЧЕНИЙ И СОГЛАШЕНИЙ

1. Заглавные и строчные буквы.

Для дальнейшего понимания применяются следующие правила использования заглавных и строчных букв в данном руководстве. Все ключевые слова (команды, инструкции и т.д.) показаны заглавными буквами. Переменные, которые могут быть заданы, строчными буквами. Ввод с клавиатуры может быть осуществлен в обоих вариантах.

2. Кнопки и переключатели.

Кнопки на пульте ручного управления и переключатели на операционной панели контроллера, упоминаемые в данном руководстве, заключены в прямоугольный контур

Например: RUN/HOLD

3. Сокращения.

Некоторые ключевые слова могут быть сокращены.

4. Пробел.


Пробел необходимо вводить между командным словом и параметром для их разделения. Пробел необходимо вводить, если параметры не разделены запятыми. Лишние пробелы или табуляции игнорируются системой.

ПРИМЕЧАНИЕ* параметр - необходимые данные, чтобы завершить команды или другие функции. Для примера, в команде SPEED, данные параметра необходимы для того, чтобы определить скорость робота. Когда команда или функция используют несколько параметров, запятая или пробел отделяет каждый параметр.

Пример SPEED 50

5. Кнопка ENTER

Мониторные команды и программные инструкции вводятся нажатием клавиши Enter.

В данном руководстве ENTER кнопка может быть в виде .

6. Пропускаемые параметры

Многие мониторные команды и программные инструкции имеют параметры, которые можно пропустить. Но пропущенный параметр должен отделяться запятой, если дальше следуют еще параметры.

7. Числовые значения.

Значения выражаются в десятичном исчислении, если не задано иначе. Математические выражения могут использовать указанные величины как аргументы. Однако, обратите внимание, что приемлемые значения ограничены. Следующие правила показывают, как значения интерпретируются в различных случаях.

(1) расстояние

Используется, чтобы определить длину перемещения робота между двумя точками.

Единица измерения для расстояния миллиметр (мм); единица измерения опускается при вводе. Входные значения могут быть любыми отрицательными и положительными

(2) углы

Определяет и изменяет положение робота в указанной позиции, и описывает количество вращений сустава робота. Значения могут быть отрицательными и положительными, с максимальными углами, ограниченными 180 градусами или 360 градусами, в зависимости от используемых команд.

(3) скалярные переменные

Если не отмечено иначе, эти переменные определяют реальные значения. Значения для переменных могут быть в диапазоне от $-3.4E+38$ до $3.4E+38$ ($-3.4 \times 10^{38} \sim 3.4 \times 10^{38}$). Когда значение превышает ± 999999 , оно выражается как $xE+y$ (x мантисса, y - экспонента).

(4) номер сустава

Выражает сустав робота в целом числе от 1 до номера доступного сустава (стандартный тип имеет 6 суставов). Суставы пронумерованы, начиная с основного сустава. (Обычно выражаются JT1, JT2).

(5) номер сигнала

Идентифицирует бинарные сигналы (ON/OFF). Значения даются как целые числа и находятся в следующих диапазонах.

	Standard range	Maximum range
Внешний выходной сигнал	1 – 32	1 – 96
Внешний входной сигнал	1001 – 1032	1001 – 1096
Внутренний сигнал	2001 – 2256	2001 – 2256

Отрицательные числа указывают состояние OFF

8. Ключевые слова.

В общем, имена переменных могут свободно присваиваться внутри AS системы. Однако, ключевые слова, определяющие команды, инструкции и т.д. не могут присваиваться именам позиций, переменным и т.д.

3.2. ИНФОРМАЦИЯ О ПОЗИЦИИ, ЧИСЛОВАЯ ИНФОРМАЦИЯ, СИМВОЛЬНАЯ ИНФОРМАЦИЯ

Есть три типа информации в AS системе: позиционная* информация, числовая информация, и символьная информация.

ПРИМЕЧАНИЕ* "Позиция" прежде назывался "местоположением", но в соответствии с международным стандартом (Международная организация по стандартизации), в этом руководстве, упоминается как позиция, чтобы выразить и координату позиции и ориентацию робота одним словом.

3.2.1 ПОЗИЦИОННАЯ ИНФОРМАЦИЯ

Позиционная информация используется для точного определения положения и ориентации робота в заданной рабочей области, т.е. эту информацию можно понимать как координатную информацию. Координата точки, определяющей положение и ориентацию

робота в пространстве, находится в центре фланца робота (TCP – tool center point). В AS системе каждой запоминаемой координате присваивается имя. Имя начинается обязательно с буквы или знака #. Именам нельзя присваивать наименования команд, инструкций, функций.

Позиция определяется тем, где робот находится и каким образом он поворачивается, поэтому, когда робот проинструктирован, чтобы двигаться, две вещи делаются в одно и то же время:

1. TCP робота перемещается в указанную позицию.
2. Инструментальная система координат робота вращается к указанному положению.

В AS системе возможны два типа задания координатной точки в трехмерном пространстве: последовательностью угловых смещений каждой из осей суставов, относительно какого-то начального положения (в дальнейшем будем называть угловая координата), последовательностью линейных и угловых смещений начала базовой системы координат (в дальнейшем будем называть декартовая координата)

1. Угловая координата

Используя значения кодера (кодирующего устройства), по количеству вращений вала каждого из суставов рассчитываются угловые смещения каждой из осей суставов относительно какого-то начального положения. Последовательность величин смещений каждой оси задается в градусах. Данная последовательность при запоминании задаст положение и ориентацию центра фланца робота в пространстве. При запоминании точки в угловых координатах необходимо перед буквой поставить знак #.

Пример

```
JT1   JT2   JT3   JT4   JT5   JT6
#pose = 0.00, 33.00, -15.00, 0, -40, 30
```

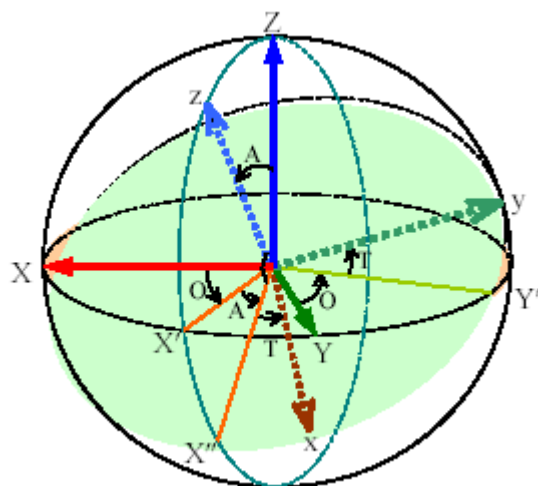
2. Декартовая координата

Описывает положение координат по отношению с рекомендуемыми координатами. Если не определено иначе, это относится к значениям преобразования координат инструмента относительно основных координат робота. Позиция дается значениями XYZ для основных координат, и положением O, A, T углов Эйлера*.

Некоторые из обычно используемых значений преобразования:

значение преобразования инструмента, описывает положение инструментальной системы координат относительно нулевой инструментальной системы координат, и значения преобразования системы координат детали, описывающие положение инструментальной системы координат относительно координат детали.

Положение и ориентация центра фланца в пространстве задается последовательностью смещений начала инструментальной системы координат относительно начала базовой системы координат координатами X, Y, Z и углами Эйлера O, A, T .



Углы Эйлера

O – угол между плоскостью Zz и плоскостью XZ

A – угол между осью z и осью Z

T – угол между осью x и осью X''

Ось X'' лежит в плоскости Zz и образует с осью z угол 90 градусов.

Имя декартовой координаты начинается с буквы.

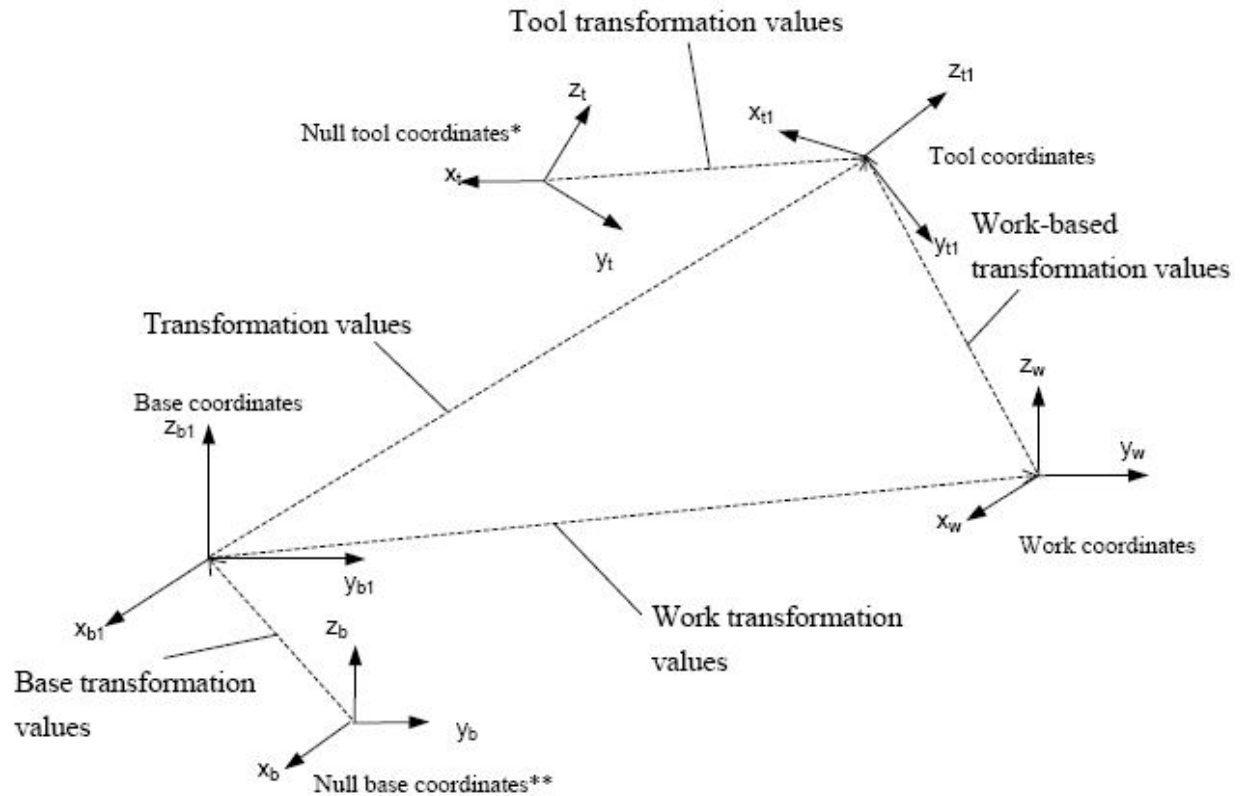
Пример

	X	Y	Z	O	A	T
pose =	0,	1434,	300,	0,	0,	0

Если робот имеет больше чем шесть осей, значение дополнительной оси показывают с декартовым значением

Example

	X	Y	Z	O	A	T	JT7
pose =	0,	1434,	300,	0,	0,	0	1000



ОБРАТИТЕ ВНИМАНИЕ *, нулевые координаты инструмента имеют свое начало координат в центре монтажной поверхности фланца, и они описаны значениями преобразования инструмента (0,0,0,0,0,0).

ОБРАТИТЕ ВНИМАНИЕ, ** Нулевые основные координаты установлены как значение по умолчанию для робота, и описаны основными значениями преобразования (0,0,0,0,0,0).

Угловые координаты и декартовы координаты имеют преимущества и недостатки. Используйте их так, чтобы удовлетворить вашу потребность.

	Значение угловых координат	Значение декартовых координат
Преимущества	Достигается точность воспроизведения и нет никакой двусмысленности о конфигурации робота в позиции	<ul style="list-style-type: none"> · начало инструментальной системы координат, используемое в автоматическом режиме не изменяется даже если инструмент заменен. (Нулевой сдвиг координат инструмента) · Может использовать относительные координаты (например, координаты детали) · Удобны для того, чтобы обрабатывать как данные, показанные в значениях XYZOAT

Недостатки	<p>TCP изменяется, когда меняется инструмент (нулевые координаты инструмента остаются теми же самими)</p> <ul style="list-style-type: none"> · Не может использовать относительные координаты (например, координаты детали, и т.д.) 	<ul style="list-style-type: none"> · Координаты изменятся согласно базовым или инструментальным значениям преобразования, поэтому необходимо полное понимание эффекта любого изменения для безопасного использования · конфигурация Робота может измениться, если она не установлена перед повторяющимися движениями
Советы по использованию	<p>Установка стартовой позиции в программе</p> <ul style="list-style-type: none"> · Установка конфигурации робота в или как раз перед позицией, заданной в декартовых координатах · Использование для других общих положений 	<ul style="list-style-type: none"> · Описание относительных координат типа координат детали · Описание позиции, которая должна быть изменена, используя числовые значения с функциями типа SHIFT · Описание позиции, которая должна быть изменена по информации датчика

(ПРИМЕЧАНИЕ)

1. В отличие от позиции, определенной угловыми координатами, где конфигурация робота устанавливается однозначно, когда позиция задана декартовыми координатами, робот может принять различную конфигурацию относительно этой позиции. Это происходит, потому что значения декартовых координат только устанавливают XYZOAT значения координат инструмента робота и не определяют значение оси каждого сустава. Поэтому, перед стартом робота в повторном режиме, убедитесь, что установили конфигурацию робота, используя команды конфигурации (LEFTY) или делая запись значений угловых координат.

2. Так как декартовые значения описаны основными координатами, если центр координат сдвинут, используя команду/инструкцию BASE, TCP робота будет также сдвинута на ту же самую величину. Это - одно из преимуществ использования декартовых значений, но обратите внимание, что изменение основных координат будет влиять на все декартовые точки.

Неудачное применение может вызвать несчастные случаи типа столкновения с периферийными устройствами.

Учитывайте то же самое предостережение при использовании команды/инструкции TOOL.

3.2.2 ЧИСЛОВАЯ ИНФОРМАЦИЯ

В AS системе числовые величины и выражения могут быть использованы как числовая информация. Числовое выражение есть величина, выраженная числами или переменными в комбинации с операторами и функциями. Числовые выражения могут использоваться не только как математические вычисления, но и как параметры в командах, инструкциях, функциях.

Для примера, в DRIVE команде, три параметра: номер сустава, градус, и скорость задаются. Параметры могут быть выражены либо в числовых значениях, либо в выражениях, как показано в примере:

Пример

```
DRIVE 3,45,75
```

```
DRIVE joint, (start+30)/2, 75
```

Первая инструкция задает движение третьего сустава на 45 градусов со скоростью 75%. Вторая инструкция задает движение второго сустава на +30 градусов со скоростью 75%, если joint=2, start=30

Числовые величины в AS системе подразделяются на три типа:

1. Действительные числа

Действительные числа могут быть целыми и дробными, положительными и отрицательными или нулем в пределах от $-3.4E+38$ до $3.4E+38$. Действительные числа могут быть выражены в экспоненциальном представлении. Символ E разделяет мантиссу и экспоненту.

Пример

8.5E3	$8.5 \cdot 10^3$
6.64	$6.64 \cdot 10^0$
-9E-5	$-9.0 \cdot 10^{-5}$
-377	$-377 \cdot 10^0$

Обратите внимание, что первые семь цифр допустимы, но количество допустимых цифр может быть уменьшено через процедуры вычисления.

Реальные значения без дробных частей (целые числа) называют целыми числами.

Диапазон от $-16,777,216$ до $+16,777,215$ и для тех, чьи значения превышают этот предел, первые семь цифр допустимы. Целочисленные значения обычно вводятся в десятичные числа, хотя временами является удобным выражение в двоичном или шестнадцатеричном виде. ^В впереди числа означает число в двоичном виде, ^H впереди числа означает число в 16-ом виде.

Пример

<code>^B101</code>	(5 in decimal)
<code>^HC1</code>	(193 in decimal)
<code>−^B1000</code>	(−8 in decimal)
<code>−^H1000</code>	(−4096 in decimal)

2. Логические величины

Логические величины имеют только два состояния ON/OFF или TRUE/FALSE (ВЕРНО/ЛОЖНО). Величина −1.0 присваивается для состояния TRUE или ON, величина 0.0 присваивается для состояния FALSE или OFF. ON, OFF, TRUE, FALSE используются, как ключевые слова в AS программировании.

3. ASCII величины

Показывают числовые значения каждого символа ASCII кода. Символ, имеет впереди апостроф (') чем отличается от других величин

Пример

`'A '1 'v '%`

3.2.3 СИМВОЛЬНАЯ ИНФОРМАЦИЯ

Символьная информация передается в AS систему, как строка ASCII символов, заключенных двойные кавычки (" "). Т.к. двойные кавычки означают начало и конец символьной цепочки, они не могут быть использованы как составная часть в строке символов. Также управляющие символы ASCII (CTRL, CR, LF и т.д.) не могут быть включены в символьную цепочку.

Пример

`>PRINT "KAWASAKI"`

```
graph TD
    command[command] --> G1[>PRINT]
    string[character string] --> G2["KAWASAKI"]
```

3.3 ПЕРЕМЕННЫЕ

В AS системе, названия могут быть присвоены информации о позиции, числовой информации, и символьной информации. Эти названия называются переменными, и переменные могут быть разделены на два типа:

глобальные переменные и локальные переменные. Если иначе не отмечено, глобальные переменные упоминаются как переменные.

3.3.1 ПЕРЕМЕННЫЕ (ГЛОБАЛЬНЫЕ ПЕРЕМЕННЫЕ)

Переменные для позиционной информации, числовой информации, и символьной информации называют позиционной переменной, реальной переменной *, строковой переменной, соответственно. Несколько значений могут быть сгруппированы и быть определены под одним названием {именем}, используя переменную типа массив.

ПРИМЕЧАНИЕ*, так как большинство числовых значений, используемых в AS - действительные числа, числовые переменные названы переменные действительного числа или реальные переменные. Однако, обратите внимание на то, что целые числа, логические значения и значения ASCII все выражаются, используя значения действительного числа. Поэтому, реальная переменная может выражаться в любых из этих значений.

Как только переменная определена, она сохраняется с этим значением в памяти. Поэтому, может использоваться в любой программе.

3.3.2 ЛОКАЛЬНЫЕ ПЕРЕМЕННЫЕ

В отличие от глобальных переменных, описанных выше, локальные переменные переопределяются каждый раз, когда программа выполняется, и не сохраняются в памяти. Переменная с "." (период) в начале имени считается локальной переменной. Локальные переменные полезны в случаях, когда несколько программ используют то же самое имя переменной, причем значение переменной изменяется каждый раз, когда программа выполняется. Локальные переменные могут также быть используемы как параметр подпрограммы. (См. также 4.4.2 Подпрограммы с Параметрами.)

(ПРИМЕЧАНИЕ)

1. Локальные переменные не могут быть определены, используя мониторные команды.
2. Так как локальные переменные не сохраняются в памяти, значение локальной переменной .pose не может быть отображено, используя команду.

> POINT. pose

Чтобы видеть текущее значение локальной переменной, установите ее значение в глобальную переменную в программе, где локальная переменная определена, и затем используйте команду POINT.

POINT a=.pose Выполняет программу, которая определяет локальную переменную до использования команды POINT.

```
>POINT a
X[mm] Y[mm] Z[mm] O[deg] A[deg] T[deg]
xxxxxxx xxxxxxx xxxxxxx xxxxxxx xxxxxxx xxxxxxx
Change? ( If not hit RETURN key ) ☐
```


3.4 ИМЕНА ПЕРЕМЕННОЙ

Имена переменной должны начинаться с алфавитного символа и могут содержать только символы, числа, периоды, и символы подчеркивания. Символы могут быть введены или в верхнем регистре, или в нижнем регистре (на экране будут появляться в нижнем регистре). Длина переменной ограничена пятнадцатью символами. Только первые пятнадцать символов будут допустимы с более длинными названиями {именами}. Следующие некоторые примеры имен, которые не могут использоваться:

3p	- первый символ - не алфавитная буква
part#2	- “#” префикс для имени координаты и не может быть использована в середине имени переменной
random	- ключевое слово

(ПРИМЕЧАНИЕ)

1. Переменным, описывающим угловые координаты, предшествует символ “#”, чтобы дифференцировать их от значений декартовых координат. Символьным строковым переменным предшествует “\$” для дифференцирования их от реальных переменных.

pick (декартова координата)	#pick (угловая координата)
count (реальная переменная)	\$count (строковая переменная)

2. Все переменные могут использоваться как переменные типа массив. Массивы состоят из нескольких значений под тем же самым именем, и эти значения отличаются друг от друга их индексным значением. Каждое значение в массиве называют элементом массива. Чтобы определить элемент массива, присвойте значению элемента индекс, заключенный в скобки. Например, “part [7]” указывает седьмой элемент массива “part”. Для индексов, используйте целые числа в пределах диапазона от 0 до 9999. Для трехмерных массивов используют синтаксис, подобный этому: part [7, 1, 1] = 1.

3. Когда переменная определена, эта переменная может использоваться в различных программах. Поэтому, будьте внимательны, чтобы не сделать ненужных изменений переменных, используемых в различных программах.

3.5 ЗАДАНИЕ ПЕРЕМЕННЫХ ПОЗИЦИИ

Переменные, которые описывают информацию о позиции, называют переменными позиции. Переменная позиции определена только, когда значение присвоено для нее. Остается неопределенной, пока значение не присвоено, и если программа использует неопределенную переменную при выполнении, происходит ошибка.

Переменные позиции удобны для использования в следующих случаях:

1. Данные одной и той же позиции могут использоваться неоднократно, без необходимости обучать позицию каждый раз.
2. Определенная переменная позиции может использоваться в различных программах.
3. Определенная переменная позиции может использоваться или изменяться, чтобы определить различные положения.
4. Расчетные значения могут использоваться как информация позиции вместо трудоёмкого процесса обучения положений робота, используя пульт ручного управления.

5. Переменные позиции можно называть по-разному, так что программы могут быть сделаны более понятными.

Переменные позы определяются следующим образом.

3.5.1 ЗАДАНИЕ МОНИТОРНЫМИ КОМАНДАМИ

1. Команда HERE записывает текущие данные позиции робота под указанным именем

Пример 1 Использование угловых координат

Начните имя переменной с #, чтобы дифференцировать его от декартовых координат.

После выполнения команды, появятся угловые координаты текущей позиции:

```
> HERE #pose 
JT1      JT2      JT3      JT4      JT5      JT6
xxxxxxx xxxxxxx xxxxxxx xxxxxxx xxxxxxx xxxxxxx
Change?
>
```

Пример 2 использование декартовых значений

После выполнения команды, появятся декартовые координаты текущей позиции :

```
> HERE pose 
X [мм]   Y [мм]   Z [мм] O [градус] [градус] T [градус]
xxxxxxx xxxxxxx xxxxxxx xxxxxxx xxxxxxx xxxxxxx
Change?
>
```

2. Команда POINT используется, чтобы определить позицию, используя другую определенную переменную позиции или, определить ее вводом данных с клавиатуры.

Пример 1 Использование текущего значения угловой координаты

(1) Определение новой, неопределенной переменной

```
>POINT #pose 
JT1      JT2      JT3      JT4      JT5      JT6
0.000    0.000    0.000    0.000    0.000    0.000
Change ? (if not, hit RETURN only) 
>
```

(2) изменение значения определенной переменной

```
>POINT #pose 
JT1      JT2      JT3      JT4      JT5      JT6
10.000   20.000   30.000   40.000   50.000   40.000
Change ? (if not, hit RETURN only) 
```

Введите значения, которые надо изменить

30, , , 20, ;изменяет значение JT1 и JT 5 в 30 и 20

(3) присвоение значения определенной переменной

```
>POINT  pose_1=pose_2    
      JT1    JT2    JT3    JT4    JT5    JT6  
      10.000 20.000 30.000 40.000 50.000 40.000  
Change ? (if not, hit RETURN only) 
```

Появится значение, которое будет определено как pose_1 (недавнее значение pose_2).

Нажмите для того чтобы установить значения как они есть, или измените их в том же самом порядке как в (2) выше.

Пример 2 Использования значений декартовых координат

Выполняйте те же самые процедуры как выше, только имя переменной не должно начинаться с #.

3.5.2 ЗАДАНИЕ ПРОГРАММНЫМИ ИНСТРУКЦИЯМИ

1. HERE инструкция сохраняет текущую позицию робота под указанным именем.

HERE pose

2. POINT инструкция присваивает переменной позиции значение предварительно определенной позиции.

POINT pose_1=pose_2

Значениям “pose_1” присваиваются значения определенной переменной “pose_2”.
Ошибка произойдет, если “pose_2” не определен.

(ПРИМЕЧАНИЕ)

Когда имя переменной позиции начинается с # , переменная описывает значения угловых координат (т.е. #pick, #start).

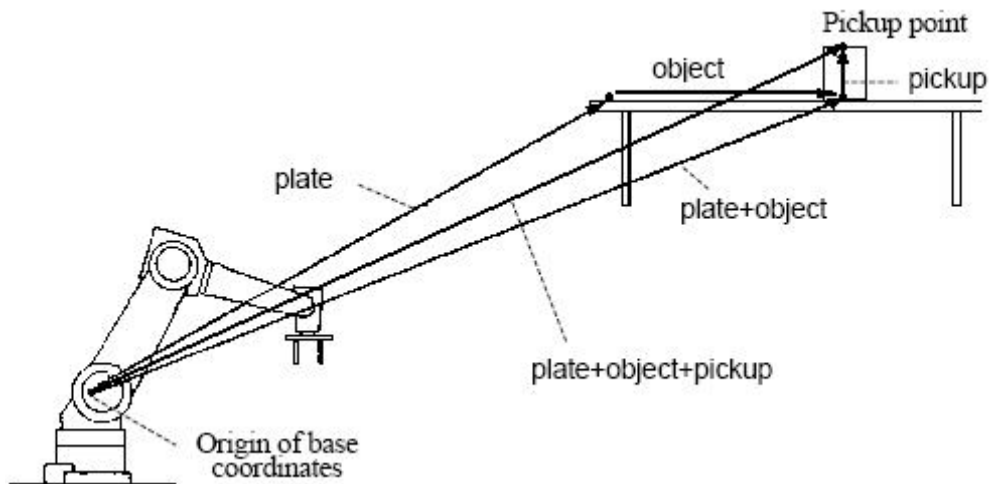
Когда переменная позиции не имеет префикса и начинается с алфавитной буквы, переменная описывает значения декартовых координат (pick, start).

3.5.3 ИСПОЛЬЗОВАНИЕ ЗНАЧЕНИЙ СОСТАВНЫХ (СЛОЖНЫХ) КООРДИНАТ

Значения преобразования между двумя координатами могут быть выражены как комбинация значений преобразования между двумя или больше переходными координатами. Это называют значением сложной координаты или значением относительной координаты.

Например, если "plate" – имя переменной для декартовых координат, описывающих координаты пластины относительно основных координат, а "object" – положение объекта относительно положения пластины, то значения составных координат объекта, относительно основных координат робота может быть описано как “plate+object”.

В примере ниже, даже если положение пластины изменится, только координаты пластины будут нуждаться в пересмотре, и остальные могут использоваться без переобучения.



Значения составных координат могут быть определены, используя любую команду или инструкцию, используемую для определения переменной позиции (Самое простое использовать команду HERE.)

Сначала, при помощи пульта ручного управления переместите инструмент робота в позицию, которую нужно назвать "plate". Затем, введите команду, как показано ниже, чтобы определить положение пластины.


➤ HERE plate

Затем, переместите инструмент робота в позицию, которая будет названа "object" и введите:

> HERE plate+object

Значение преобразования "object" теперь определяет текущую позицию относительно plate* (Если "plate" – не определена в этой точке, "object" не будет определен, и произойдет ошибка).

ОБРАТИТЕ ВНИМАНИЕ * То что появляется на экране после ввода команды HERE – значение декартовой координаты для крайне правой переменной (то есть "object" в этом случае). Это – не значения для "plate+object". Чтобы увидеть значения для "plate+object", используйте команду WHERE, когда робот находится в этом положении. Наконец, переместите робот в положение, где он поднимается и введите:

> HERE plate+object+pick up 

Эта последняя команда определяет "pick up" относительно составной координаты "object". Как показано выше, значения сложных координат определены комбинацией нескольких составных координат, отделенные "+". Не включайте никаких пробелов между "+" и значениями преобразования. Используя этот метод, Вы можете комбинировать так много составных координат, как необходимо.

Если робот должен взять объект в позиции, указанной как "pick up", определенной относительно "object", программа будет написана следующим образом:

JMOVE plate+object+pickup

или LMOVE plate+object+pickup

(ПРИМЕЧАНИЕ)

1. Не изменяйте порядок, в котором выражено относительное преобразование. Для примера, если составная координата “b” определена относительно координаты переменной “a”, “a+b” ожидаемый результат, но “b+a” не может быть.
2. Данные позы "object" и "pick up" из примера выше определены в отношении к другим данным позиции. Поэтому, не используйте команды типа “JMOVE object”, “LMOVE pick up”, если Вы не уверены относительно ее цели и ее эффекта на программу.

При использовании многократных составных координат, используйте команду POINT, чтобы уменьшить время для вычисления значений составных координат. Например, для того чтобы достичь позиции "pick up" и чтобы двигаться в эту позицию, Вы могли бы ввести:

```
JAPPRO plate + object + pickup, 100  
JMOVE plate + object + pickup
```

Вместо этого, если Вы введете как показано ниже, это будет экономить время вычисления:

```
POINT x = plate + object + pickup  
JAPPRO x, 100  
LMOVE x
```

Эти две программы приводят к одному и тому же движению, но последняя вычисляет значение составной координаты только однажды, так что время выполнения короче, когда команда POINT используется.

В таких простых примерах, различие будет незначительно, но в более сложных программах, это может дать большое различие, и уменьшить полное время цикла.

(ПРИМЕЧАНИЕ)

Для роботов с 7 суставами, обратите внимание на следующее:

1. При использовании команды POINT, обратите внимание на значение JT7. Например, в

POINT p=p1+p2

Значение JT7, присваиваемое в “p” будет значением JT7 в “p2”. Значение крайне правой Переменной в правой стороне выражения присваивается в переменную с левой стороны “=”.

2. При назначении определенного значения в JT7, добавьте “/7” в конце команды POINT.

Например,

```
POINT/7 p = TRANS(,,,,, value)
```

присваивает "value" для значения JT7 переменной "p".

3.6 ЗАДАНИЕ РЕАЛЬНЫХ ПЕРЕМЕННЫХ

Реальные переменные определяются, используя команду назначения (=). Формат для того, чтобы назначить реальную переменную:

Real_variable_name = numeric_value

Пример a=10.5
 count=i*2+8
 Z [2] =Z [1] +5.2

Переменная с левой стороны может быть любой скалярной переменной (то есть единицей счета) или элементом массива (то есть, x [2]). Переменная определена только, когда значение присвоено для нее. Она остается неопределенной до присвоения значения, и если программа выполняется, используя неопределенную переменную, происходит ошибка.

Числовое значение с правой стороны может быть константой, переменной или числовым выражением. Когда команда присвоения обрабатывается, сначала вычисляется значение с правой стороны присвоения, и затем значение присваивается в переменную левой стороны.

Если переменная с левой стороны команды - новая и никогда не назначалась прежде, значение справа присваивается этой переменной автоматически. Если переменная с левой стороны уже определена, новое значение заменит текущее значение. Например, команда "x=3" присваивает значение 3 в переменную "x". Это читается, "присвоение 3 в x", а не "x равен 3". Следующий пример ясно иллюстрирует порядок обработки:

$$x = x+1.$$

Если этот пример - математическое уравнение, и он читается "x, равен x плюс 1", что не имеет смысла.

Как команда присвоения, это читается, "присвоить значение x плюс 1 в x". В этом случае, сумма текущего значения "x" и 1 вычисляется, и затем итоговое значение присваивается в "x" как новое значение. Такое уравнение требует, чтобы x был определен заранее, как указано ниже :

$$\begin{aligned} x &= 3 \\ x &= x+1 \end{aligned}$$

В этом случае, значение результата "x" - 4.

3.7 ЗАДАНИЕ ПЕРЕМЕННЫХ СИМВОЛЬНОЙ СТРОКИ

Переменные символьной строки определяются, используя команду присвоения (=).
Формат для назначения символьной переменной:

`$string_variable=string_value`

Пример `$a1=$a2`
`$error mess [2] = "time over"`

Строковая переменная слева может быть переменной (то есть, `$name`), или элементом массива (то есть, `$line [2]`).

Переменная определена только тогда, когда значение назначено для нее. Остается неопределенной, пока значение не присвоено, и если программа выполняется, используя неопределенную переменную, происходит ошибка.

Символьная строка с правой стороны может быть строковой константой, строковой переменной или строковым выражением. Когда команда присвоения обработана, сначала вычисляется значение с правой стороны, и затем значение присваивается переменной с левой стороны.

`$name = "KAWASKI HEAVY INDUSTRIES LTD."`

В вышеупомянутой команде, строка, включенная в “ ” будет присвоена переменной “`$name`”. Если переменная с левой стороны команды никогда не использовалась прежде, эта строка будет присваиваться автоматически. Если переменная с левой стороны уже была определена, эта команда заменит текущую строку на новую строку с правой стороны.

3.8 ЧИСЛОВЫЕ ВЫРАЖЕНИЯ

Числовые выражения могут состоять из цифр, переменных, определенных функций или других числовых выражений, объединенных операторами. Все числовые выражения, вычисляемые системой, приводятся к значению действительного числа. Числовые выражения могут использоваться иногда вместо числового значения. Они могут использоваться как параметры в мониторинговых командах и программных инструкциях, или как индексы массива.

Интерпретация значения зависит от контекста, в котором выражение появляется. Для примера, выражение, указанное для индекса массива интерпретируется как приведение к целочисленному значению. Выражение, указанное для логического значения интерпретируется как ложь, когда оно оценено как 0, и истина если оно является отличным от 0.

3.8.1 ОПЕРАТОРЫ

Для того, чтобы описывать выражения, используются арифметические, логические, и бинарные операторы. Все операторы комбинируют два значения, чтобы получить одно результирующее значение. Исключения: это два оператора (NOT, COM) работают с одним значением, и оператор (—) оперирует с одним или двумя значениями. Операторы описаны ниже.

Arithmetic Operators	+	Addition	сложение
	-	Subtraction or negation	вычитание, отрицательное
	*	Multiplication	число
	/	Division	умножение
	^	Power	деление
	MOD	Remainder	возведение в степень остаток
Relational Operators	<	Less than	меньше чем
	<=, =<	Less than or equal to	меньше или равно
	=	Equal	равно
	<>	Not equal to	не равно
	>=, =>	Greater than or equal to	больше или равно
	>	Greater than	больше
Logical Operators	AND	Logical AND	логическое И
	NOT	Logical complement	логическое отрицание
	OR	Logical OR	(дополнение)
	XOR	Exclusive logical OR	логическое ИЛИ
Binary Operators	BAND	Binary AND	исключение логического ИЛИ
	BOR	Binary OR	бинарное И
	BXOR	Binary XOR	бинарное ИЛИ
	COM	Complement	бинарное XOR дополнение

(ПРИМЕЧАНИЕ)

1. Относительный оператор “==” - оператор, для того чтобы проверить, равны ли два значения, и отличается от указателя присвоения “=“.
2. Бинарный оператор BOR выполняет операцию OR для соответствующего двоичных битов двух числовых значений. (в этом примере значение выражено в бинарном исчислении, но эта операция может использоваться с любыми системами исчисления).

$\text{^B101000 BOR ^B100001} \rightarrow \text{^B101001}$

Этот результат отличается от того, что Вы можете получить при операции OR.

$\text{^B101000 ИЛИ ^B100001} \rightarrow -1$ (ИСТИНА)

В этом случае, ^B101000 и ^B100001 интерпретируются как логические значения, и т.к. оно не является 0 (ЛОЖЬ), выражение оценено как ИСТИНА.

3.8.2 ПОРЯДОК ОПЕРАЦИЙ

Выражения вычисляются согласно последовательности приоритетов. Приоритет упомянут ниже, от 1 до 14. Обратите внимание, что порядок операций может меняться, используя круглые скобки, чтобы группировать компоненты выражения. Когда выражения, заключенные в круглые скобки вычисляются, сначала вычисляется выражение в пределах самой внутренней пары круглых скобок, и затем обрабатываются внешние пары..

1. Вычисление функции и массивов
- 2.Обработка относительных операторов символьных строк (См. 3.7 Строковые Выражения)
3. обработка оператора возведения в степень “^”
4. обработка унарных операторов “-“ (отрицательное число), NOT, COM

5. Процесс умножения "*" и деления "/" слева направо
6. Вычисление остатка (операция MOD) слева направо
7. Обработка сложения "+" и вычитания "-" слева направо
8. Обработка относительных операторов слева направо
9. Обработка операторов BAND слева направо
10. Обработка операторов BOR слева направо
11. Обработка операторов BXOR слева направо
12. Обработка операторов AND слева направо
13. Обработка операторов OR слева направо
14. Обработка операторов XOR слева направо

3.8.3 ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ

Логические выражения приводят к логическому значению ИСТИНА или ЛОЖЬ. Логическое выражение может использоваться в программе как условие для определения следующей операции в программе. В следующем примере, простое логическое выражение, "x > y", используется в подпрограмме, чтобы определить которой из двух переменных назначить "максимальной" переменной.

```
IF x > y GOTO 10
max=y
GOTO 20
10 max=x
20 RETURN
```

При оценке логических выражений, значение 0 считают ЛОЖЬЮ и все значения отличные от нуля считаются ИСТИНОЙ. Поэтому, все реальные значения или реальные значения выражения могут использоваться как логическое значение.

Например, следующие две инструкции имеют те же самые значения, но вторая инструкция проще для понимания.

```
IF x GOTO 10
IF x <> 0 GOTO 10
```

3.9 СТРОКОВЫЕ ВЫРАЖЕНИЯ

Строковые выражения состоят из символьных строк, строковых переменных, определенных функций или другого строкового выражения, объединенных операторами. Следующие операторы используются со строковыми выражениями.

String operator	+	Combine
Relational operators	<	Less than
	<=, =<	Less than or equal to
	=	Equal to
	<>	Not equal to
	>=, =>	Greater than or equal to
	>	Greater than

Результатом использования строкового оператора будет строка, а для относительных операторов будет реальное значение.

При использовании относительных операторов с символьными строками, строки сравниваются символ за символом из первой символьной строки. Если все символы одинаковы, эти две строки считаются равными, но если есть даже одно различие, строка с символом, имеющим более высокий код символа, оценивается как большая строка. Если одна из строк короче, более короткая считается меньшей. В относительных операциях со строками, пробелы и позиции табуляции расцениваются как символы.

(ПРИМЕЧАНИЕ)

Заглавные и строчные буквы в строковых выражениях считаются разными символами.

4.0 AS ПРОГРАММЫ

Эта глава дает краткий обзор об AS программах. Она объясняет, как создать и выполнить программы, и движения робота. Для лучшего понимания, фактически используйте реальную систему или PC-ROSET*, когда вы читаете эту главу.

ПРИМЕЧАНИЕ* PC-ROSET - имитатор робота персонального компьютера, совместимый с AS системой.

4.0 AS программы

4.1 Типы AS программ

4.2 Создание и редактирование программ

4.3 Выполнение программ

4.4 Процесс выполнения программы

4.5 Движение робота

4.1 ТИПЫ AS ПРОГРАММ

Программа это последовательность программных инструкций, описывающих движение робота, функционирование внешних, внутренних сигналов, различные вычисления. Программа должна быть логически верной, отражать правильный технологический процесс, включать в себя программные элементы, обеспечивающие меры безопасности. Программное имя содержит не более 15 символов, начинается с алфавитной буквы, продолжается буквами, цифрами, периодами. Вы можете создать столько программ, сколько позволяет память контроллера. Обычно программы создаются, используя AS системный режим редактирования, но Вы можете также использовать отдельный компьютер, с загруженным KRterm или KCwin32 программным обеспечением или PC-ROSET и более поздней загрузкой, созданных программ в память робота.

4.1.1 ПРОГРАММА УПРАВЛЕНИЯ РОБОТОМ

Данные программы являются программами, которые управляют движениями робота. Вы можете использовать все программные инструкции AS языка программирования для создания программ.

4.1.2 ПРОГРАММА УПРАВЛЕНИЯ ПРОИЗВОДСТВЕННЫМ ПРОЦЕССОМ (PC PROGRAM)

PC программы или программы управления производственным процессом есть программы, выполняющиеся одновременно с программами управления роботом. PC программы обычно используются для управления и контроля внешним оборудованием посредством контролирования I/O сигналов. PC программы и программы управления роботом могут взаимодействовать друг с другом при помощи общих переменных величин или внутренних сигналов.

PC программы и программы управления роботом используют инструкции совместно. Поэтому, в некоторых случаях, PC программа может быть выполнена как программа управления роботом, несмотря на то, что PC программы не могут использовать инструкции движения, инструкции, которые вызывают движение робота, за исключением инструкции BRAKE. Инструкции BASE и TOOL также невозможно использовать для PC программ.

4.1.3 АВТОСТАРТ

PC программа может быть установлена для автоматического запуска, когда силовое питание включено.

1. Устанавливаем системный переключатель AUTOSTART.PC (AUTOSTART2.PC – AUTOSTART5.PC) в состояние ON.
2. При создании программы, запуск которой происходит автоматически, необходимо программе присвоить имя AUTOSTART.PC (AUTOSTART2.PC – AUTOSTART5.PC)

Некоторые мониторные команды могут быть выполнены в программах при использовании программной инструкции MC т.е. MC CONTINUE.

Ниже дан образец программы автостарта. В этом примере робот контролирует **MOTOR** **POWER** и выполняет программу pg1, когда включается силовое питание. Для более легкого понимания проверки безопасности не введены в эту программу, но в действительности необходимо включить процедуры проверки безопасности.

```
autostart.pc( )  
1 WAIT SWITCH (POWER)  
2 WAIT SIG(27)  
3 MC EXECUTE pg1
```

Ожидает включение силового питания привода
Проверяет, находится ли робот в позиции безопасности (HOME1)
Выполняет программу pg1.

ПРИМЕЧАНИЕ* Обучение позиции безопасности и присвоение приоритетному сигналу HOME1 номер входного сигнала общего назначения 27, выполняется до выполнения программы.

4.2 СОЗДАНИЕ И РЕДАКТИРОВАНИЕ ПРОГРАММ

В этом разделе, показана простая программа, для того чтобы обучить робот выполнить задачу. Программа – список процедур, которые робот будет выполнять. При выполнении программы посредством AS системы, шаги программы обрабатываются сверху донизу и операции, определенные в каждом шаге, выполняются роботом.

4.2.1 AS ПРОГРАММНЫЙ ФОРМАТ

Каждая строка (шаг) AS программного языка выражена в следующем формате.

step number	label	program instruction	;comment
-------------	-------	---------------------	----------

1. Номер шага

2. Номер шага автоматически назначается для каждой строки программы. Шаги пронумерованы последовательно, начиная с 1, и автоматически перенумеровываются всякий раз, когда строки вставляются или удаляются.

3. Метка

Метки используются в программе, чтобы выполнить программный переход. Метка может быть любой целым числом от 1 до 9999, или строкой до 15 алфавитно-цифровых символов, периода или символа подчеркивания (начинается с алфавитного символа), сопровождаемый двоеточием (:). Метки вставлены в начале программной строки, справа после номера шага. Метки могут использоваться как адресат перехода из любого места внутри программы.

4. Комментарий


Точка с запятой (;) указывает, что вся информация направо от точки с запятой - комментарий.

Комментарии не обрабатываются как команды программы, при выполнении программы, и только используются для того, чтобы объяснять содержание программы. Вы можете делать строку программы только с комментариями без меток или инструкций. Пустые строки могут также быть сделаны, чтобы улучшить понимание программы. (Пустая строка состоит из не менее одного пробела или позиции табуляции после точки с запятой.)

4.2.2 КОМАНДЫ РЕДАКТОРА

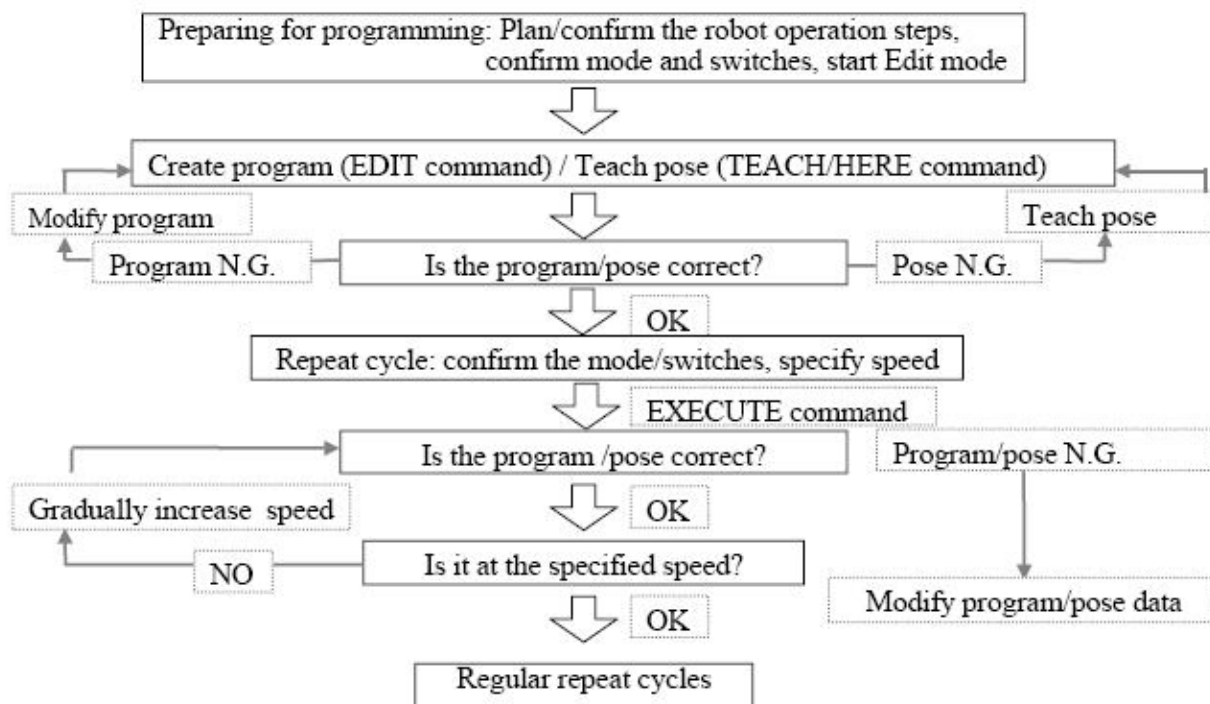
Следующие команды редактора используются, чтобы создать и редактировать программы. (выделенные цветом параметры могут быть опущены.)

рц

EDIT program name, step	Запускает режим редактирования
Program instructions	Переход к новому шагу для ввода инструкций
ENTER key()	Переход к следующему шагу без изменения текущего шага
D step count	Удаление шагов
E	Завершение режима редактирования, переход в мониторный режим
F character string	Поиск символов и отображение этой строки
I	Вставка нового шага
L	Отображение предыдущего шага
M /existing characters /new_characters	Замена существующих символов новыми символами
O	Расположение курсора в текущем шаге для редактирования
P step count	Отображение выбранного количества программных шагов
R character string	Замена символов внутри шага
S step number	Выбор программного шага
XD	Вырезает выбранный шаг или шаги и записывает их в буфер обмена
XY	Копирует выбранный шаг или шаги и записывает их в буфер обмена
XP	Вставляет содержимое буфера обмена
XQ	Вставляет содержимое буфера обмена в обратном порядке
XS	Показывает содержимое буфера обмена
T	Обучение во время режима редактирования

4.2.3 ПРОЦЕДУРЫ ПРОГРАММИРОВАНИЯ

Программирование осуществляется, как показано в следующих шагах:



4.2.4 СОЗДАНИЕ ПРОГРАММ

В программе AS, две вещи нужно обучить для робота:

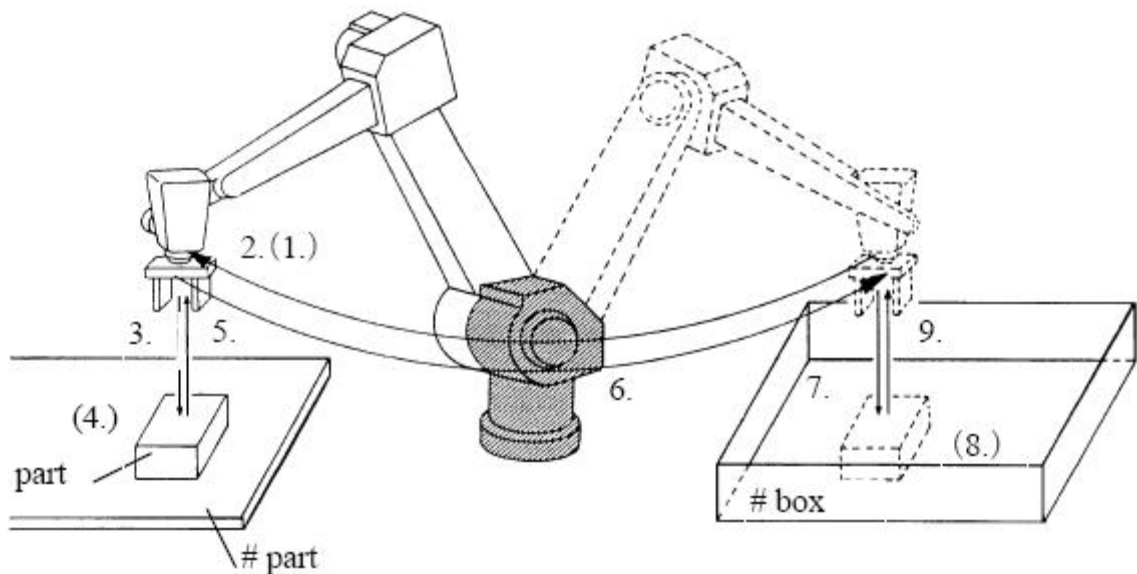
1. Условия работы для робота
2. Траектория, по которой движется инструмент робота

Вот - типовая программа. Робот выполнит задачу, показанную на следующей странице: поднять деталь, подаваемую движущимся транспортером, и переместить ее в коробку.

Сначала определите все движения, требуемые для того чтобы выполнить задачу:

1. Проверить, является ли схват открытым.
2. Переместиться в положение на 50mm выше детали (#part) на транспортере.
3. Переместиться прямо вниз к детали (#part).
4. Взять деталь и закрыть схват.
5. Переместиться прямо вверх на 150mm транспортера..
6. Переместиться в позицию на 200mm выше коробки (#box).
7. Переместить деталь вниз в коробку.
8. Открыть схват и выпустить деталь.
9. Переместиться снова в позицию на 180mm выше коробки.

Переменные #part, #box, которые выражают положение и конфигурацию, называются данными позиции в AS системе. Определите имена переменных позиций как показано в Главе 3 перед выполнением программы..



AS редактор используется, для того чтобы создавать и редактировать программы. Для того чтобы создать программу, названную “demo”, введите “EDIT demo ☐”. появится экран, отображающий следующее

```
:  
> EDIT demo  
.PROGRAM demo  
1 ?
```

Теперь AS ожидает ввода первого шага. Введите “OPENI ☐” после “1?”

```
> EDIT demo  
.PROGRAM demo  
1 ? OPENI ☐  
2 ?
```

Введите следующую инструкцию “JAPPRO #part, 50 ☐” для второго шага

```
> EDIT demo  
.PROGRAM demo  
1 ? OPENI  
2 ? JAPPRO #part, 50 ☐  
3 ?
```

Введите остальную часть программы тем же самым способом. Исправьте любые ошибки при вводе шагов, нажимая **Backspace** перед нажимом ☐.

Если ☐ клавиша нажата в конце ошибочного шага, сообщение об ошибках появляется, и этот шаг бракуется. В этом случае, введите шаг снова. Когда полная программа была введена, экран должен выглядеть следующим образом:

```
>EDIT demo
```



```
.PROGRAM
1 ? OPENI
2 ? JAPPRO #part,50
3 ? LMOVE #part
4 ? CLOSEI
5 ? LDEPART 150
6 ? JAPPRO #box,200
7 ? LMOVE #box
8 ? OPENI
9 ? LDEPART 180
10 ? E ☐
>
```

Последний шаг “E ☐” не является командой для робота, но эта команда завершает режим редактирования (см. также таблицу в 4.2.1). Теперь программа завершена. Когда программ выполняется, AS система выполняет шаги в порядке от шага 1 до шага 9.
См. 11. Примеры программ для более подробной информации об их создании.

4.3 ВЫПОЛНЕНИЕ ПРОГРАММЫ

Управляющие программы робота и РС программы выполняются различными способами.

4.3.1 ВЫПОЛНЕНИЕ УПРАВЛЯЮЩИХ ПРОГРАММ РОБОТА

Для того чтобы выполнить программу установите переключатель **TEACH/REPEAT** в положение REPEAT. Следующее, убедитесь, что переключатель **TEACH LOCK** на пульте ручного управления в положении OFF. Затем включите силовое питание привода **MOTOR POWER** и установите переключатель **HOLD/RUN** в положение RUN.

1. Выполнение программы посредством команды EXECUTE

Во-первых, установите мониторинговую скорость. Робот будет двигаться с этой скоростью при выполнении программы.

Скорость должна быть установлена ниже 30 % с начальной установкой 10%
> SPEED 10 ☐

Для того чтобы начать выполнение, используйте команду EXECUTE. Вид как показано ниже:

> EXECUTE demo ☐

Робот должен затем выполнить поставленную задачу. Если движения робота выполняются не так, как предполагалось, установите переключатель **HOLD/RUN** в положение HOLD. Робот замедлится и остановится. . В случае аварийной ситуации нажмите кнопку **EMERGENCY STOP** на панели контроллера или пульте ручного управления. Тормозная система робота сработает и робот остановится немедленно. Если робот движется корректно со скоростью 10%, увеличьте скорость.

> SPEED 30 ☐

> EXECUTE demo ☐ Робот движется со скоростью 30% .

> SPEED 80

> EXECUTE demo Робот движется со скоростью 80%.

После того, как команда EXECUTE была реализована не менее одного раза, кнопка **CYCLE START** на панели контроллера может использоваться, чтобы выполнить программы.

Чтобы выполнять программу не один раз, введите номер повторений после имени программы:

> EXECUTE demo,5

Выполняет 5 раз

> EXECUTE demo,-1

Выполняет программу непрерывно

2. Выполнение программы посредством команды PRIME

Установите мониторинг скорости таким же образом как с командой EXECUTE, и выполните команду PRIME

>PRIME demo

Робот теперь готов выполнить программу. Нажмите **CYCLE START** на операционной панели, чтобы начать выполнение. Выполнение также может быть начато, используя команду CONTINUE.

3. Выполнение программы через команду STEP или клавишу **CHECK**

Существует возможность проверить движение и содержимое программы при выполнении программы шаг за шагом. Используйте либо мониторинг команды STEP или **CHECK** клавишу* на пульте ручного управления.

ПРИМЕЧАНИЕ* Когда используется клавиша **CHECK**, программное выполнение задерживается в конце каждой инструкции движения.

Во время выполнения управляющей программы робота, некоторые мониторинговые команды запрещены. Более того EXECUTE команда не может быть введена дважды во время выполнения.

4.3.2 ОСТАНОВ ПРОГРАММ

Существует несколько способов для того чтобы остановить программы во время работы. Следующие три описывают порядок от самого срочного до менее срочного.

1. Нажмите кнопку **EMERGENCY STOP** либо на операционной панели, либо на пульте ручного управления. Тормоза сработают и робот немедленно остановится. До тех пор пока нет аварийной ситуации используйте методы 2 или 3.

2. Установите переключатель **HOLD/RUN** на операционной панели в положение HOLD. Робот замедлится и остановится.

3. Ввод команды ABORT останавливает программное выполнение после того, как робот завершит текущий шаг (инструкцию движения).

> ABORT ↵

HOLD команда может также быть использована для останова выполнения.

> HOLD ↵

4.3.3 ВОЗОБНОВЛЕНИЕ ВЫПОЛНЕНИЯ УПРАВЛЯЮЩИХ ПРОГРАММ РОБОТА

В зависимости от того, как робот был остановлен, существует несколько способов возобновления выполнения программ.

1. Когда робот был остановлен при помощи кнопки **EMERGENCY STOP**, разблокируйте кнопку **EMERGENCY STOP**, и нажмите **MOTOR POWER** для того чтобы включить силовое питание привода. Робот начнет движение, когда вы нажмете кнопку **CYCLE START**.

2. Когда переключатель **HOLD/RUN** был использован для останова робота, поверните переключатель в положение RUN для продолжения выполнения программы.

3. Для того чтобы продолжить после ABORT или HOLD команд, или когда программное выполнение было прервано ошибкой, используйте команду CONTINUE. (когда перезапуск происходит после ошибки, ошибка должна быть сброшена до возобновления выполнения программы)

> CONTINUE 

4.3.4 ВЫПОЛНЕНИЕ РС ПРОГРАММ

РС программы выполняются при помощи мониторной команды PCEXECUTE, или при помощи программной инструкции, которая выполняется изнутри управляющей программы робота. PCABORT команда может быть использована для останова выполнения РС программы в любое время. PCEND команда заканчивает выполнение программы после завершения текущего цикла. PCCONTINUE команда возобновляет выполнение программы, прерванной либо командой PCABORT, либо из-за ошибки. (когда перезапуск происходит после ошибки, ошибка должна быть сброшена до возобновления выполнения программы)

4.4 ПРОЦЕСС ВЫПОЛНЕНИЯ ПРОГРАММЫ

Программные инструкции в соответствии с правилами выполняются от верха до низа программы.

Этот последовательный процесс изменяется, когда есть команда типа GOTO или IF.... GOTO. CALL инструкция вызывает и выполняет различные программы, но это не изменяет порядок процесса, когда выполняется команда; RETURN, команда возвращения обработки в программу, вызвавшую другую программу и возобновление выполнения головной программы с места, где она была прервана.

Инструкция WAIT останавливает программу от перехода к следующему шагу до выполнения указанного условия. Инструкции PAUSE И HALT останавливают программы в шаге, где они используются.

Инструкция STOP не может остановить выполнение в некоторых случаях. Если указанные циклы для выполнения остаются, выполнение продолжается с первого шага основной программы. (Даже если инструкция STOP выполнена в подпрограмме, выполнение

возвращается в начало главной программы.), если нет никаких оставшихся циклов, выполнение останавливается в шаге, где применена инструкция.

4.4.1 ПОДПРОГРАММА

Основная программа может быть временно приостановлена и другая программа, названная подпрограммой, может быть вызвана и выполнена. Используя подпрограмму, Вы можете делать программу в модульной структуре, которая проще для понимания.

4.4.2 ПОДПРОГРАММА С ПАРАМЕТРАМИ

Параметры могут использоваться с подпрограммами для более удобного использования. Например, когда вычисления, которые используют различные входные данные, периодически повторяются, создайте подпрограмму, для того чтобы сделать вычисление. Используйте команду CALL, чтобы выполнить переход к подпрограмме, и используйте входные данные как параметры в вычислении. (См. примеры 1,2 ниже)

До 25 параметров могут быть установлены, используя реальные переменные, переменные позиции или строковые переменные. Тип переменной должен быть одинаковым в основной программе и подпрограмме. При присвоении имени параметра к значениям координаты помещается знак "&" перед именем переменной параметра, для того чтобы отличаться от переменных действительного числа. Также, используйте локальные переменные в CALL адресате (подпрограмме).

Пример 1 Значение переменной действительного числа "c" есть сумма вводимых данных "a" и "b".

```
main()  
1 a=1  
2 b=2  
3 CALL calc(a,b,c)  
4 TYPE c
```

```
calc(.aa,.bb,.cc)  
1 .cc=.aa+.bb
```

Пример 2 Значение координаты "c" есть сумма координат "a" и "b".

```
position()  
1 point a = trans(10)  
2 point b = trans(0,20)  
3 CALL add(&a,&b,&c)  
4 point d = c
```

```
add(&.aa,&.bb,&.cc)  
1 point .cc=.aa+.bb
```

4.4.3 АСИНХРОННАЯ ОБРАБОТКА (ПРЕРЫВАНИЕ)

При некоторых состояниях, подобных таким, когда есть ошибка или когда определенный внешний сигнал вводится, выполнение программы может быть прервано, и другая программа будет выполнена. Это происходит независимо от процесса выполнения основной программы и называется асинхронной обработкой (прерыванием). Как только

указанный сигнал (например внешний сигнал или ошибка) обнаружен, прерывание происходит независимо от выполнения основной программы. Этот процесс активизируется, используя инструкцию ON (или ONI)... CALL.

4.5 ДВИЖЕНИЕ РОБОТА

4.5.1 СИНХРОНИЗАЦИЯ ДВИЖЕНИЯ РОБОТА И ВЫПОЛНЕНИЯ ШАГА ПРОГРАММЫ

В AS системе, синхронизация выполнения программы и движения робота может быть изменена при помощи установки системных переключателей. Для примера, синхронизация выполнения шага изменяется следующим образом, когда системный переключатель PREFETCH.SIGINS становится ON (разрешает более раннюю обработку команд сигналов I/O) или OFF (запрещает более раннюю обработку команд сигналов I/O)

```
JMOVE part1
SIGNAL 1
JMOVE part2
SIGNAL 2
```

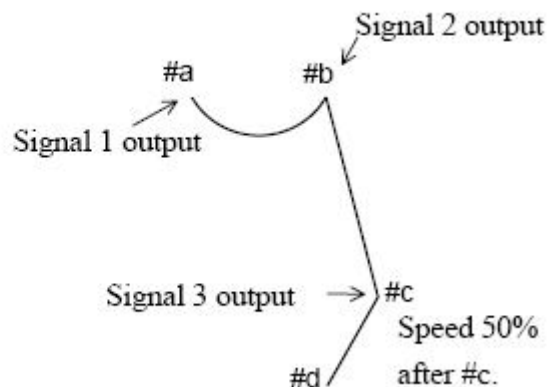


Когда PREFETCH.SIGINS в состоянии ON, внешний сигнал 1 (SIGNAL 1) выводится как только робот начинает движение в точку part1. когда программа переходит ко второй инструкции JMOVE, она ожидает пока робот не достигнет part1 до выполнения этой инструкции. Как только робот достигнет part 1, он начинает двигаться к part 2, и в это же самое время выводится внешний сигнал 2 (SIGNAL 2).

когда PREFETCH.SIGINS в состоянии OFF, сигналы выводятся после того как робот достигнет координату инструкции движения и оси совместятся.

Пример ниже демонстрирует, как программные шаги выполняются в AS системе, когда PREFETCH.SIGINS в состоянии ON.

```
1 JMOVE    #b
2 SIGNAL   1
3 a=2
4 LMOVE    #c
5 SIGNAL   2
6 SPEED    50
7 LMOVE    #d
8 SIGNAL   3
```



Если вышеупомянутая программа выполняется, когда робот - в #а, шаги продолжаются в следующем порядке:

1. В #а, робот планирует движение JMOVE #b и начинает двигаться по направлению к #b.
2. Как только начинается движение, следующий шаг, SIGNAL 1, выполняется, то есть, сигнал 1 включается сразу, после того как робот отходит от #а.
3. Выполнение переходит в шаг 4, планируется LMOVE #с, и ожидается, когда робот достигнет #b.
4. Как только робот достигает #b, он начинает двигаться по направлению к #с. Выполнение переходит к шагу 7 (планируется движение LMOVE #d), и ожидается, когда робот достигнет #с.

(ПРИМЕЧАНИЕ)

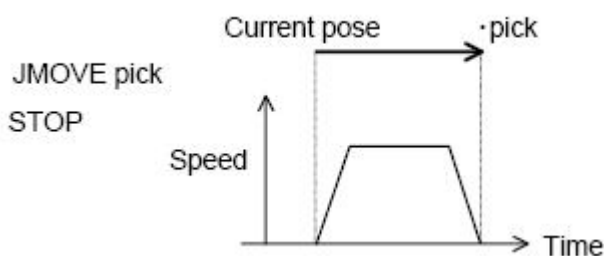
Когда PREFETCH.SIGINS находится в состоянии ON, программа обрабатывает следующий шаг, не дожидаясь пока робот достигнет указанной позиции. Однако, синхронизацию затрагивают другие параметры настройки и команды/инструкции типа команды WAIT или переключателя CP. WAIT команда приостанавливает обработку шагов, пока данное состояние не удовлетворено. Когда переключатель CP в состоянии OFF, программа обрабатывает все шаги до шага, который включает инструкцию движения, и останавливается там перед продолжением. Сохраните в примечании параметры настройки системных переключателей и команды при программировании.

Как демонстрируется здесь, важно обратить внимание, что параметры настройки системных переключателей и некоторые программные инструкции затрагивают синхронизацию, в которой, AS система обрабатывает программу и движения робота. Обратите пристальное внимание на синхронизацию выходных сигналов во время программирования.

Для подробностей относительно каждого системного выключателя, обратитесь к 7.0 или к «Руководство Оператора».

4.5.2 НЕПРЕРЫВНАЯ ТРАЕКТОРИЯ (CP) ДВИЖЕНИЯ

Этот пример показывает выполнению одной инструкции движения.

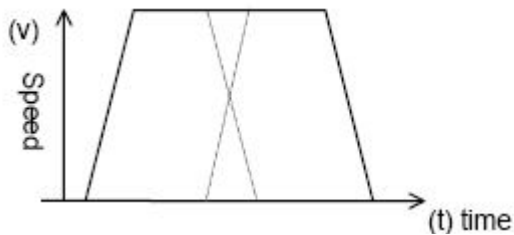


При выполнении инструкции движения подобно показанной выше, робот ускоряется плавно до установки текущей скорости, при движении по направлению к позиции "pick". Поскольку робот приближается к "pick", он постепенно замедляется, пока не останавливается в позиции. Серия движений типа этого, совершенной одной инструкцией движения, называют "сегментом движения".

В случае рисунка ниже, если переключатель системы CP находится в состоянии ON, робот сначала ускоряется, чтобы достигнуть указанную скорость, но не замедляется, когда приближается к pos.1. Вместо этого, робот делает плавный переход для движения по

направлению pos. 2. Когда робот приближается к pos. 2, он постепенно замедляется и останавливается в этой точке. Это движение состоит из двух команд движения, и - таким образом состоит из двух сегментов движения.

```
JMOVE pos.1  
JMOVE pos.2  
STOP
```



Движение подобное этому, где робот выполняет ряд движений, делая плавный переход между сегментами движения, не останавливаясь в каждой координате, называется СР (Непрерывная траектория) движением. Выключение системного переключателя СР отключает функцию СР. Если переключатель СР выключен, робот замедляется и останавливается в конце каждого сегмента движения.

(см. 5.6 Команда SWITCH и 6.9 инструкция ON/OFF, для того как установить переключатель СР)

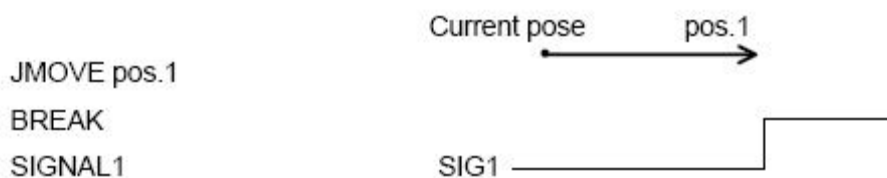
Движения СР могут использоваться и в прямолинейных движениях, и в угловых интерполированных движениях или в комбинации из них. Например, движения СР могут использоваться для всех следующих шагов:

прямолинейное движение (например. LDEPART) → угловое интерполированное движение (например. JAPPRO) → прямолинейное движение (например. LMOVE).

4.5.3 ПРЕРЫВАНИЕ СР ДВИЖЕНИЯ

Некоторые инструкции могут приостановить выполнение программы, пока робот фактически не достигает координаты. Это называют прерыванием СР движения. Эти инструкции полезны, когда робот должен быть неподвижен, в то время как некоторые операции выполняются (например, закрытие схвата).

См. пример ниже.



Инструкция JMOVE начинает движение робота по направлению к pos.1. Далее выполняется BREAK инструкция. Эта инструкция задерживает выполнение программы до тех пор пока движение в позицию pos.1 не завершится. В этом случае внешний сигнал не выходит до тех пор пока робот не остановится.

Следующие инструкции приостанавливают выполнение программы до тех пор пока движение робота не завершилось полностью.

Однако, будьте внимательны, для того чтобы не использовать эти инструкции, когда робот должен двигаться.

BASE	BREAK	BRAKE	CLOSEI	HALT	OPENI	PAUSE	RELAXI
TOOL	ABOVE	BELOW	DWRIST	UWRIST	LEFTY	RIGHTY	

В добавление к выше сказанному, ONI инструкция также прерывает выполнение программы, но обратите внимание, что прерывание, установленное при помощи ONI инструкции может произойти в любом месте сегмента движения.

4.5.4 ОТНОШЕНИЕ МЕЖДУ ПЕРЕКЛЮЧАТЕЛЕМ CP И ИНСТРУКЦИЯМИ ACCURACY, ACCEL, И DECEL

·ACCURACY инструкция ... Устанавливает точность позиционирования робота в конце каждого сегмента движения.

(Когда робот входит в диапазон, установленный этой командой, он считает, что достиг точки, и начинает движение в следующую точку)

(ПРИМЕЧАНИЕ)

1. Робот замедляется и останавливается, если команда не дана перед выполнением завершения текущего движения. Некоторые из причин, которые вызывают такую ситуацию:

(1) Команда WAIT выполнена, но условия, для того чтобы продолжить программу, не установлены прежде, чем движение робота закончено.

(2) Шаги программы перед следующей инструкцией движения не закончены до окончания текущего движения.

2. При перемещении в движение CP, требуется, чтобы некоторое время вычислялся переход между сегментами движения. Поэтому, если расстояние между двумя позициями очень небольшое, вычисление не может быть закончено перед достижением второй позиции, таким образом, происходит останов робота между двумя сегментами движения. Чтобы избежать этого, необходимо уменьшить скорость. Если скорость не должна быть изменена, не задавайте позиции излишне близко друг к другу.

·ACCEL команда Устанавливает ускорение робота в начале движения.

·DECEL команда Устанавливает замедление робота в конце движения.

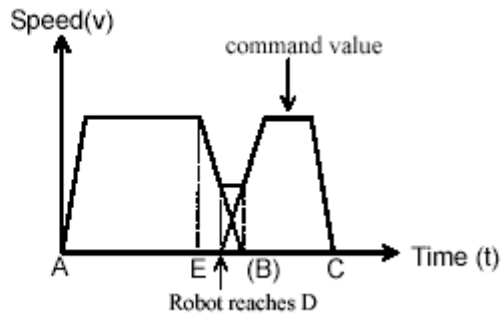
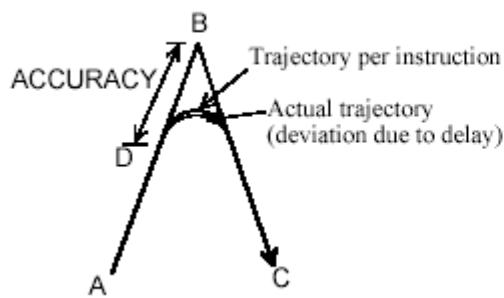
·CP переключатель , разрешает или запрещает движение CP.

4.5.4.1 CP ON ...ТИП СТАНДАРТНОГО ДВИЖЕНИЯ

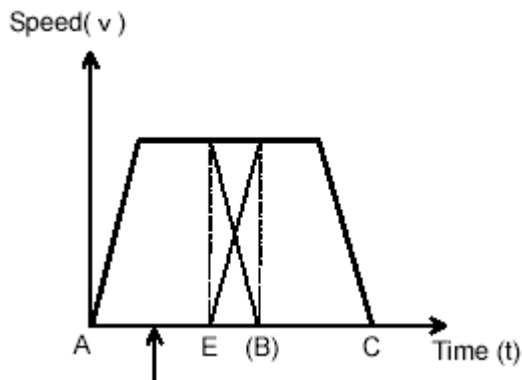
Для примера, робот делает выполнение движения, показанного ниже с CP переключателем в состоянии ON: $A \rightarrow B \rightarrow C$.

Как только текущее значение позиции попадает в диапазон точности (т.е. робот достигает точки D), наложение начинается от значений текущего движения со значением команды движения для следующей траектории.

Робот будет перемещаться непрерывно по направлению следующей траектории в соответствии с командными значениями.

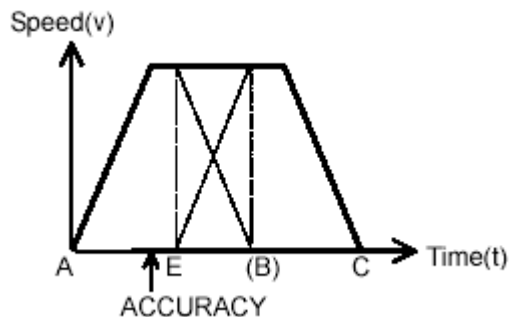


Чем больше диапазон, определяющий точность, тем раньше начинается наложение. Однако ускорение для следующей траектории не начинается раньше точки, где робот начинает замедление (точка E), следовательно можно говорить, что эффект точности достигается в определенной величине, т.е. нет эффекта в установке значений точности большем, чем расстояние между B и E.

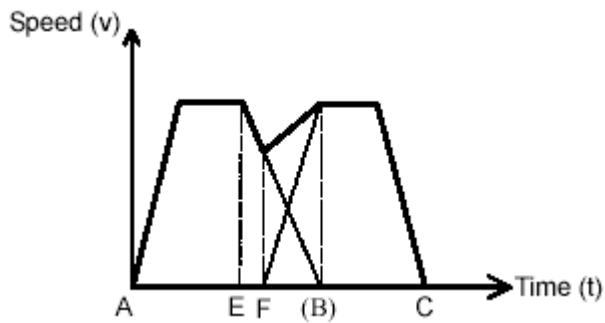


Даже, если командная величина достигает точки точности в это время, ускорение для следующей траектории не может начаться до замедления, начинающегося в точке E.

Если значения ускорения и замедления установлены достаточно маленькими, наложение начинается раньше, и робот движется по траектории с большим радиусом, но общее время для достижения точки C отличается незначительно.



Даже если замедление уменьшено, а ускорение для следующей траектории увеличено, совместная скорость не будет превышать определенную максимальную скорость, т.к. наложение не начнется до достижения роботом точки F (точки, где начинается ускорение). Иными словами, время для того, чтобы выполнить полностью замедление и ускорение одно и то же (точка B).



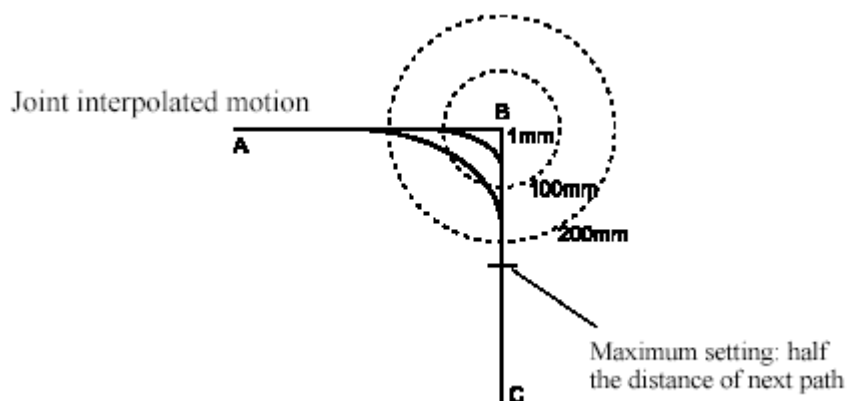
4.5.4.2 CP ON ... ТИП ДВИЖЕНИЯ 2

В типе движения 2, понятие ускорения и скорости в линейном движении и круговом движении отличаются от стандартного типа движения. Стандартный тип движения и тип движения 2 могут использоваться в одинаковых программах без изменений, но реальная траектория движения и скорость движения будут меняться.

1. Установка точности

Точность в угловом интерполированном движении

Траектория движения робота, соответствующая установленной точности, показана на рисунке, изображенном ниже. В примере значения точности в точке В - 1 мм, 100 мм, 200 мм. Тем же самым способом, как стандартное движение, робот начинает движение по кратчайшему расстоянию до достижения точки В, но не обязательно начинает разворот в точке, где он входит в диапазон точности. Как близко робот подойдет к точке В до начала разворота, определяется при помощи угла каждой оси, вычисленного пропорционально значению точности. При установке значения точности достаточно большим, робот может пройти по кратчайшей траектории кратчайшим расстоянием либо оставшееся расстояние текущей траектории, либо половину расстояния следующей траектории от В до С.

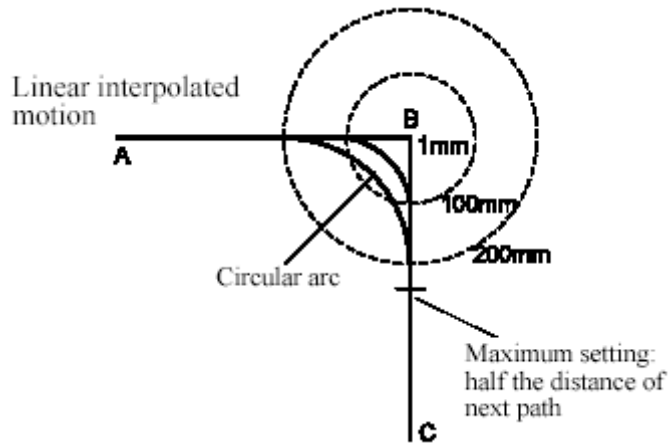


2. Точность линейного и кругового интерполированного движения

Траектория движения робота, соответствующая установленной точности, показана на рисунке, изображенном ниже. В примере значение точности в точке В - 1 мм, 100 мм, 200 мм. Робот начинает разворот в точке, где он вошел в диапазон точности. Робот движется по круговой траектории внутри радиуса диапазона точности. При установке значения точности достаточно большим, робот может пройти по кратчайшей траектории

кратчайшим расстоянием либо оставшееся расстояние текущей траектории, либо половину расстояния следующей траектории от В до С.

Величина точности может быть установлена равной половине расстояния второй траектории.



При движении по кратчайшему пути, время цикла может быть уменьшено. Тем не менее, когда следующие состояния установлены, обработка установки точности будет такой же, как и в стандартном движении:

- когда инструкции ожидания выполняются в точке В;
- когда work/tool изменяются в точке В;
- когда режим интерполяции меняется на посуставную (угловую) интерполяцию;
- когда модель движения изменяется в точке В (обычная модель ↔ движение, базирующееся на фиксированной инструментальной системе координат).

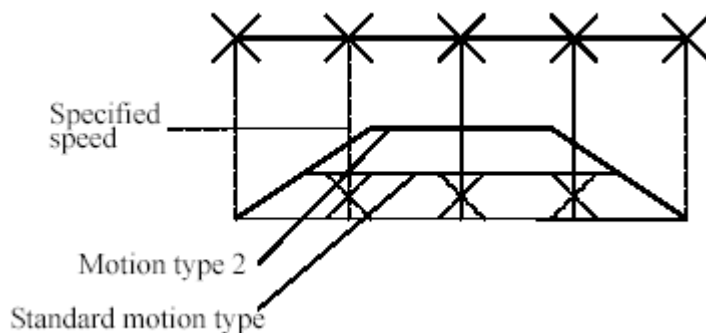
2. Установка скорости

Скорость интерполированного движения суставов

Такая же, как в стандартном типе движения.

Скорость в линейном и круговом интерполированном движении

В типе движения 2, если значение точности установлено большим и конфигурация робота не изменяется между двух позиций, заданная скорость достигается, даже если расстояние между двумя позициями мало.



Тем не менее, когда следующие состояния установлены, обработка установки точности будет такой же, как и в стандартном движении:

- когда инструкции ожидания выполняются в точке В;

- когда work/tool изменяются в точке В;
- когда режим интерполяции меняется на суставную интерполяцию;
- когда модель движения изменяется в точке В, из обычного режима (деталь фиксирована, инструмент движется) в режим фиксированных инструментальных размеров.

(ПРИМЕЧАНИЕ)

При попытке выполнить программу, где изменение положения робота происходит в пределах короткого расстояния, время для осуществления изменения положения будет превышать расчетное время для прохождения этого расстояния с заданной скоростью. В этом случае отдается приоритет движению в суставах (угловому движению), соответственно изменение позиции не может быть осуществлено с заданной скоростью.

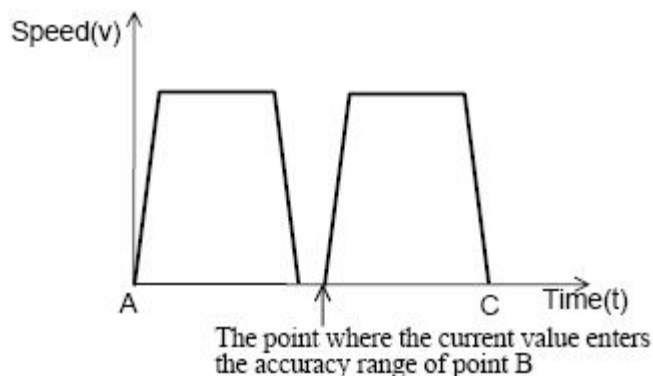
3. Скорость при круговой интерполяции

В типе движения 2 максимальная скорость автоматически устанавливается соответственно способности робота выполнить характерное круговое интерполированное движение. Робот следует круговой траектории внутри окружности диапазона точности. Максимальная скорость прохождения этой траектории также устанавливается возможностями робота.

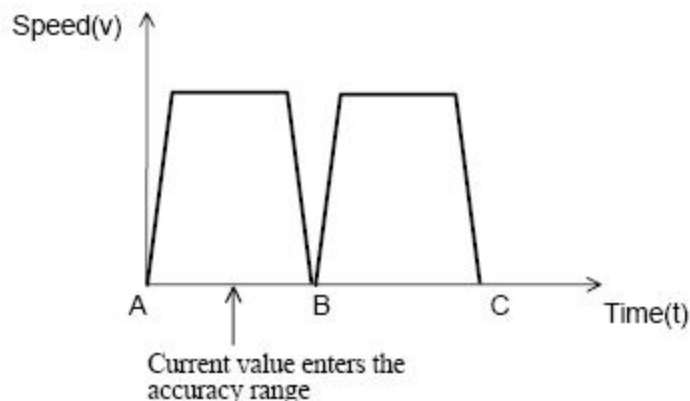
4.5.4.3 CP OFF

Когда переключатель CP находится в состоянии OFF, наложение движений не происходит. Ускорение для второго отрезка начинается после того как первый сегмент движения закончен и текущее значение находится в диапазоне ACCURACY*.

ПРИМЕЧАНИЕ* Для примера, для FS10, значение по умолчанию 1 мм



Когда переключатель CP находится в состоянии OFF, движения по второму отрезку начнется только тогда, когда скорость замедления первого движения достигнет 0, даже если диапазон точности больше чем окончание первого отрезка.



4.5.5 ДВИЖЕНИЕ ПО УКАЗАННОЙ ТРАЕКТОРИИ

Прямолинейные интерполяции и угловые интерполяции движения - стандартные функции для всех роботов.

Однако, иногда необходимо переместить робот по указанному или расчетному пути.

AS система может выполнять вычисления, в то время как робот двигается, делая всевозможные, сложные движения.

Эту особенность называют “Движением по указанной траектории”.

Система разрешает выполнение движений через программный цикл, который выполняет ряд непрерывных вычислений для движения на близкие расстояния, в то время когда выполняются инструкции движения. Такой цикл программы возможен, потому что AS может выполнить инструкции, несвязанные с движением, в то время как робот перемещается. Вычисленные сегменты движения плавно стыкуются, используя функцию CP.

Следующее - пример программы для движения по указанному пути. Инструмент робота будет следовать по траектории, определенной серией данных позиции, указанных переменной типа массив "path".

```
FOR index=0 TO 10  
LMOVE path[index]  
END
```

path[0] - path[10] должны быть обучены ручным методом либо при помощи вычисления.

4.5.6 УСТАНОВКА ДАННЫХ НАГРУЗКИ

При установке данных нагрузки для текущего движения робота, оптимальное ускорение и замедление при нагрузке определяются автоматически. Установите правильные данные нагрузки согласно текущему движению робота.

CAUTION

Всегда устанавливайте правильную массу загрузки и центр тяжести. Неправильные данные могут ослабить или сократить долговечность частей или вызвать перегрузку, ошибочные отклонения. Для детальной информации см. команду/инструкцию WEIGHT.

Данные нагрузки могут быть установлены автоматически, используя вспомогательную функцию 0406

См. Руководство Оператора для подробностей.

5.0 МОНИТОРНЫЕ КОМАНДЫ

Эта глава группирует мониторные команды в следующих категориях, и описывает каждую команду подробно. Мониторная команда состоит из ключевого слова, выражающего команду и параметра (ов) после этого ключевого слова, как показано в примере ниже.

- 5.1 Команды редактирования
- 5.2 Команды управления программой и данными
- 5.3 Команды сохранения программ и данных
- 5.4 Команды управления программой
- 5.5 Команды позиционной информации
- 5.6 Команды управления системой
- 5.7 Команды бинарных сигналов
- 5.8 Команды отображения сообщений

Пример



Параметры, отмеченные цветом, могут быть опущены
Всегда вводите пробел между ключевым словом и параметром

5.1 КОМАНДЫ РЕДАКТИРОВАНИЯ

EDIT	Запускает программу редактора.
C	Заканчивает редактирование текущей программы и заменяет ее другой программой (Change).
S	Выбирает программный шаг для отображения (Step).
P	Отображает заданное количество программных шагов (Print).
L	Выбирает предыдущий шаг (Last).
I	Вставляет новый шаг (Insert).
D	Удаляет программный шаг (Delete).
F	Поиск для символов (Find).
M	Замена символов (Modify).
R	Замена символов (Replace).
O	Размещение курсора в текущем шаге (One line).
E	Завершение редактирования (Exit).
XD	Вырезание и запись выбранного шага (ов) в буфер обмена.
XY	Копирование и запись выбранного шага (ов) в буфер обмена.
XP	Вклеивание содержимого буфера.
XQ	Вклеивание содержимого буфера в обратном порядке.
XS	Показ содержимого буфера.
T	Обучение инструкций движения в режиме редактирования (опция)

EDIT `program name, step number`

Назначение

Входит в режим редактирования, что позволяет создавать и редактировать программы.

Параметры

Программное имя

Определяет программу для редактирования. Если программное имя не задано открывается последняя программа, вызванная для редактирования. Для написания новой программы задается новое имя для создаваемой программы.

Номер шага

Выбирает номер шага, с которого начинается редактирование. Если номер шага не задан, редактирование начинается с шага, на котором был сделан выход из режима редактирования.

(ПРИМЕЧАНИЕ)

Программа не может редактироваться во время выполнения программы. Программа не может быть выполнена или удалена во время редактирования. Если вызывается программа, редактируемая в настоящее время, происходит ошибка.

C `program name, step number`

Назначение

Заменяет программу, в настоящее время находящуюся в редакторе на другую

Программное имя

Определяет программу для редактирования.

Номер шага

Выбирает номер шага, с которого начинается редактирование. Если номер шага не задан, редактирование начинается с первого шага

S `step number`

Назначение

Выбирает и отображает шаг программы, выбранный для редактирования

Параметр

Номер шага.

Если номер шага не задан, происходит переход на первый шаг программы. Если номер шага больше чем шагов программы, происходит переход на шаг, следующий за последним шагом программы.

P step count

Назначение

Отображает число шагов программы, заданных в параметре, начиная с текущего шага.

Параметр

Количество шагов

Задаёт количество шагов для просмотра. Если количество шагов не задано, отображаются все шаги, следующие за текущим шагом.

Объяснение

Отображается только указанное количество шагов. Последний шаг в списке готов к редактированию.

L

Назначение

Отображает предыдущий шаг.

I

Назначение

Вставляет шаг перед текущим шагом. Заканчивается ввод данной команды, нажатием клавиши Enter. Все команды, заданные до выхода из режима вставки, вводятся в программу.

Объяснение

Шаги после вставленной строки перенумеровываются. Чтобы выйти из режима вставки, нажмите клавишу **ENTER**. Все строки, вставленные перед выходом из режима вставки, вставлены в программу.

Пример

CLOSEI инструкция вставлена между шагами 3 и 4.

```
1?OPENI
2?JAPPRO #PART, 500
3?LMOVE #PART
4?LDEPART 1000
5? S 4 ;
4 LDEPART 1000
4? I ;Type the I command.
4I CLOSEI ;
5I ↵ ;
5 LDEPART 1000 ;Step 4
5?
```

отображает 4 шаг для вставки шага перед ним.

напечатать команду вставки

напечатать инструкции для вставки.

нажать ввод для завершения вставки шагов.

шаг 4-ый стал пятым.

(ПРИМЕЧАНИЕ)

Для того чтобы вставить линию пробелов, нажмите Spacebar или TAB , затем Enter, пока вы находитесь в режиме вставки.

D step count

Назначение

Удаляет заданное число шагов, следующих за текущим шагом.

Параметр

Количество шагов

Задаёт число шагов для удаления, начиная с текущего шага. Если количество шагов не задано, удаляется только текущий шаг. Если заданное количество шагов больше шагов программы, удаляются все шаги, следующие за текущим шагом. После удаления все оставшиеся шаги автоматически перенумеровываются и сдвигаются вверх.

Объяснение

Удаляет только указанное количество шагов, начинающихся с текущего шага. После того, как шаги удалены, все остающиеся шаги перенумеровываются и автоматически продвигаются и отображаются.

(ПРИМЕЧАНИЕ)

Если количество шагов указано больше чем количество шагов в программе, все шаги после текущего шага удалены.

F character string


Назначение

Отыскивает в программе с текущего шага до последнего символьную цепочку, и переходит на первый шаг программы, содержащий в себе символьную цепочку.

Параметр

Символьная цепочка

Задаёт последовательность символов для нахождения и перехода на шаг программы, содержащего эти символы.

1?F abc 

3 JMOVE abc

3?

M/exiting characters/new characters

Назначение

Изменяет символы текущего шага.

Параметр

Введенные ранее символы

Определяет, какие символы должны быть перезаписаны в данной строке.

Новые символы

Задаёт, какие символы должны заменить ранее введенные символы.

Пример заменяет координату abc на def

```
4      JMOVE abc
4?M/abc/def 
4      JMOVE def
4?
```

R character string

Назначение

Заменяет ранее введенные символы в данном шаге на последовательность заданных символов.

Параметр

Символьная цепочка

Задаёт новые символы для замены ранее введенных символов.

Пояснение

Последовательность действий для использования R команды следующая:

1. Используя клавишу пробел , установить курсор под первым символом, который необходимо изменить.
2. Нажимаем клавишу и затем
3. Вводим новый символ (ы) замены. Обратите внимание, что введенные символы не заменяют символы над курсором, но два пробелов слева, начинающихся над R. (См. пример ниже)
4. Нажимаем

Т.к. клавиша Enter нажата, AS система проверяет корректна ли линия, если есть ошибка ввод игнорируется

Например

Изменить скорость движения с 20% на 35%, используя R команду.

```
1. SPEED 20, ALWAYS
1?      R 35 (Enter)
1. SPEED 35, ALWAYS
1?
```

О

Назначение

Устанавливает курсор за последним символом выбранного шага.

Например

Изменить имя точки aa1 на cc1, используя O команду.

```
3.JMOVE aa1
3?O
3.JMOVE aa1(BackSpace) ; удаляем aa1, используя клавишу BS
3.JMOVE cc1(Enter)      ; вводим cc1
3.JMOVE cc1
3.?
```

(ПРИМЕЧАНИЕ)

Эта команда не может быть использована с пульта ручного управления

Е

Назначение

Выход из режима редактирования и возврат в мониторный режим

XD **step count**

Назначение

Вырезает заданное количество шагов из программы и записывает их в отдельный буфер.

Параметр

Количество шагов

Задаёт количество шагов для отсекаания и записи в отдельный буфер, начиная с текущего шага. Если параметр не задан, в буфер записывается только текущий шаг. До десяти шагов может быть отсечено.

Объяснение

Вырезает указанный номер шагов и сохраняет их в буфере.

Команда XY копирует, но не вырезает шаги, но команда XD вырезает шаги. Остающиеся шаги в программе перенумеровываются соответственно.

XY step count

Назначение

Копирует заданное количество шагов, включая текущий шаг, и записывает их в буфер вставки.

Параметр

Количество шагов

Задаёт число шагов для копирования и записи в буфер. Может копироваться до десяти шагов. Если параметр не задан, копируется только текущий шаг.

Объяснения

Копирует указанное количество шагов, включая текущий шаг и сохраняет их в буфер вставки.

Команда XD вырезает шаги, а XY копирует шаги. Программа остается той же самой и нумерация шагов не изменяется после того, команда XY была использована.

XP

Назначение

Вставляет содержимое буфера до текущего шага. Эта команда работает только тогда, когда в буфере есть содержимое.

Объяснение

Используйте XD и XY до этой команды, чтобы сохранить желательное содержание в буфере вставки.

XQ

Назначение

Вставляет содержимое буфера перед текущим шагом в обратном порядке.

Объяснение

Вставляет содержание буфера в обратном порядке тому, как это было бы вставлено, используя XP команду.

XS

Назначение

Отображает содержимое буфера.

Если буфер пустой, ничего не отобразится.

T `variable name`

Назначение

Разрешает режим обучения инструкций движения (JMOVE, LMOVE.), используя ПРУ, в режиме редактирования программы.

Параметр

Имя точки

Присваивает обученной координате имя, заданное в параметре с последовательным присвоением цифр от нуля и далее, автоматически вводит команду движения в обученную точку (декартовую или угловую). Если имя точки задано в виде A[], координаты точки задаются переменной типа «массив». В этом случае переменные не могут быть использованы в числах элементов. Если имя точки не задано, обучаемая координата запоминается, как угловая координата, без присвоения имени точке, т.е. как постоянная.

Объяснение

Введите эту команду во время режима редактирования. Когда команда выполнен, на пульте ручного управления появится специальный экран обучения. Движения, обученные здесь зарегистрированы, как команды в программе, и записываются в шаге, где команда T введена. Когда больше чем один шаг обучается, переменная переименовывается, увеличивая последнее число в имени переменной. См. Руководство оператора для большего количества подробностей.

Например

С параметром

2. JAPPRO #a

? T pos

3. JMOME pos0

<обучение, используя ПРУ. Нажмите Cancel для того чтобы вернуться в AS> (здесь обучается три шага)

4. JMOVE pos1

5. LMOVE pos2

и т.д.

Без параметра

2. JAPPRO #a

? T

<обучение, используя ПРУ. Нажмите Cancel для того чтобы
вернуться в AS> (здесь обучается два шага)

3.

4.

3 JMOVE #[0,10,20,0,0,0]

4 JMOVE #[10,10,20,0,0,0]

И т.д.



WARNING

Пульт ручного управления должен быть подключен к контроллеру, для того чтобы использовать эту команду. Также робот должен быть в режиме обучения, и TEACH LOCK в состоянии ON.

5.2 КОМАНДЫ УПРАВЛЕНИЯ ПРОГРАММОЙ И ДАННЫМИ

CARD_FDIR*	Списки имен программ и переменных на PC карте.
LIST	Отображает все программные шаги и значения переменных.
LIST/P	Отображает все программные шаги .
LIST/L	Отображает все координаты и их значения.
LIST/R	Отображает все реальные переменные и их значения.
LIST/S	Отображает все строковые переменные и их данные.
DELETE	Удаляет программы и переменные из памяти робота.
DELETE/P	Удаляет программы из памяти робота.
DELETE/L	Удаляет координаты из памяти робота..
DELETE/R	Удаляет реальные переменные из памяти робота .
DELETE/S	Удаляет строковые переменные из памяти робота .
CARD_FDEL *	Удаляет программы и переменные из из PC карты.
CARD_VERIFY *	Устанавливает ON/OFF функции проверки работы передачи в PC карту.
RENAME	Меняет имя программы.
XFER	Копирует шаги одной программы в другую.
COPY	Копирует программы.
CARD_COPY*	Копирует программы на PC карте.
TRACE	Устанавливает ON/OFF функции TRACE (опция)
SETTRACE	Резервирует память для регистрирования (опция)
RETRACE	Освобождает память, зарезервированную SETTRACE. (опция)
LSTRACE	Отображает зарегистрированные данные (опция)

ПРИМЕЧАНИЕ* Эти команды используются для того чтобы управлять памятью PC карты, но такие же команды работают с дискетами, если CARD_ заменить на FD_. Смотрите пояснения для каждой команды для более полного понимания.

CARD_FDIR

FD_FDIR

Назначение

Отображает каталог имен файлов, записанных соответственно на PC карте или на дискете. CARD_FDIR обращается к данным на PC карте, и FD_FDIR обращается к данным на дискете.

Пояснение

При использовании CARD_FDIR и FD_FDIR команд, отображаются все подпрограммы и переменные, используемые в программе.

```
>FD_FDIR [↵]
```

```
>CARD_FDIR
```

Когда переключатель SCREEN в ON, отображение не прокручивается и останавливается в конце экрана. Для того чтобы продолжить отображение нажмите **[Spacebar]**. Для того чтобы закончить нажмите **[Enter]**.

(ПРИМЕЧАНИЕ)

Имена программ и переменных с * or ~, отображенных в начале имени, поясняют что содержимое этих программ или переменных не являются пока что определенными

```
LIST program name, .....  
LIST/P program name,.....  
LIST/L pose(location) variable,.....  
LIST/R real variable,.....  
LIST/S string variable,.....
```

Назначение

Выводит на экран листинг заданных программ или листинги других данных.

Параметр

Имя программы (/P), имя точки (/L), реальные переменные (/R), строковые переменные (/S).

Задаёт тип данных для вывода на экран. Если параметр не задан, выводятся все данные, содержащиеся в памяти. Если выбирается переменная массива, все элементы массива отображаются на экране.

Пояснение

Команда LIST отображает все программные имена, их подпрограммы и переменные. Но с другой стороны, LIST/P команда отображает содержимое только главной команды.

>LIST 

>LIST/L 

>LIST/R 

>LIST test* 

Если листинг содержит информации больше, чем вмещает экран и переключатель SCREEN находится в положении ON, вывод информации приостанавливается при заполнении экрана. Продолжить вывод информации можно при помощи клавиши ПРОБЕЛ, прекратить вывод информации можно при помощи клавиши ENTER.

DELETE program name,.....
DELETE/P program name,.....
DELETE/L pose variable,.....
DELETE/R real variable [array elements],.....
DELETE/S string variable [array elements],.....

Назначение

Удаляет заданные данные из памяти контроллера

Параметры

Имя программы (/P), имя точки (/L), реальные переменные (/R), строковые переменные (/S).

Пояснение

Команда DELETE удаляет заданную программу полностью, т.е. если программа является головной программой, не используется как подпрограмма, удаляется она сама и все данные, используемые в программе:

все подпрограммы, координаты точек и их имена, заданные как в головной программе, так и в подпрограммах, переменные, строковые переменные. (Однако, если данные используются в других программах, они не удаляются.)

В случае, когда удаляемая программа является подпрограммой, необходимо сделать ее головной программой, иначе она не удалится.

DELETE/P команда, в отличие от команды DELETE, удаляет только заданную программу, не удаляя данные, используемые в этой программе.

Если элементы массива не определены с DELETE/R и DELETE/S командами, все элементы массива будут удалены. Если элементы массива заданы, удаляются только эти элементы.

Например

>DELETE test

Удаляет программу “test” и все подпрограммы и переменные, используемые в ней.

>DELETE/P pg11,pg12

Удаляет только программы под именами “pg11”, “pg12”

>DELETE/R a

Удаляет элементы массива переменных под именем a

>DELETE/R a[10]

Удаляет десятый элемент массива a

CARD_FDEL file name,.....

FD_FDEL file name,.....

Назначение

Удаляет заданные файлы, соответственно с PC карты или дискеты.

Параметры

Имя файла.

Задаёт имя файла для удаления

Объяснение

Команды CARD_FDEL И FD_FDEL удаляют программы в указанном файле полностью; то есть, основная программа непосредственно и, если следующие данные используется в программе, они также удаляются.

(Однако, данные, используемые в других программах не удаляются).

- Все подпрограммы, вызываемые программой или подпрограммами в пределах этой программы.
- Все переменные позиции используемые в программе и в подпрограммах в этой программе.
- Все реальные переменные, используемые в программе и в подпрограммах в этой программе.
- Все строковые переменные, используемые в программе и в подпрограммах в этой программе.

CARD_VERIFY mode

FD_VERIFY mode

Назначение

Устанавливает ON/OFF функций контроля периферийных устройств, когда данные записываются на PC карту или дискету.

Параметр

Режим

0: Устанавливает OFF для функций контроля

1: Устанавливает ON для функций контроля

2: Устанавливает режим, выбранный последним

Если режим не задан, по умолчанию устанавливается 2.

RENAME new program name=existing program name

CARD_RENAME new program name=existing program name

FD_RENAME new program name=existing program name

Назначение

Изменяет имя программы, содержащейся в данный момент в памяти. RENAME изменяет имя программы, содержащейся в памяти контроллера, CARD_RENAME изменяет имя программы, хранящейся на PC карте, FD_RENAME на диске

Параметр

Параметр, стоящий слева в равенстве, задает новое имя программы.

Параметр, стоящий в равенстве справа, задает имя программы, которую надо переименовать.

Пояснение

Если имя новой программы уже существует, команда RENAME выдаст ошибку.

```
>FD_RENAME test=test.tmp
```

XFER destination program name, step number1=source program name, step number2, step count

Назначение

Копирует и переносит шаги из одной программы в другую программу.

Параметр

1. Имя программы адресата

Задаёт программу для получения копируемых данных. Если программное имя не существовало, данные переносятся в новую программу с этим именем.

2. Номер шага 1

Задаёт номер шага, до которого вставляются копируемые данные. Если шаг не задан, данные вставляются в конце заданной программы.

3. Имя программы источника

Задаёт имя программы, из которой копируются данные.

4. Номер шага 2.

Задаёт номер шага в программе источнике, откуда копируются данные. Если номер шага не задан, копирование начинается с начала программы.

5. Количество шагов

Задаёт количество шагов для копирования из программы источника, начиная с шага п.4. Если количество шагов не задано, копируются все шаги, следующие за номером шага 2.

ПРИМЕЧАНИЕ Если в программе адресате для осуществления изображения информации используются STATUS, PCSTATUS команды или команды редактора, XFER команда не может быть использована.

COPY new program name=source program name+source program name+....
CARD_COPY new program name=source program name+source program name+
FD_COPY new program name=source program name+source program name+.....

Назначение

Копирует существующую программу в новую программу. FD_COPY копирует программы на дискете. CARD_COPY копирует программы на PC диске.

Параметры

Новое программное имя

Задаёт имя программы, куда должна скопироваться существующая программа.

Имя программы источника

Задаёт имя программы, которая должна быть скопирована.

Пояснение

Когда заданы две или более программ источников, программы комбинируются в одной программе под новым именем. Имя, заданное для новой программы, не должно быть таким же, как имя уже существующей программы.

TRACE stepper number: O N/O F F

Назначение

Отслеживает и регистрирует реальное содержимое управляющих программ робота и PC программ, которые были выполнены.

Параметры

1. номер повторителя

Задаёт тип программы для отслеживания выбором номера:

1: программа робота

1001: PC program 1 1004: PC program 4

1002: PC program 2 1005: PC program5

1003: PC program 3

Если тип не задан, все программы регистрируются.

2. ON/OFF

Начинает/завершает отслеживание.

Пояснение

Если необходимая память не зарезервирована, используя SETTRACE команду до выполнения TRACE ON, происходит ошибка (P2034) “Memory undefined for logging” .

SETTRACE `step count`

Назначение

Резервирует необходимую память для записи отслеживаемых данных.

Параметр

Количество шагов

Задаёт число шагов для записи в установленных границах от 1 до 9999. Если параметр не задан, в памяти записывается до ста шагов.

Пояснение

Часть памяти резервируется, для того чтобы вместить информацию, заданную параметром и информацию о существующих AS программах и PC программах.

Если TRACE ON и LSTRACE выполняются без резервирования памяти для регистрирования, произойдет ошибка (P2034) «Memory undefined for logging» . Если SETTRACE команда используется во время регистрации, произойдет ошибка (P2033) «Logging is in process», и все отслеживания отключатся (завершатся)

RESTRACE

Назначение

Освобождает память, зарезервированную командой SETTRACE

Пояснение

Если RESTRACE команда используется во время регистрации, произойдет ошибка (P2033) «Logging is in process»

LSTRACE `stepper number: logging number`

Назначение

Отображает записанные данные о AS программах или PC программах

Параметры

Задаёт тип программы при помощи выбора номера

1: Robot program

1001: PC program 1

1004: PC program 4

1002: PC program 2

1005: PC program 5

1003: PC program 3

Если тип программы не задан, отображается запись о программах робота.

Регистрационный номер строки

Задаёт номер строки, с которой начинается вывод информации на дисплей.

Если номер не задан, выбирается первая строка.

Пояснение

Если необходимая память не была зарезервирована при помощи SETTRACE команды до выполнения LSTRACE команды, будет иметь место ошибочное состояние (ошибка P2034) «Memory undefined for logging».

Если LSTRACE команда задается во время регистрации, возникнет ошибочное состояние (P2033) «Logging is in process».

Когда LSTRACE команда выполнена, зарегистрированные данные появляются на дисплее. Быстрый просмотр всех данных, может быть осуществлен, используя следующие команды:

N (enter) – отображает следующие 9 строк

L (enter) – отображает предыдущие 9 строк

S number (enter) – отображает информацию, заданную регистрационным номером строки и 4 строки регистрируются до и после этой строки. Если номер строки не определен, отображается информация с 1 по 9 строки. Если номер больше чем существует строк, отображается наибольший номер.

F character (enter) – отображает строки, которые содержат в себе заданные символы и 4 строки регистрируются до и после этой строки. Если символ не задан, по умолчанию используется символ предыдущей команды F. Если символ не находится в данных, ничего не отображается.

E (enter) – заканчивает просмотр и возвращается в AS мониторный режим.

(enter) – отобразит только 9 строк.

91	pg1	31	JOINT SPEED9 ACCU1	TIMER0	TOOL1	WORK0	CLAMP1 (OFF,0,0,0)	2
92	pg1	32	SIGNAL	14:sig on				
93	pg1	33	JOINT SPEED9 ACCU1	TIMER0	TOOL1	WORK0	CLAMP1	
			(OFF,0,0,0)	2				
94	pg1	34	CALL "sub1"					
95	sub1	1	PRINT "SUB1"					
96	sub1	2	xyz:					
97	sub1	3	JMOVE a					
98	sub1	4	JMOVE b					
99	sub1	5	JMOVE c					

5.3. КОМАНДЫ СОХРАНЕНИЯ ПРОГРАММ И ДАННЫХ

CARD_FORMAT	форматирует PC карту.
FD_FORMAT	форматирует дискету.
SAVE/P *	сохраняет программы
SAVE/L *	сохраняет координаты.
SAVE/R *	сохраняет реальные переменные.
SAVE/S *	сохраняет символьные строки.
SAVE/A *	сохраняет вспомогательную информацию
SAVE/SYS *	сохраняет данные системы.
SAVE/ROB *	сохраняет данные робота.
SAVE/ELOG *	сохраняет данные журнала ошибок.
LOAD *	загружает программы и данные в память робота.

ПРИМЕЧАНИЕ* эти команды записывают данные в персональный компьютер. Для того чтобы сохранить данные на PC карте или дискете добавьте префикс CARD_ или FD_ в команду. Смотрите пояснения к каждой команде.

CARD_FORMAT

FD_FORMAT **format type**

Назначение

Форматирует PC карту или дискету

Параметры

Тип формата

Задается для форматирования дискеты

1: 1.44 MB

2: 1.25 MB

Если параметр не задан, по умолчанию 1.

Назначение

Эта команда перезаписывает данные формата в памяти и создает новую файловую директорию. Предыдущее содержимое стирается из памяти.

Замечание

Новые SRAMPC карты должны быть отформатированы для дальнейшей с ними работы. Перепрограммируемая ATA PC карта не форматируется. Обычно они продаются отформатированными.

Пример

```
>CARD_FORMAT ☐  
Are you sure? (Yes:1, NO:0)? 1☐  
Now formatting card...  
>  
Formatting done.
```

SAVE/SEL **file name=program name,.....**

CARD_SAVE/SEL **file name=program name,.....**

FD_SAVE/SEL **file name=program name,.....**

Назначение

Save команда записывает программы и различные данные на винчестер компьютера (используется только, когда персональный компьютер подключен к контроллеру робота)

CARD_SAVE команда записывает программы и другие данные на PC карту.

FD_SAVE команда записывает программы и другие данные на дискету.

Параметры

1.Имя файла

Записывает заданные программы в файл с заданным именем. Если расширение не задано, автоматически задается расширение “.as”

2. Программное имя

Выбирает программы для записи. Если программное имя не задано, записываются все программы, хранящиеся в памяти контроллера.

Пояснение


Команды SAVE/P, SAVE/L, SAVE/R, SAVE/S, SAVE/SYS записывает выбранный тип данных в отдельный файл. Команда SAVE все типы файлов в один файл.

Команда SAVE (без /SEL) записывает выбранные программы, включая любые переменные и подпрограммы, используемые этими программами.

Команда SAVE/SEL записывает только программы без подпрограмм и переменных, использующихся в этих программах.

Замечание

Если выбранное имя файла уже существует на программном носителе, а команда записи выполнена, ранее созданный файл переименовывается, в расширение добавляется буква “b”. Для примера, если файл “file1.as” уже существовал в памяти, и команда >SAVE file1 была выполнена, ранее созданный файл переименуется “file1.bas”, файл созданный позже будет “file1.as”.

```
>SAVE f3=cycle,motor 
```

Запишет под именем файла “f3.as” системные данные, две программы “cycle” и “motor”, подпрограммы, вызываемые из этой программы и переменные, используемые в программах.

SAVE/P/SEL file name=program name,....
SAVE/L/SEL file name=program name,....
SAVE/R/SEL file name=program name,....
SAVE/S/SEL file name=program name,....
SAVE/A file name
SAVE/SYS file name
SAVE/ROB file name
SAVE/ELOG file name
CARD_SAVE/P/SEL file name=program name,....
CARD_SAVE/L/SEL file name=program name,....
CARD_SAVE/R/SEL file name=program name,....
CARD_SAVE/S/SEL file name=program name,....
CARD_SAVE/A file name
CARD_SAVE/SYS file name
CARD_SAVE/ROB file name
CARD_SAVE/ELOG file name

FD_SAVE/P/SEL file name=program name,....
FD_SAVE/L/SEL file name=program name,....
FD_SAVE/R/SEL file name=program name,....
FD_SAVE/S/SEL file name=program name,....

FD_SAVE/A file name
FD_SAVE/SYS file name
FD_SAVE/ROB file name
FD_SAVE/ELOG file name

Назначение

Записывает на программный носитель внешней памяти следующие файлы

Файл, содержащий программы (/P)

Файл, содержащий информацию о координатах (/L)

Файл, содержащий информацию о реальных переменных (/R)

Файл, содержащий информацию о строковых переменных (/S)

Файл, содержащий информацию о вспомогательных данных (/A)

Файл, содержащий системные данные (/SYS)

Файл, содержащий данные о роботе (/ROB)

Файл, содержащий данные об ошибках (/ELOG)

Как с SAVE командой, CARD_SAVE/ и FD_SAVE/ команды используются для того чтобы записать файлы на PC карту и дискету соответственно (SAVE/ команду используют только тогда, когда персональный компьютер подключен к контроллеру. См. 2.6.2

Параметры

1. Имя файла

Записывает данные под определенным именем. Имена различаются расширением для каждого типа данных. Если расширение не задано, следующие расширения автоматически присваиваются файлам в соответствии с типом данных

Программы	.PG	Системные данные	.SYS
Координаты	.LC	Данные о роботе	.RB
Реальные переменные	.RV	Данные об ошибках	.EL
Строковые переменные	.ST		
Вспомогательная информация	.AU		

2. Имя программы

Выбирает имя программы, чтобы сохранить. Если не указано, все программы и данные, находящиеся в памяти будут сохранены в файле на диске.

Объяснение

1. SAVE/P

Запоминает в указанном файле на диске выбранную программу (ы) и подпрограммы, вызываемые этой программой (ми) (включая подпрограммы, вызываемые подпрограммами).

Имена программы (программ), которые были сохранены в файле, отображаются на системном терминале. Некоторые дополнительные имена программы, указанные командой SAVE могут появиться. Они являются именами подпрограмм, которые вызываются этой программой. Эти подпрограммы сохраняются в том же самом файле как программа.

Программы сохраняются в файле в алфавитном порядке независимо от порядка, в котором они были сохранены.

2. SAVE/L, SAVE/R, SAVE/S

Сохраняет только переменные, используемые в указанной программе (ах) и подпрограмме (ах), вызываемой этой программой (ми). (/L: сохраняет только координаты, /R: сохраняет только реальные переменные, /S: сохраняет только строковые переменные)

3. SAVE/A

Сохраняет вспомогательную информацию.

4. SAVE/SYS

Сохраняет системные данные.

5. SAVE/ROB

Сохраняет данные, принадлежащие определенной роботу.

6. SAVE/ELOG

Сохраняет файл регистрации ошибок. Эта команда не может быть введена вместе с другими командами сохранения.

Например, SAVE/ELOG/R не функционирует.

7. Если /SEL введен с/P,/L,/R,/S, только основная программа и переменные, используемые только в основной программе сохраняются. Подпрограммы и переменные, используемые в подпрограммах, не сохраняются.

Если указанное имя файла уже существует в памяти, то существующий файл автоматически переименован с “b” перед расширением файла. (См., что примечание относительно команды SAVE).

```
>SAVE/L file2=pg1.pg2 
```

LOAD/Q file name

CARD_LOAD/Q file name

FD_LOAD/Q file name

Назначение

LOAD команда загружает файлы из памяти компьютера в память контроллера. (Используется только, когда РС подключен к контроллеру).

CARD_LOAD команда загружает файлы с PC карты в память контроллера.

FD_LOAD команда загружает файлы с дискеты в память компьютера.

Параметры

Имя файла

Загружает файлы с заданным именем. Если расширение не задано, автоматически задается расширение “.as”

Пояснение

Эта команда загружает данные в память контроллера. Попытка загрузить программное имя такое, как существует в памяти контроллера, приведет к ошибке и выполнение команды загрузки прервется.

(ПРИМЕЧАНИЕ)

Когда загружаются координаты, реальные переменные, строковые переменные, имена которых уже существуют в памяти, данные в памяти перезаписываются без любых предупреждений. (программы не перезаписываются)


Первоначальные данные удаляются, если команда отменяется при перезаписи данных в памяти.

При задании LOAD команды с /Q до загрузки данных или программ появляется следующий диалог:

Load? (1: Yes, 0: No, 2: Load all, 3: Exit)

- 1: загрузить данные
- 0: не производить загрузку данных и перейти к следующим данным
- 2: загрузить все данные без опроса
- 3: не производить загрузку, закрыть команду

Если есть нечитаемые или некорректные шаги в программе, следующие сообщения появятся: “The step format is incorrect (0: Continue load 1: Delete program and exit)”. Если работа продолжается при помощи ввода 0, используйте редактор, для того чтобы откорректировать шаг, после чего программа может быть загружена.

```
>LOAD  pallet   
Loading...  
System data  
Program    a1()  
Program    test()  
:  
Transformation values  
Joint interpolation values  
Real values  
Loading done.  
>
```

5.4 КОМАНДЫ УПРАВЛЕНИЯ ПРОГРАММОЙ

SPEED	Устанавливает мониторную скорость.
PRIME	Подготавливает программу для выполнения.
EXECUTE	Выполняет программу
STEP	Выполняет один шаг программы.
MSTEP	Выполняет одну инструкцию движения.
ABORT	Останавливает выполнение после завершения выполнения текущего шага.
HOLD	Останавливает выполнение.
CONTINUE	Возобновляет выполнение программы.
STPNEXT	Выполняет программу по шагам.
KILL	Инициализирует стек выполнения.
DO	Выполняет одну программную инструкцию.

SPEED monitor speed

Назначение

Устанавливает мониторную скорость в процентах

Параметр

Мониторная скорость

Устанавливает скорость в процентах. Если эта величина 100, скорость будет установлена 100% от максимальной. Если 50, скорость будет установлена 50% от максимальной скорости.

Если будет установлена опция отмены ограничения скорости, то значение может быть установлено до 99999 (%)

Пояснение

Если задана мониторная скорость (при помощи этой команды) и программная скорость (установленная в программе при помощи программной инструкции SPEED) приоритетной является мониторная скорость. Для примера, если мониторная скорость 50, а программная 60 при выполнении программы скорость будет 30.

(ПРИМЕЧАНИЕ)

Максимальная скорость робота автоматически становится 100 %, если результат мониторной и программной скорости превышает 100 %.

По умолчанию установка мониторной скорости 10%.

Эта команда не будет затрагивать скорость движения, выполняемую в настоящее время. Новая установка скорости вступает в силу после того, как текущее движение и запланированное движение закончены.

>SPEED 30

>SPEED 50

>SPEED 100

>SPEED 200

PRIME program name, execution cycles, step number

Назначение

Подготавливает систему так, что программа может быть выполнена, используя переключатель. Эта команда, однако не запускает выполнение программы.

Параметры

1. Имя программы

Выбирает программу, которую нужно подготовить к выполнению. Если имя не задано, выбирается программа, выполненная последней, или подготовленная последней к выполнению.

2. Количество циклов для выполнения программы.

Устанавливает сколько раз выполняется программа. Если параметр не установлен, выполняется 1 раз. Непрерывное выполнение задается отрицательным числом -1.

3. Номер шага

Выбирает шаг, с которого начинается выполнение программы. Если параметр не задан, выполнение программы начинается с первого шага.

Объяснение

Эта команда только готовит систему к выполнению программы. Она не выполняет программу. Программа может быть выполнена, используя команду CONTINUE после команды PRIME, подготовившей систему. Программа может также быть выполнена, используя переключатель `CYCLE START`.

(ПРИМЕЧАНИЕ)

При использовании этой команды, стек выполнения в памяти робота инициализирован; то есть любая информация, указывающая программе приостановиться (например команда HOLD, или ошибка) будет потеряна. Например, если выполнение программы приостановлено, в то время, как выполняется подпрограмма (информация запоминается в стеке), и затем подпрограмма выполнена, используя эту команду с `CYCLE START` или CONTINUE команду (стек инициализирован), обработка не может возвратиться в главную программу, поскольку стек был инициализирован.

EXECUTION `program name, execution cycles, step number`

Назначение

Выполняет запуск программ.

Параметры

1. Имя программы

Задаёт программу для выполнения. Если параметр не задан, выполняется программа, заданная последней.

2. Количество циклов для выполнения программы.

Устанавливает количество циклов для выполнения программы. Если параметр не установлен, выполняется 1 раз. Непрерывное выполнение задается отрицательным числом -1.

3. Номер шага

Выбирает шаг, с которого начинается выполнение программы. Если параметр не задан, выполнение программы начинается с первого шага.

Пояснение

Выполняет заданную программу робота с заданного шага. Выполнение повторяется заданное количество раз.



CAUTION

Когда используется эта команда, следующие состояния устанавливаются автоматически

SPEED 100 ALWAYS
ACCURACY 1 ALWAYS

STOP инструкция или последний шаг программы заканчивает цикл.

>EXECUTE test,-1 

STEP program name, execution cycles, step number

MSTEP program name, execution cycles, step number

Назначение

Выполняет один шаг программ

Параметры

Имя программы

Задаёт программу для выполнения. Если параметр не задан, выполняется программа, заданная последней.

Количество циклов для выполнения программы.

Устанавливает количество циклов для выполнения программы. Если параметр не установлен, выполняется 1 раз. Непрерывное выполнение задается отрицательным числом -1.

Номер шага

Выбирает шаг, с которого начинается выполнение программы. Если параметр не задан, выполняется первый шаг программы.

Если параметры не заданы, выполняется шаг, следующий после последнего выполненного шага.

Пояснение

Эта команда может быть выполнена без параметров в следующих случаях:

1. после PAUSE инструкции
2. после программного останова в случаях, отличных от ошибочных состояний
3. когда предыдущая программная инструкция была выполнена при использовании STEP команды.

MSTEP команда выполняет только один сегмент движения (т.е. одну инструкцию движения и шаги до следующей инструкции движения). STEP команда выполняет только один шаг программы (робот необязательно должен двигаться).

ABORT

Назначение

Останавливает выполнение программы

Пояснение

Останавливает выполнение программы после завершения текущего шага.
Если робот двигается, выполнение останавливается после завершения движения.
Продолжить программное выполнение возможно, используя команду CONTINUE.

Замечание

В AS системе движение робота и выполняемый шаг могут быть не всегда совпадающими.
Т.е., если выполнение шагов происходит быстрее, чем выполнение движения, робот может совершить еще одно движение до останова.

HOLD

Назначение

Останавливает выполнение программы

Пояснение

Останавливает движение робота немедленно. В отличие от **EMERGENCY STOP** переключателя, силовое питание не выключается. Эта команда имеет такой же эффект, как переключатель **HOLD/RUN** в положении HOLD. Продолжить выполнение программы можно, используя команду CONTINUE.

CONTINUE **NEXT**

Назначение

Продолжает выполнение программы, прерванное PAUSE, ABORT, HOLD командами или ошибочным состоянием системы. Эта команда так же может быть использована для старта программ, готовность к выполнению которых задается командами PRIME, STEP, MSTEP.

Параметр

NEXT Следующий шаг

Если параметр не введен, выполнение продолжается с шага, в котором произошло прерывание, в противном случае со следующего шага.

Пояснение

Применение параметра NEXT для рестарта программ различаются в зависимости от способа останова выполнения программ.

1. Программа остановлена во время выполнения шага или движения
CONTINUE продолжает выполнение программы с прерванного шага, повторяя его.
CONTINUE NEXT продолжает выполнение программы со следующего шага

2. Программа остановлена после завершения выполнения шага. CONTINUE, CONTINUE NEXT продолжают выполнение с шага, следующего за шагом, выполненным ранее.

3. Программа приостановлена при помощи WAIT, SWAIT, TWAIT инструкций CONTINUE NEXT перепрыгивает (игнорирует) эти инструкции и продолжает выполнение программы со следующего шага.

(ПРИМЕЧАНИЕ)

Команда CONTINUE не может продолжить выполнение программы когда:

- программа закончилась должным образом
- программа была остановлена, используя команду HALT
- использовалась команда KILL

STPNEXT

Назначение

Выполняет следующий шаг, когда системный переключатель STP_ONCE в положении ON.

Пояснение

Когда системный переключатель STP_ONCE в положение ON программа может выполняться пошагово по возрастанию.

KILL

Назначение

Инициализирует стек для программ.

Пояснение

Если программа остановлена PAUSE инструкцией, ABORT командой или ошибкой, программный стек не изменяется. Команда KILL используется для инициализации стека. Если команда KILL выполнена, CONTINUE команда не эффективна, т.к. она не является программой стека.

DO program instruction

Назначение

Выполняет единичную программную инструкцию. (Некоторые программные инструкции не могут быть выполнены при помощи этой команды)


Параметр

Программная инструкция

Задаёт определённую программную инструкцию. Если параметр не задан, выполняется последняя, выполненная инструкция.

Пояснение

Программные инструкции обычно записываются в программах и выполняются как программные шаги. Однако команда DO разрешает выполнение единичной программной инструкции без создания программ для выполнения этой инструкции.

>DO JMOVE safe 

>DO HOME 

с

5.5 КОМАНДЫ ПОЗИЦИОННОЙ ИНФОРМАЦИИ

HERE	Определяет переменную позицию как текущую позицию.
POINT	Определяет переменную позиции.
POINT/X	Устанавливает X значение переменной позиции.
POINT/Y	Устанавливает Y значение переменной позиции.
POINT/Z	Устанавливает Z значение переменной позиции.
POINT/OAT	Устанавливает OAT значение переменной позиции.
POINT/O	Устанавливает O значение переменной позиции.
POINT/A	Устанавливает A значение переменной позиции.
POINT/T	Устанавливает T значение переменной позиции..
POINT/7	Устанавливает значение седьмой оси для переменной позиции

HERE pose variable name

Примечание

Выражение «координата точки» не означает, что точка имеет одну координату, точка всегда имеет столько координат, сколько степеней подвижности у робота.

Выражение «имя точки» означает, что у точки есть координаты и их конкретное название

Назначение

Задаёт имя переменной позиции для текущего положения (запоминает координату точки в пространстве).

Точка может быть декартовой, угловой, сложной.

Параметр

Присваивает заданной координате имя. Имя должно начинаться с буквы или префикса #.

Может быть декартовой, угловой, сложной.

Пояснение

Позиция может быть декартовой, угловой, сложной координатой.

Примеры

>HERE #pick запоминает координату точки в углах под именем pick

>HERE place запоминает координату точки в декартовых координатах
под именем place

>HERE a+b запоминает координату точки как сложную точку

POINT pose variable name=pose values, joint displacement values

Назначение

Присваивает координате, имеющей имя и стоящей слева в равенстве, значение координат точки, имеющей имя и стоящей в правой части равенства.

Параметры

1. Определяет имя координаты, которую необходимо задать (декартовую, угловую, сложную)

2. Задаёт имя реально существующей координаты. Если параметр не задан, на дисплее появляются координаты точки pose variable name с запросом на изменение “Change?”. При этом можно изменить координату точки, поставив в соответствующие координаты необходимые значения. Выполнение команды и её закрытие происходит путем нажатия клавиши (Enter).

3. Этот параметр должен быть установлен, когда в левой части равенства координата задается, как угловая, а первая координата правой части равенства, как декартовая. (если

параметр в левой части не задан как угловая координата, параметр не может быть установлен).

Угловая координата правой части равенства определяет конфигурацию робота в пространстве в новой позиции. Если параметр не задан, текущая конфигурация используется для задания переменной позиции.

Объяснение

Присваивает значения позиции, указанные справа в переменную позиции. Когда значения позиции не указаны, любые значения, уже определенные для этой позиции отображаются на терминале, и могут быть отредактированы. Если переменная позиции не определена, отображенные значения будут 0, 0, 0, 0, 0, 0.

Как только POINT выполняется, значения позиции отображаются в сопровождении соответствующего сообщения "Change?" и подсказкой. Значения могут тогда быть отредактированы. Для того чтобы выйти, нажмите только **ENTER** в подсказке.

Если переменная определена угловыми координатами, то значения суставов появляются на дисплее. Если переменная определена декартовыми координатами, значения XYZOAT отображены. XYZ значения описывают позицию начала инструментальной системы относительно основной системы координат. Значения OAT описывают ориентацию инструментальной системы координат. Когда переменная выражена в значениях координат сложной точки, задается крайне правая переменная сложной точки. Если другие переменные, используемые в составном значении не определены, эта команда завершается ошибкой.

(ПРИМЕЧАНИЕ)

Когда виды значений правой и левой части равенства различаются, эта команда работает следующим образом:

1. POINT transformation values=joint displacement values

Значение угловых координат правой части преобразуются в декартовы координаты и присваиваются в переменную левой части.

2. POINT joint displacement values=transformation values, joint displacement values

Значение декартовых координат правой части преобразуются в значения угловых координат и присваиваются в переменную в левой части равенства. Если значение угловых координат справа заданы, значение декартовых координат преобразуются, делая конфигурацию робота такой же, как заданные значения угловых координат. Если параметр не задан, значения декартовых координат преобразуются с текущей конфигурацией робота.

Когда задаются значения, максимум 9 десятичных цифр могут быть введены, но точность ввода с более чем 9 числами не может быть гарантирована.

Примеры

>POINT #park

```
JT1    JT2    JT3    JT4    JT5    JT6
10.000 15.000 20.000 0.000 30.000 90.000
Change? (if not, hit RETURN only) (Enter)
,,,35
```

JT1 JT2 JT3 JT4 JT5 JT6
10.000 15.000 20.000 35.000 30.000 90.000
Change? (if not, hit RETURN only) (Enter)

>POINT pick1=pick (Enter)

присваивает декартовой координате pick1 значение декартовой координаты pick

>POINT pos0=#pos0 (Enter)

преобразует угловую координату #pos0 в декартовую и присваивает декартовой координате pos0 данное значение

>POINT #pos1=pos1,#pos2 (Enter)

преобразует декартовую координату "pos1" в угловую, используя конфигурацию робота, определенную позицией #pos2, присваивает это значение точке "#pos1"

POINT/X transformation variable name=transformation values
POINT/Y transformation variable name=transformation values
POINT/Z transformation variable name=transformation values
POINT/OAT transformation variable name=transformation values
POINT/O transformation variable name=transformation values
POINT/A transformation variable name=transformation values
POINT/T transformation variable name=transformation values
POINT/7 transformation variable name=transformation values

Назначение

Присваивает элементы декартовой координаты, заданной в правой части равенства соответствующим элементам декартовой координаты, определенной в левой части равенства. Эти значения будут отображены на дисплее для редактирования.

Параметры

1. Определяет имя координаты, компонент которой необходимо задать (декартовую, сложную)

2. Задаёт имя реально существующей координаты. Если параметр не задан, знак = можно опустить

Пояснение

Присваивает определенные компоненты значения позиции (X,Y,Z,O,A,T), указанные справа в соответствующий компонент декартовой координаты слева. Как только POINT выполняется, значения позиции отобразятся в сопровождении соответствующего сообщения "Change?" и подсказкой. Значения могут тогда быть отредактированы. Для того чтобы выйти, нажмите только **ENTER** в подсказке.

>POINT/OAT a2 = a1

X[mm]	Y[mm]	Z[mm]	O[deg]	A[deg]	T[deg]
0.	0.	0.	10.	15.	30.

Change?(If not, hit RETURN only) ☐

5. 6. КОМАНДЫ УПРАВЛЕНИЯ СИСТЕМОЙ

STATUS	Отображает системное состояние.
WHERE	Отображает текущие данные позиции для робота.
IO	Отображает состояние двоичных сигналов.
FREE	Отображает объем свободной памяти.
TIME	Отображает и устанавливает текущее время и дату.
ULIMIT	Устанавливает верхний предел движения робота.
LLIMIT	Устанавливает нижний предел движения робота.
BASE	Изменяет основные значения преобразования.
TOOL	Определяет значения преобразования инструмента.
SETHOME	Устанавливает домашнюю позицию.
SET2HOME	Устанавливает домашнюю позицию номер 2.
ERRLOG	Отображает хронологию условий ошибки.
OPLOG	Отображает хронологию операций.
SWITCH	Отображает установку системных выключателей.
ON	Включает системный переключатель.
OFF	Отключает системный переключатель.
ZSIGSPEC ввода - вывода.	Устанавливает и отображает общее количество сигналов
HSETCLAMP	Устанавливает заданные по умолчанию спецификации фиксатора.
DEFSIG	Отображает или устанавливает программно установленные приоритетные сигналы.
ZZERO	Отображает или устанавливает данные обнуления.
ERESET	Сбрасывает состояние ошибки.
SYSINIT	Инициализирует всю систему.
HELP	Отображает распечатку команд/инструкций AS языка.
ID	Отображает информацию j версии программного

обеспечения.

WEIGHT	Устанавливает данные веса нагрузки .
BATCHK	Разрешает/запрещает проверку низкого напряжения батареи.
ENCCHK_ EMG	Устанавливает приемлемый диапазон отклонения при проверке позиции робота: поза при аварийном останове в сравнении с позицией, когда робот перезапускается.
ENCCHK_ PON	Устанавливает приемлемый диапазон для различия в значении кодера при включении силового питания в сравнении с значением, когда силовое питание было выключено в прошлый раз.
SLOW_ REPEAT	Устанавливает скорость режима замедленного повтора.
REC_ ACCEPT	Разрешает/запрещает запись и или изменение программ.
ENV_ DATA	Устанавливает автоматическое Отключение таймера серводвигателя и ПРУ подключение / отключение.
ENV_ 2DATA	Устанавливает терминал подключение/отключение.
CHSUM	Очищает ошибку контрольной суммы.
PLCAOUT	Устанавливает реальные значения на выходные данные. (Опция)
TPLIGHT	Включает подсветку пульта ручного управления . (Опция)
IPEAKLOG	Отображает достигнутый максимум текущих значений. (Опция)
IPEAKCLR	Сбрасывает максимумы текущих значений. (Опция)
OPEINFO	Отображает операционную информацию. (Опция)
OPEINFOCLR	Очищает операционную информация . (Опция)

STATUS

Назначение

Отображает состояние системы, дает информацию о выполняемой или выполненной программе.

1....	Robot status:				
	REPEAT mode:				
2....	Environment:				
	Monitor Speed(%) = 10.0				
	Program Speed(%)ALWAYS = 100.0				
	ALWAYS Accu.[mm]= 1.0				
3....	Stepper status: Program is not running.				
4....	Execution cycles				
	Completed cycles: 3				
	Remaining cycles: Infinite				
5....	Program name	Prio	Step number		
	test	0	1	WAIT	sig(1001)

1. Состояние робота

Текущее состояние робота - одно из следующих:

Состояние ошибки: ошибка произошла; пробуйте операцию сброса ошибки.

Силовое питание привода выключено: силовое питание привода OFF.

Режим обучения: силовое питание включено; робот управляется, используя пульт ручного управления.

Автоматический режим: Силовое питание включено; робот управляется программой робота.

Режим повтора начало цикла разрешен: силовое питание привода включено; программа робота выполняется.

Программное ожидание: силовое питание привода включено; программа робота выполняется, и в состоянии ожидания (выполнение WAIT, SWAIT, или команда TWAIT).

2. Среда

Текущая мониторная скорость (в процентах)

3. Состояние шагового выполнения

Текущее состояние выполнения шага.

4. Циклы выполнения

Законченные циклы: Выполнение циклов повторения уже закончено (от 0 до 32767)

Оставшиеся циклы: Оставшиеся циклы для выполнения. Если отрицательный номер был определен для выполнения цикла в команде EXECUTE, "бесконечный" отображается.

5. Имя программы

Имя программы или шаг, текущее выполняемый, или состояние ожидания.

WHERE display mode

Назначение

Отображает на дисплее положение робота, в данный момент времени.

Параметр

Выбирает режим, в котором отображается информация. Таких режимов 16, 7-16 опции. Т.е. WHERE ,...,WHERE 16. Если параметр не задан, текущее положение робота показано в угловых и декартовых координатах. Изображение режима не изменится до тех пор, пока не нажмете **ENTER**.

WHERE	отображает текущее положение робота в декартовых или угловых координатах (JT1, JT2, ..., JT3
WHERE 1	Отображает текущую позицию робота в углах.
WHERE 2	Отображает текущую позицию робота в XYZOAT в базовых координатах (mm, deg).
WHERE 3	Отображает текущее обученное значение (deg).
WHERE 4	Отображает смещение от обученного значения (bit).
WHERE 5	Отображает значение кодера каждого сустава (bit).
WHERE 6	отображает скорость каждого сустава (deg/s).
WHERE 7	Отображает текущую позицию, включая внешние оси. (Option)
WHERE 8	Отображает текущее положение фиксированных координат детали (Option)
WHERE 9	Отображает обученное значение каждого сустава для декартовой координаты.
WHERE 10	Отображает ток двигателя.
WHERE 11	Отображает скорость двигателя.
WHERE 12	Отображает значения текущих декартовых координат, выраженные в базовых координатах другого робота
WHERE 13	Отображает значения текущих декартовых координат, выраженные в инструментальных координатах другого робота
WHERE 14	Отображает обученное значение тока двигателя.
WHERE 15	Отображает первоначальные данные кодера.
WHERE 16	Отображает скорость TCP.

>WHERE 

JT1	JT2	JT3	JT4	JT5	JT6
9.999	0.000	0.000	0.000	0.000	0.000
X[mm]	Y[mm]	Z[mm]	O[deg]	A[deg]	T[deg]
15.627	88.633	930.000	-9.999	0.000	0.000

IO/E signal number

Назначение

Отображает текущее состояние всех внешних внутренних входных/выходных сигналов.

Параметр

Сигнальные номера

- 1.....отображает 1-32, 1001-1032, 2001-2032
- 2.....отображает 33-64, 1033-1064, 2033-2064
- 3.....отображает 65-96, 1065-1096, 2065-2096
- 4.....отображает 97-128, 1097-1128, 2097-2128

По умолчанию отображается 1

Пояснение

Если системный переключатель DISPIO_01 в положении OFF, «о» будет отображать сигналы в положении ON, «х» сигналы в положении OFF.

Приоритетные сигналы отображаются заглавными буквами («О» «X»). Если системный переключатель DISPIO_01 в положении ON, «1» сигнал включен, «0» сигнал выключен.

Если «/E» введен вместе с командой, сигнальные номера 3001 и далее отображаются вместе с сигналами, пронумерованными 1-, 1001-, 2001-.

На дисплее постоянно обновляется информация о состоянии сигналов.

Прервать вывод информации можно клавишей (Enter)

>IO/E 

32 -	1	xxxx	xxxx	xxxx	xxXX	xxxx	XXXX	XXXO	XXXO
1032 -	1001	xxxx	xxxx	xxxx	xxXX	xxxx	XXXX	XXXO	XXXO
2032 -	2001	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
3032 -	3001	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx

>

>IO 

32 -	1	0000	0000	0000	0000	0000	0000	0001	0001
1032 -	1001	0000	0000	0000	0000	0000	0000	0000	1000
2032 -	2001	0000	0000	0000	0000	0000	0000	0000	0000

>

FREE

Назначение

Дает информацию о наличии свободной памяти контроллера в процентах и битах.

>FREE 

Total memory 262144 bytes

Available memory size 262128 bytes (99%)

TIME year – month – day hour:minute:second

Назначение

Устанавливает и отображает текущую дату и время

Параметр

Год – месяц – дата часы: минуты: секунды

Устанавливаются в формате, описанном ниже. Когда устанавливается параметр часы: минуты: секунды параметр год – месяц – дата может быть пропущен.

Пояснение

Эта команда устанавливает календарное и суточное время на ПРУ робота. Границы значений для каждого параметра следующие:

Год (00 – 99)

Месяц (01 – 12)


Дата (01 – 31)

Час (0 – 23)

Минута (0 – 59)

Секунда (0 – 59)

Текущее время или введенные значения отображаются сопровождаемыми соответствующим сообщением “Change? ” Чтобы изменить данные, введите новые значения. Нажмите клавишу **ENTER**, чтобы закончить команду.

>TIME 02-04-29 09:45:46 

>TIME

Current time 02-04-29 09:47:33

Change? (If not , hit RETURN only)

02-05-17

Current time 02-05-17 09:47:33

Change? (If not , hit RETURN only)

> 

ULIMIT joint displacement values
LLIMIT joint displacement values

Назначение

Устанавливает и отображает верхний/нижний пределы границ движения робота.

Параметр

Имя точки в угловых координатах

Устанавливает программно новые пределы (верхний/нижний) границ движения робота в угловых координатах.

Пояснение

Если параметр задан, на дисплее появляются угловые координаты точки, которые можно изменить после вопроса “Change?”, на необходимые пользователю. Введите требуемые значения, как делается в команде POINT. Для того чтобы выполнить команду нажмите **ENTER** клавишу.

Если параметр не задан, отображается текущее положение робота, которое можно изменить или не менять в зависимости от ответа на вопрос “Change?”.

```
>ULIMIT                                     Displays the current setting.
      JT1   JT2   JT3   JT4   JT5   JT6
Maximum 120.00 60.00 60.00 190.00 115.00 270.00 (
Current  30.00 15.00 25.00 -40.00  60.00  15.00 (
Change? (If not , hit RETURN only)
>110,50
      JT1   JT2   JT3   JT4   JT5   JT6
Maximum 120.00 60.00 60.00 190.00 115.00 270.00
Current 110.00 50.00 25.00 -40.00  60.00  15.00
Change? (If not , hit RETURN only) ↵
```

```
>ULIMIT #upper ↵
```

```
:
```

```
>LLIMIT #low ↵
```

BASE transformation values

Назначение

Задаёт сдвиг начала координат (0,0,0,0,0,0) декартовой системы координат на определенные величины (X,Y,Z,O,A,T), заданные в параметре, создавая тем самым новую относительную систему координат.

Параметр

Имя точки (точка декартовая, сложная)

Определяет новую базовую систему координат. Точка здесь описывает положение начала координат новой системы по отношению к началу координат нулевой базовой системы координат. Если параметр не определен, на дисплее появляется информация об установленной системе координат в данный момент времени.

Пояснение

Команда BASE используется, как правило, когда есть неоднократно повторяющиеся траектории движения. Задается траектория движения в нулевой системе координат, затем, задав нужную базовую систему, без обучения точек, произойдет повторение траектории движения.

Следует учитывать, что координата точки (параметр) не может быть создана, путем обучения и запоминая, используя команду HERE. Координата точки создается аналитически (вычисляется) и вводится в память компьютера при помощи команды POINT.

Если "NULL" определяется для параметра, основные значения преобразования установлены как “нулевая база” (XYZOAT=0, 0, 0, 0, 0, 0,). Когда система инициализирована, основные значения преобразования автоматически установлены как нулевая база.

После того, как новое значение преобразования базы установлено, отображается сообщение “Change? ”. Чтобы изменить значения, введите новые значения, отделенные запятыми и нажмите **ENTER**. Если никакой параметр не определен, отображаются текущие значения.

Когда робот перемещается в позицию, определенную декартовыми координатами базы или вручную, используя базовый режим, система автоматически вычисляет позицию робота, принимая во внимание декартовы координаты базы, определенные здесь.

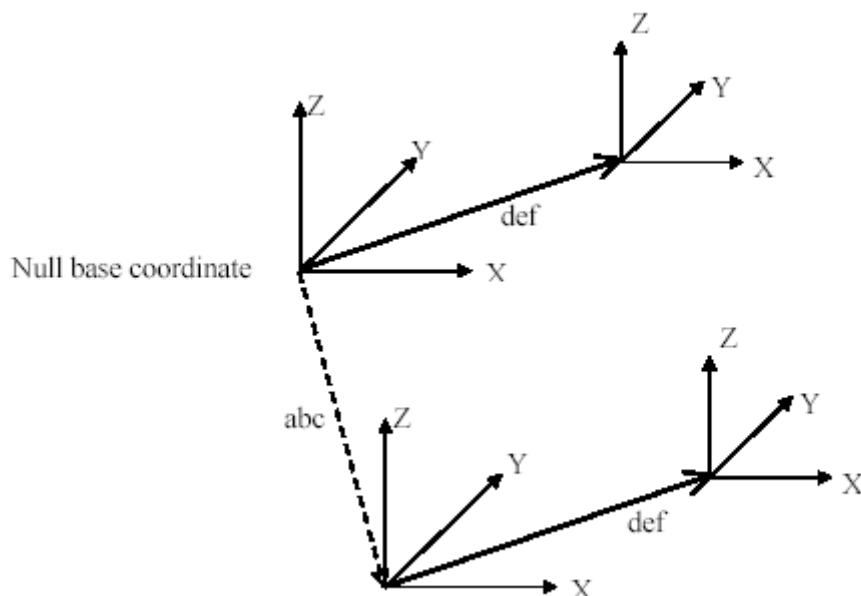
Когда переменная позиции (координата) используется, как параметр и если эта переменная позиции переопределена, обратите внимание, что следует заново переопределить позицию, используя команду BASE, чтобы изменение вступило в силу. BASE команда не влияет позиции, заданные в угловых координатах.

[NOTE]

```
BASE abc   
DO JMOVE def 
```

```
BASE NULL   
DO JMOVE def 
```

Даже если координата “def” является одной и той же в обоих примерах, робот выйдет в различные точки назначения. Смотри рисунок ниже.



>BASE Displays the current base transformation values.

X[mm]	Y[mm]	Z[mm]	O[deg]	A[deg]	T[deg]
-300.	0.	0.	0.	0.	0.

Change? (If not , hit RETURN only)

>BASE NULL Changes the base transformation value to the null base.
(X, Y, Z, O, A, T)=(0, 0, 0, 0, 0, 0)

>BASE abc

TOOL transformation values

Назначение

Определяет новую инструментальную систему координат, которая задается как сдвиг инструментальной системы координат относительно нулевой инструментальной системы на определенные расстояния, выраженные в координатах X,Y,Z,O,A,T

Параметр

Имя точки (точка декартова, сложная)

Определяет новую инструментальную систему координат. Параметр здесь описывает положение начала координат новой системы по отношению к началу координат нулевой инструментальной системы координат. Если параметр не определен, на дисплее появляется информация об установленной инструментальной системе координат в данный момент времени.

Пояснение

Инструментальная система координат с началом 0,0,0,0,0,0 находится в центре фланца робота (6-ой сустав), плоскость XOY совпадает с плоскостью фланца, ось Z параллельна оси фланца.

Если установлена нулевая инструментальная система координат, вращение вокруг осей оставляет неподвижной точку начала инструментальной системы координат, но при наличии инструмента (все технологические операции требуют наличие какого-нибудь инструмента), конец инструмента перемещается в пространстве. При обучении и дальнейшем прохождении сложных траекторий, особенно, где требуется изменение ориентации на небольших расстояниях по ходу движения, могут возникнуть осложнения такие, как неравномерное движение от точки к точке, невозможность пройти по прямой траектории, во время сложного перехода возможно столкновение с обрабатываемой поверхностью. Чтобы избежать данных ситуаций используется команда TOOL. Начало координат переносится на конец инструмента и при вращении неподвижной остается рабочая часть инструмента.

Если "NULL" определяется для параметра, основные значения преобразования установлены как “нуль инструмента” (XYZOAT=0, 0, 0, 0, 0, 0,). Нулевые координаты инструмента имеют свое начало координат в центре монтажного фланца для инструмента и оси параллельны осям последнего сустава робота. Когда система инициализирована, основные значения преобразования автоматически установлены как нулевая база.

После того, как новое значение преобразования для инструмента установлено, отображается сообщение “Change? ”. Чтобы изменить значения, введите новые значения, отделенные запятыми и нажмите **ENTER**. Если никакой параметр не определен, отображаются текущие значения.

Когда робот перемещается в позицию, определенную декартовыми координатами базы или ручную, используя базовый режим или инструментальный режим, система автоматически вычисляет позицию робота, принимая во внимание декартовы координаты инструмента, определенные здесь.

Когда переменная позиции (координата) используется, как параметр и если эта переменная позиции переопределена, обратите внимание, что следует заново переопределить позицию, используя команду TOOL, чтобы изменение вступило в силу. TOOL команда не влияет на позиции, заданные в угловых координатах.

```
>TOOL grip 
```

```
>TOOL NULL 
```

OPLOG

Назначение

Отображает журнал операций

Пояснение

Отображает последние сто операций в формате, показанном ниже. При заполнении экрана просмотр продолжается нажатием клавиши пробел.

(Вспомогательная функция 0703)

Пример

>OPLOG (Enter)

1 – [02/07/17 10:04:46](SIGNAL:00) [PNL]

SWITCH switch name,...,switch name=ON
SWITCH switch name,...,switch name=OFF

Назначение

Отображает и изменяет системные переключатели и их установки.

Параметры

1. Имя переключателя

Отображает состояние выбранного системного переключателя. Если параметр не задан, отображает все системные переключатели и их установки.

При вводе более одного переключателя, имена разделяются запятой.

2. ON/OFF (включен/выключен)

Задаёт включение/выключение выбранного системного переключателя. Если параметр не задан, отображается последняя установка переключателя.

```
>SWITCH                               Displays all system switches and their setting.
*POWER      ON      *REPEAT          ON
*RUN         ON      *CS              OFF
*RGSO        OFF     *ERROR           OFF
*TRIGGER     ON      *TEACH_LOCK     OFF
CHECK.HOLD   OFF     CP              ON
CYCLE.STOP   OFF     OX.PREOUT       ON
PREFETCH.SIGINS OFF   QTOOL          OFF
REP_ONCE     OFF     RPS             OFF
STP_ONCE     OFF     AFTER.WAIT.TMR   ON
MESSAGES     ON      SCREEN          ON
AUTOSTART.PC OFF     AUTOSTART2.PC    OFF
AUTOSTART3.PCOFF   ERRSTART.PC OFF
DISPIO_01     OFF    HOLD.STEP        OFF
FLOWRATE      OFF    SPOT_OP          OFF
>[F4]
```

Switch name,.... ON
Switch name,....OFF

Назначение

Задавая имя системного переключателя, устанавливается состояние включения/выключения в соответствии с ON/OFF

Параметр

Имя переключателя

Включает/выключает заданный переключатель. При задании более одного имени переключателя, они разделяются запятой.

Текущая установка системного переключателя может быть закрыта при помощи SWITCH команды.

```
>MESSAGES ON [F4]
```

```
>SCREEN, MESSAGES ON [F4]
```

```
>MESSAGES OFF [F4]
```

```
>SCREEN, MESSAGES OFF [F4]
```

ZSIGSPEC

Назначение

Отображает и изменяет общее число внешних и внутренних I/O сигналов.

Пояснение

Отображается текущая установка и вопрос “Change?”.

Изменение происходит при ответе на вопрос “Change?”. Эта команда изменяет только сигналы, соответствующие данным аппаратным средствам, данному программному обеспечению.

>ZSIGSPEC ☐

DO, DI, INT (DO=Ext. output signal, DI= Ext. input signal, INT=Int. signal)

64 64 128

Change? (If not , hit RETURN only)

32,32,32

DO, DI, INT

32 32 32

Change? (If not , hit RETURN only) ☐

HSETCLAMP

Назначение

Присваивает номера сигналов, для того чтобы управлять фиксаторами для погрузочно-разгрузочных операций.

Пример

В примере ниже, фиксатор 3 устанавливается как двойной соленоид.

>HSETCLAMP ☐

	Clamp 1	Clamp 2	Clamp 3	Clamp 4
	Spot weld	Handling	Not used	Not used
'ON'out.signal	10	0	24	24
'OFF'out.signal	9	11	0	0
	Clamp 5	Clamp 6	Clamp7	Clamp 8
	Not used	Not used	Not used	Not used
'ON'out.signal	24	24	24	24
'OFF'out.signal	0	0	0	0

Clamp number (1~8, ENTER only:No change, CTRL+C:Exit) 3

;Select clamp number

Номер фиксатора

выбор номера фиксатора

Как задать фиксатор схвата?

(1:Defined as Handling clamp, 0:Not used, ENTER only:No change, CTRL+C:Exit)1

; введите 1, для того чтобы задать фиксатор схвата.

Для одного электромагнитного клапана, задается один сигнал.
Для двойного электромагнитного клапана, задаются два сигнала.

'ON' out. signal

(0:Not used, ENTER only:No change, CTRL+C:Exit) Change ? 12 ; Устанавливает канал 12 в состояние ON

'OFF' out. signal

(0:Not used, ENTER only:No change, CTRL+C:Exit) Change ?; 13

Устанавливает канал 13 в состояние OFF

	Clamp 1	Clamp 2	Clamp 3	Clamp 4
	Spot weld	Handling	Handling	Not used
'ON'out.signal	10	0	12	24
'OFF'out.signal	9	11	13	0
	Clamp 5	Clamp 6	Clamp7	Clamp 8
	Not used	Not used	Not used	Not used
'ON'out.signal	24	24	24	24
'OFF'out.signal	0	0	0	0

Clamp number (1~8, ENTER only:No change, CTRL+C:Exit) 3

(ПРИМЕЧАНИЕ)

Всегда используйте фиксаторы по порядку от 1 до 8. Для примера, нельзя использовать фиксатор 5 без использования фиксатора 4.

(ПРИМЕЧАНИЕ)

Ctrl + C (Exit) не может быть использована с экрана клавиатуры пульта ручного управления.

DEFSIG INPUT DEFSIG OUTPUT

Назначение

Отображает, изменяет состояние программных, приоритетных сигналов.

Параметр

INPUT, OUTPUT

OUTPUT (или 0) отображает состояние выходных сигналов, INPUT (или 1) входных сигналов. Эти состояния могут быть изменены, когда параметр внесен. Если параметр не внесен, сигналы, используемые в настоящее время как приоритетные, отображаются в листинге.

Пояснение

Сигналы в таблице ниже могут быть использованы как приоритетные сигналы.

Software Dedicated Input Signal	Software Dedicated Output Signal
EXT. MOTOR ON	MOTOR_ON
EXT. ERROR RESET	ERROR
EXT. CYCLE START	AUTOMATIC
EXT. PROGRAM RESET	CYCLE START
Ext. prog. select (JUMP_ON, JUMP_OFF RPS_ON, RPSxx)	TEACH MODE HOME1, HOME2
EXT_IT	POWER ON
EXT. SLOW REPEAT MODE	RGSO
	Ext. prog. select (JMP_ST, RPS_ST)

Следующие коды могут использоваться с этими командами

L: перейти к предыдущему шагу

N: перейти к следующему шагу

Q: отменить последнюю операцию

E: выход из команды

Примечание

1. Внешняя программная выборка

Когда JMP сигнал устанавливается как приоритетный, сигналы JMP-ON, JMP-OFF, JMP-ST автоматически становятся приоритетными. JMP-ST является выходным сигналом, но устанавливается он под контролем DEFSIG INPUT команды

Когда RPS сигнал устанавливается как приоритетный, сигналы RPS-ON, RPS-OFF, RPS-ST автоматически становятся приоритетными. RPS-ST является выходным сигналом, но устанавливается он под контролем DEFSIG INPUT команды

2. RPS программа

Если не менее чем один из следующих сигналов установлен как приоритетный, появляется подсказка и RPS программа должна быть введена.

3. Сигнальные номера

Сигналы могут быть установлены внутри следующих границ

Приоритетные выходные сигналы: 1 – число установленных сигналов

Приоритетные входные сигналы: 1001 – число установленных сигналов

4. Другое

Если номер сигнала присвоено приоритетному сигналу, оно не может быть присвоено другому приоритетному сигналу или использовано как общий сигнал.

Пример

Следующий пример показывает программно установленные приоритетные сигналы

```
>DEFSIG
Dedicated signals are set

EXT. MOTOR ON = 1032
EXT. ERROR RESET = 1031
EXT. CYCLE START = 1030
MOTOR ON = 32
ERROR = 31
AUTOMATIC = 30
    Condition : Panel switch in RUN.
    Condition : Panel switch in REPEAT.
    Condition : Repeat continuous.
    Condition : Step continuous.
CYCLE START = 29

TEACH MODE = 28
HOME1 = 27
> ☐
```

Следующий пример сбрасывает выбор программно установленного, приоритетного выходного сигнала MOTOR_ON, изменяет сигнальный номер AUTOMATIC в 30, выбирает TEACH MODE как приоритетный сигнал и устанавливает номер сигнала 3.

```
>DEFSIG OUTPUT
MOTOR ON    Dedication cancel? (Enter 1 to cancel.) 1 ☐
ERROR       Dedication cancel? (Enter 1 to cancel.) ☐
    Signal number    31    Change ? ( 1 - 32 ) ☐
AUTOMATIC    Dedication cancel? (Enter 1 to cancel.) ☐
    Signal number    2    Change ? ( 1 - 32 ) 30 ☐
CYCLE START    Dedication cancel? (Enter 1 to cancel.) ☐
    Signal number    29    Change ? ( 1 - 32 ) ☐
TEACH MODE    Dedication set? (Enter 1 to set.) 1 ☐
    Signal number    0    Change ? ( 1 - 32 ) 3 ☐
HOME1        Dedication cancel? (Enter 1 to cancel.) ☐
> ☐
```

ZZERO joint number

Назначение

Устанавливает значение кодера в соответствии с известной механической позиции робота. Также, количество смещения между механической позицией робота и началом координат кодера (нулевые данные) могут быть отображены, используя эту команду.

Параметр

Номер сустава

(1) Сбрасывает счетчик вращения кодера:

Введите номер сустава плюс 100. Например, чтобы сбросить счетчик вращения кодера для второго сустава введите:

ZZERO 102 ↵

Если “100” введен как номер сустава, все суставы сбрасываются в 0 °.

(2) Для того чтобы установить нулевые данные:

Чтобы установить нулевые данные кодера для сустава два, введите:

ZZERO 2 ↵

Если “0” введен как номер сустава, все суставы устанавливаются в 0 ° как нулевые данные.

Если никакой номер сустава не задан, отображаются текущие данные кодера, и нулевые данные.

(ПРИМЕЧАНИЕ)

Сбросьте счетчик вращений кодера до установки нулевых данных.



Используйте эту команду только для следующих целей:

- 1. Для того чтобы проверить, изменились ли нулевые данные, когда позиция манипулятора не соответствует заданной позиции.**
- 2. Для того чтобы исправить обнуление данных, если они неожиданно изменились.**

Когда нулевые данные изменены, значения для позиций робота также изменяются. Поэтому знайте, что та же самая программа будет заканчивать движение в различных координатных точках и траектория до и после обнуления данных изменится.

Пример

1. Следующая команда отображает нулевые данные:

```

>ZZERO 
      JT1      JT2      JT3      JT4      JT5      JT6
Set data 268435456 268435456 268435456 268435456 268435456 268435456
Current 268435456 268435456 268435456 268435456 268435456 268435456
data
Change? (If not , hit RETURN only) 

      JT1      JT2      JT3      JT4      JT5      JT6
OFFSET   0       0       0       0       0       0
Change? (If not , hit RETURN only) 
>

```

2. Следующая команда сбрасывает счетчик вращения кодера для всех суставов.

```

>ZZERO 100
** Encoder rot. counter reset (all joints) **
Are you sure? (Enter 1 to execute) 1 
Setting complete.
>

```

3. Следующая команда сбрасывает счетчик вращения кодера для 2 сустава для выбранной позиции сустава.

```

>ZZERO 102
** Encoder rot. counter reset (joint 2) **
Current angle (deg, mm) ? 0 
Are you sure? (Enter 1 to execute) 1 
Setting complete.
>

```

4. Следующая команда устанавливает нулевые данные так, что текущая позиция является 0°.

```

>ZZERO 0 
      JT1      JT2      JT3      JT4      JT5      JT6
Set data 268427264 268427264 268427264 268427264 268427264 268427264
Current 268427264 268427264 268427264 268427264 268427264 268427264
Set current values of all joints as zeroing data? (Enter 1 to set.)1 
Setting complete. 

```

5. Устанавливает, чтобы робот признал значение кодер, как задание текущего положения 2 сустава 0°

```
>ZZERO 2   
Current angle (deg.mm)? 0   
Change? (If not, Press RETURN only.)   
Encoder value? (Current=268435456, Enter 1to set current value) 1   
Zeroing value=268435456 (268419072-268451840) OK? (Enter 0 to change)   
Setting complete.   
>
```

ERESET

Назначение

Сброс ошибочного состояния, аналогично кнопке **ERROR RESET** на операционной панели

Объяснение

Когда команда ERESET выполнена, выводится сигнал ERROR_RESET. Однако это команда неэффективна, когда ошибка происходит непрерывно.

SYSINIT

Функция

Удаляет все программы и данные в памяти и инициализирует определенные параметры.

Объяснение

Инициализирует систему и удаляет все программы, координаты, числовые переменные, и строковые переменные.

(ПРИМЕЧАНИЕ)

Все программы и переменные удаляются из памяти, когда эта команда выполняется.

HELP	alpha character
HELP/M	alpha character
HELP/P	alpha character
HELP/F	alpha character
HELP/PPC	alpha character
HELP/MC	alpha character
HELP/DO	alpha character
HELP/SW	alpha character

Назначение

Распечатывает листинги команд и инструкций AS языка.

Параметр

Задаёт с какой буквы алфавита начинается команда, инструкция и т.д. Если параметр пропущен, листинг распечатывается полностью

Для примера, ввод алфавитного символа с командой HELP отобразит мониторную команду или программную инструкцию, начинающуюся с этой буквы. Ввод HELP/F команды с алфавитным символом отобразит функцию, начинающуюся с этой буквы.

/M – листинг мониторных команд

/P – листинг программных инструкций

/F – листинг функций

/PPC – листинг программных инструкций, используемых в PC программах

/MC – листинг мониторных команд, используемых с MC инструкциями

/DO – листинг программных инструкций, используемых с DO командой

/SW – листинг системных переключателей

Для некоторых команд и инструкций будут отображаться и параметры

>HELP/M 

ABORT	BASE	BITS	BATCHK	CONTINUE	COPY	DEFSIG
DELETE	DIRECTORY	DLYSIG	DO	EDIT	ERESET	ERRLOG

>HELP/F 

#DEST	#PPOINT	\$CHR	\$DECODE	\$ENCODE	\$ERROR
\$LEFT	\$MID	\$RIGHT	\$SPACE	ABS	ASC

ID

Назначение

Выводит на дисплей информацию о версии программного обеспечения, встроенную в данный контроллер.

Пояснение

Отображает следующую информацию.

Имя робота: имя робота, подключенного в настоящее время

Количество суставов: количество суставов робота

Серийный номер: серийный номер робота

Версия программного обеспечения: номер версии AS программного обеспечения

Серво: номер версии программного обеспечения сервоуправления

Количество сигналов: общее количество входных, выходных, внутренних сигналов доступных в этой системе

Количество фиксаторов: общее количество фиксаторов, доступных в этой системе

Тип движения: тип движения робота

Тип сервоуправления: тип программного обеспечения сервоуправления

(ПРИМЕЧАНИЕ)

Если вышеупомянутая информация не соответствует фактическому роботу, войдите в контакт с нами немедленно. Не включайте силовое питание привода, не задавайте роботу выполнение каких-либо операций движения.

>ID 

Robot name : JS005-E001 Num of axes: 6 Serial No.1

Software version : version 000004-04...97/01/27 13:11

Servo : SAOA00-UX120-01

Number of signals: output=32 input=32 internal=256

Clamp number :2 MOTION TYPE: 1 SERVO TYPE:1

WEIGHT load mass , center of gravity location X, center of gravity location Y, center of gravity location Z, inertia moment ab. X axis, inertia moment ab. Y axis, inertia moment ab. Z axis

Назначение

Устанавливает данные массы нагрузки (вес инструмента и детали). Данные используются, чтобы определить оптимум для ускорения робота.

Параметр

1. Масса нагрузки

Масса инструмента и детали (в килограммах). Диапазон: 0.0 до максимально возможной нагрузки..

2. Расположение центра силы тяжести (мм)

X значение x центра силы тяжести в инструментальной системе координат

Y значение y центра силы тяжести в инструментальной системе координат

Z значение z центра силы тяжести в инструментальной системе координат

3. Момент инерции относительно Оси X, момент инерции относительно оси Y, момент инерции относительно Оси Z (Опция)

Устанавливает момент инерции вокруг каждой из осей. (kg·m²)

Момент инерции вокруг каждой оси определяется как момент вокруг осей координат, параллельных нулевой инструментальной системе координат с центром вращения в центре силы тяжести инструмента.

Объяснение

Если никакие параметры не определены, отображается текущее значение, сопровождаемое соответствующим сообщением “Change? ”



DANGER

Всегда устанавливайте правильную массу нагрузки и расположение центра тяжести. Неправильные данные могут ослабить или сократить долговечность деталей или вызвать ошибки по перегрузке / отклонениям.

BATCHK

Назначение

Разрешает/запрещает контролировать поддерживающую батарею.

Пояснение

>batchk

BATTERY ERROR CHECK(0:Ineffect, 1:Effect)

(Enter only: No change ^C:Exit): Now 1 Change ?

Если не надо делать изменения, нажмите ENTER. Введите 0 для того чтобы запретить проверку и 1 для того чтобы разрешить ее.

(ПРИМЕЧАНИЕ)

Ctrl + C (Exit) не может быть использована с экрана клавиатуры пульта ручного управления.

ENCCHK_ EMG

Назначение

Устанавливает приемлемый диапазон отклонения при проверке позиции робота при аварийном останове в сравнении с позицией, когда робот перезапущен.

Объяснение

Deviation = |(pose after motor power is reapplied) – (pose after the emergency stop)|

Отклонение= ((положение после включения силового питания привода) – (позиция после аварийного останова))

Приемлемый диапазон отклонения может быть установлен для каждого сустава. Если 0.0 установлен как диапазон, проверка отклонения не выполняется. Установка слишком маленького диапазона может вызвать ошибку, когда силовое питание привода повторно включается после аварийного останова, даже если робот работает в пределах рабочих спецификаций.

ENCCHK_PON

Назначение

Устанавливает приемлемый диапазон для разницы значениями кодера при включении силового питания привода в сравнении со значением, когда силовое питание привода было выключено последний раз.

Пояснение

Acceptable range = | (value when control power is turn ON) – (value when the control power was turned OFF the last time)|

Приемлемый диапазон может быть установлен для каждого сустава. Установка слишком маленького диапазона может вызвать ошибку при включении силового питания после аварийного останова, даже если робот работает внутри спецификаций технических характеристик.

SLOW_REPEAT

Назначение

Установка автоматической скорости в режиме замедленного повтора программ.

```
>SLOW REPEAT
SLOW REPEAT MODE Speed (1~25%)
(Enter only: No change ^C:Exit): Now 10 Change ?
```

Если режим не изменяется, нажмите только **ENTER**. Для того чтобы изменить скорость, введите новое значение и нажмите **ENTER**.

(ПРИМЕЧАНИЕ)

Ctrl + C (Exit) не может быть использовано с экрана клавиатуры на пульте ручного управления.

REC_ACCEPT

Назначение

Разрешает или запрещает запись или программное изменение функций

0 – разрешает

1 – запрещает

```
>REC ACCEPT ☐  
RECORD(0:Enable, 1:Disable)  
      (Enter only: No change ^C:Exit): Now    0 Change ?  
PROGRAM CHANGE(0:Enable, 1:Disable)  
      (Enter only: No change ^C:Exit): Now    0 Change ?
```

Введите 0 для того чтобы разрешить RECORD или PROGRAM CHANGE опцию. Введите 1 для того чтобы запретить опции.

(ПРИМЕЧАНИЕ)

1. Ctrl + C (Exit) не может быть использовано с экрана клавиатуры на пульте ручного управления.
2. Если PROGRAM CHANGE запрещено, появляется следующее сообщение, когда выполняется команда EDIT: “Program change inhibited. Set ACCEPT and operate again.” REC_ACCEPT команда не может использоваться в режиме редактирования.

ENV_DATA

Назначение

Устанавливает данные о внешних условиях для периферии. (авто выключение таймера сервоуправления или состояние установки пульта ручного управления)

```
>ENV_DATA (Enter)  
AUTO SERVO OFF TIMER (0:Servo not off)  
      (Enter only: No change ^C:exit): Now    0 Change?
```

Если изменять не надо, нажать (Enter). Ввод 0, запрещает авто выключение таймера сервоуправления. Разрешая таймер, вводим столько времени (в секундах), сколько сервоуправление запрещено.

Далее следует подсказка для установки ПРУ

```
TEACH PENDANT (0:Connect, 1:Disconnect)  
      (Enter only: No change ^C:Exit): Now    0 Change?
```

Если изменять не надо, нажать (Enter). 1 – работа робота без подключения пульта ручного управления. После отключения пульта ручного управления, установите на разъем заглушку.

(ПРИМЕЧАНИЕ)

. Ctrl + C (Exit) не может быть использовано с экрана клавиатуры на пульте ручного управления.

ENV2 DATA

Назначение

Устанавливает данные о внешних условиях для программного обеспечения.

Если изменять не надо, нажать (Enter).

Далее следует подсказка для установки терминала

>ENV2 DATA

TEACH PENDANT (0:Connect, 1:Disconnect)

(Enter only: No change ^C:Exit):Now 0 Change?

TERMINAL (0: Connect, 1:Disconnect)

(Enter only: No change ^C:Exit):Now 0 Change?

Обычно персональный компьютер используется как терминал.

(ПРИМЕЧАНИЕ)

Ctrl + C (Exit) не может быть использовано с экрана клавиатуры на пульте ручного управления.

CHSUM

Назначение

Разрешает или запрещает сброс аварийной ошибки контрольной суммы

>SHSUM

CLEAR CHECK SUM ERROR (0:Ineffect, 1:Effect)

(Enter only: No change ^C:Exit):Now 0 Change?

Если введен 0, ошибка не может быть сброшена. Если введена 1, ошибка сбрасывается. По умолчанию устанавливается «0». CHSUM сбрасывает в «0», когда силовое питание выключено.

Следующие сообщения появляются, если ошибка не может быть сброшена:

>CHSUM

Не может очистить контрольную сумму ошибок. Проверьте следующие команды или вспомогательные функции

ZZERO

DEFSIG

.

.

>

Если любые данные до сих пор содержат проверку аварийной суммы, сообщение первого примера не появляется (CLEAR CHECK SUM ERROR). Вместо него отобразится сообщение, как во втором примере, идентифицируя дополнительную неисправность.

(ПРИМЕЧАНИЕ)

Ctrl + C (Exit) не может быть использовано с экрана клавиатуры на пульте ручного управления.

PLCAOUT data number = real value

Опция

Назначение

Устанавливает значение данного реального числа в данные определенного числа.

Параметр

1. Данные числа

Определяет количество выходных данных в целом числе. Приемлемые числа из диапазона 1 - 32.

2. Реальное значение

Задаёт значение реального числа, которое должно быть установлено в выходные данные, вводится в десятичном выражении. Оно также может быть установлено в именах переменных. Приемлемые числа из диапазона 0 - 65535.

(ПРИМЕЧАНИЕ)

Эта команда действительна только, когда “Built-in Sequencer Function” опция в состоянии ON. Если опция не включена, появляется следующее сообщение об ошибке (E1102) Cannot execute, no option set up. – Check option specs.

Пример

>PLCAOUT 13=120 ↵ Sets “120 (decimal notation)” to data number 13.

TPLIGHT

Опция

Назначение

Включает подсветку пульта ручного управления.

Пояснение

Если подсветка экрана пульта ручного управления отключена, тогда эта команда включает подсветку. Если команда выполняется, когда подсветка включена, подсветка остается включенной 600 секунд.

IPEAKLOG

Опция

Назначение

Отображает пиковые значения тока для каждого сустава.

Пояснение

Отображает программное имя, номер шага, действующее значение переменного тока, и отношение пикового значения в механическом пределе или пределе двигателя, когда крутящий момент двигателя наивысший для каждого сустава.

Пример

>IPEAKLOG
Log since 00/1/24 13:44:23

Joint	Program	Step	Effective current		Date	
JT1	pg223	5	4.1[Arms]	29.5[%]	00/1/24	13:44
JT2	pg223	13	1.4[Arms]	9.8[%]	00/1/24	13:44
JT3	pg223	10	13.7[Arms]	98.2[%]	00/1/24	13:44
JT4	pg223	1	2.1[Arms]	55.5[%]	00/1/24	13:45
JT5	pg223	7	2.4[Arms]	64.8[%]	00/1/24	13:45
JT6	pg223	4	1.0[Arms]	27.8[%]	00/1/24	13:45

IPEAKCLR

Опция

Назначение

Очищает журнал пиковых значений и возобновляет регистрирование значений.

Пояснение

Зарегистрированные значения сбрасываются, используя команду IPEAKCLR, команду SYSINI, или при помощи инициализации системы, устанавливая в состояние ON dip-переключатель № 8 на 1КА плате.

Пример

>IPEAKCLR

Are you sure? (Yes:1,No;0) 1 ☐

>

OPEINFO robot number: joint number

Опция

Назначение

Отображает рабочую информацию.

Параметр

1. Номер робота

Определяет один или более роботов управляются одним контроллером.

2. Номер сустава

Задаёт для какого сустава должна отображаться информация. Если параметр не задан, отображается информация для всех суставов.

Пример

>OPEINFO 

Operation Info. (02/1/14 9:54:1 -) (FILE LOAD 02/1/14)

Control ON 0.2 [H]

Servo ON 0.1 [H]

Motor ON 4 times

Servo ON 10 times

Emergency stop (while in motion) 2 times

JT1

Operation hour 0.1 [H]

Operation distance 301.28 [x100 deg, mm]

JT2

Operation hour 0.1 [H]

Operation distance 193.84 [x100 deg, mm]

(ПРИМЕЧАНИЕ)

Пределы для накопления данных

1. Часы[H] : 69 years
2. Количество операций [раз]: приблизительно 2000 миллионов раз (1176 лет, если выполняет 10000 раз в день)
3. Пройденное расстояние [градусы, мм]: 12.5 лет, если работать со скоростью 500mm/s

OPEINFOCLR

Опция

Назначение

Сбрасывает рабочую информацию

5.7. КОМАНДЫ ДВОИЧНЫХ СИГНАЛОВ

Контроллер робота D серии использует два типа двоичных сигналов: внешние вход - выход сигналы между роботом и внешними устройствами, и внутренние сигналы входа - выхода, используемые внутри робота. Внутренние сигналы входа - выхода используются между роботом и программами РС, или как испытательные сигналы к тестовым программам прежде фактического подключения внешних устройств. Двоичные сигналы управляются или определяются при помощи использования следующих команд.

RESET	Сбрасывает все внешние I/O сигналы.
SIGNAL	Включает (или отключает) выходные сигналы.
PULSE	Включает выходной сигнал на заданное количество времени.
DLYSIG	Включает выходной сигнал, после прохождения заданного количества времени.
BITS	Устанавливает группу сигналов равную заданному значению.
SCNT	Выводит сигнал счетчика после прочтения заданного значения (опция)
SCNRESET	Очищает номер сигнала счетчика. (Опция)
SFLK	Устанавливает ON/OFF сигнала мерцания. (Опция)
SFLP	Устанавливает ON/OFF сигналов с SET/RESET сигналов (опция)
SOUT	Выводит сигнал, когда встречается определенное состояние. (опция)
STIM	Включает сигнал таймера, когда заданный сигнал становится ON (опция)
SETPICK	Устанавливает время для начала управления закрытием схвата. (опция)
SETPLACE	Устанавливает время для начала управления открытием схвата. (опция)

RESET

Назначение

Устанавливает выключение (OFF) всех внешних выходных сигналов. Приоритетные сигналы, сигналы для зажима, разжима и парадоксальные сигналы multifunction OX/WX не затрагиваются этой командой.

При использовании дополнительной установки, сигналы, используемые для экрана Панели Интерфейса не затрагиваются этой командой. (Опция)

(ПРИМЕЧАНИЕ)

Будьте внимательны при использовании этой команды, т.к. она выключает все другие сигналы, кроме упомянутых выше, даже в автоматическом режиме.

SIGNAL signal number,

Назначение

Включает (ON), выключает (OFF) заданные внешние или внутренние сигналы.

Параметр


Сигнальное число

Задаёт номер внешнего выходного сигнала или внутреннего сигнала

Сигнальное число определено, если сигнал задан как внешний выходной сигнал числами 1-32 или как внутренний сигнал 2001-2256, внешние входные сигналы не задаются.

Если сигнальное число положительное, сигнал устанавливается в ON, если отрицательное – в OFF. Если задан «0» никакие выходные сигналы не изменяются. Будет ошибкой задавать номер сигнала, когда выбранный сигнальный номер уже был присвоен приоритетному сигналу.

>SIGNAL -1,4,2010  Выключает 1 выход, включает 4 и 2010

>SIGNAL -reset,4  Если значение reset положительное, выключает сигнал, если отрицательное – включает, 4 включает

PULSE signal number, time

Назначение

Устанавливает состояние ON для выбранного сигнала на заданный промежуток времени

Параметры

1. Сигнальный номер

Задаёт внешний выходной сигнал или внутренний сигнал (только положительное число). Будет ошибкой задавать номер сигнала, когда выбранный сигнальный номер уже был присвоен приоритетному сигналу.

Сигнальное число определено, если сигнал задан как внешний выходной сигнал числами 1-32 или как внутренний сигнал 2001-2256, внешние входные сигналы не задаются.

2.Время

Устанавливает временной промежуток, в течение которого сигнал включен.

Если не задан, время автоматически задается 0.2 сек.

DLYSIG signal number, time

Назначение

Выводит выбранный сигнал по истечению заданного промежутка времени

Параметр

1. Сигнальный номер

Задаёт внешнего выходного сигнала или внутреннего сигнала. Будет ошибкой задавать номер сигнала, когда выбранный сигнальный номер уже был присвоен приоритетному сигналу. Если число положительное сигнал устанавливается ON, если отрицательное – OFF.

Приемлемые номера сигналов

Внешний выходной сигнал	1 – реальное количество сигналов или 64 (наименьшее 2)
-------------------------	---

Внутренний сигнал	2001–2256
-------------------	-----------

2. Время

Задаёт время в секундах для задержки выхода сигнала.

BIT starting signal number, number of signal=value

Назначение

Указывает группу внешних выходных сигналов или внутренних сигналов в двоичном исчислении. Состояние сигналов устанавливается ON/OFF в соответствии с бинарным эквивалентом заданного десятичного значения.

Параметр

1.Начальный сигнальный номер

Задаёт первый сигнал для установки состояния ON/OFF сигнальной группы.

2.Количество сигналов

Задаёт сигнальную группу

3.Десятичное значение

Устанавливает десятичное значение, в соответствии с которым в заданной группе сигналов устанавливается состояние ON/OFF сигналов. Десятичное значение преобразуется в двоичное значение, каждый бит двоичной величины задаёт сигнальное

состояние (включен/выключен), начиная с самого младшего двоичного разряда. Если бинарное выражение этого значения имеет больше битов чем число сигналов, устанавливается состояние только данного числа сигналов (начиная с начального сигнального номера), остальные биты игнорируются.

Если этот параметр опущен, отображается текущее сигнальное состояние

Будет ошибкой задавать номер сигнала, когда выбранный сигнальный номер уже был присвоен приоритетному сигналу.

Пояснение


Устанавливает (сбрасывает) сигнальное состояние одного или более выходных или внутренних сигналов в соответствии с заданным значением

Сигнальное число определено, если сигнал задан как внешний выходной сигнал числами 1-32 или как внутренний сигнал 2001-2256, внешние входные сигналы не задаются.

Задание сигнального номера большим, чем количество смонтированных сигналов приведет к ошибке.

>BITS 2001,3 

Отображает значение внутренних сигналов 2001, 2002, 2003

>BITS 1,8=100 

Внешние выходные сигналы 1-8 устанавливаются соответственно в состояние 01100100

SCNT counter signal number=count up signal, count down signal, counter clear signal, counter value

Назначение

Выводит сигнал счетчика, когда указанное значение счетчика достигнуто.

Параметры

Назначение

Выводит сигнал счетчика, когда заданное значение регистра достигнуто.

Параметры

1.Номер сигнала счетчика

Задается номером сигнала для вывода. Допустимые значения 3097-3128.

2.Сигнал для счета в прямом направлении

Задается сигнальным номером или логическим выражением. Всякий раз, когда сигнал изменяется от OFF к ON, счетчик подсчета в прямом направлении увеличивается на единицу.

3.Сигнал обратного счета

Задается сигнальным номером или логическим выражением. Всякий раз, когда сигнал изменяется от OFF к ON, счетчик подсчета в обратном направлении уменьшается на единицу.

4. Счетчик сигнала отбоя

Задается сигнальным номером или логическим выражением. Если этот сигнал становится ON, внутренний счетчик устанавливается в ноль.

5. Значение счетчика

Когда внутренний счетчик достигает этого значения, заданный сигнал выводится. Если дан «0» сигнал выключается.

Пояснение

Если сигнал счета в прямом направлении изменяется от OFF в ON, когда выполняется команда SCNT, тогда значение внутреннего счетчика увеличивается на единицу 1. Если сигнал обратного счета изменяется от OFF в ON, когда выполняется команда SCNT, тогда значение внутреннего счетчика уменьшается на единицу 1. Когда значение внутреннего счетчика достигнет значения, заданного в параметре (значение счетчика), выводится сигнал счетчика. Если выводится сигнал очистки счетчика, значение внутреннего счетчика становится 0. Каждый сигнал счетчика имеет свое индивидуальное значение счетчика. Для того чтобы принудительно сбросить внутренний счетчик в 0, используйте команду SCNTRESET.

Для того чтобы проверить состояние сигналов 3001 - 3128, используйте команду IO/E .
(опция)

SCNRESET counter signal number

Назначение

Возвращает в ноль значение внутреннего счетчика в соответствии с сигнальным номером счетчика.

Параметр

Номер сигнала счетчика

Задаёт номер сигнала счетчика для сброса. Установочный диапазон для номеров сигнала счетчика: 3097 - 3128.

SFLK signal number=time

Назначение

Устанавливает ON/OFF (мерцание) заданного сигнала в заданном цикле времени.

Параметры

1. Номер сигнала

Задаёт номер сигнала. Допустимые значения 3065-3096.

2. Время

Задаёт время цикла ON/OFF (реальное значение). Если значение отрицательное, мерцание прекращается.

Пояснение

Процесс ON/ OFF рассматривается в одном цикле, и цикл выполняется в определенное время.

SFLP output signal=set signal expression, reset signal expression

Назначение

Устанавливает ON/OFF выходного сигнала, используя установленный сигнал и сигнал сброса.

Параметры

1.Выходной сигнал

Задаёт номер выходного сигнала. Положительное число включает сигнал, отрицательное число выключает сигнал. Могут быть заданы только номера выходных сигналов.

2.выражение сигнала установки

Задаётся сигнальным номером или логическим выражением для установки выходного сигнала.

3. Выражение сигнала сброса

Задаётся сигнальным номером или логическим выражением для переустановки выходного сигнала.

Пояснение

Если установленный сигнал включен, выходной сигнал включается. Если переустановленный сигнал включен, выходной сигнал выключается. Выходной сигнал становится ON или OFF, когда выполняется команда SFLP, и нет, когда сигнал установки или сигнал сброса в ON.

SOUT signal number=signal expression

Назначение

Выводит заданный сигнал, когда устанавливается определенное состояние.

Параметры

1.Номер сигнала

Задаёт номер выходного сигнала

2.Сигнальное выражение

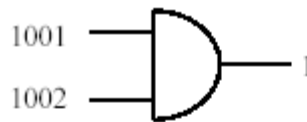
Задаёт номер сигнала или логического выражения.

Пояснение

Это команда – для логического вычисления сигналов. Используются логические выражения такие, как AND или OR. Определенный сигнал выводится, когда это состояние установится.

Example

SOUT 1 = 1001 AND 1002



SOUT 1 = 1001 OR 1002

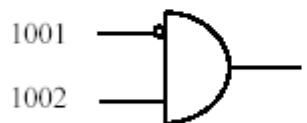


SOUT -1 = 1001 AND 1002

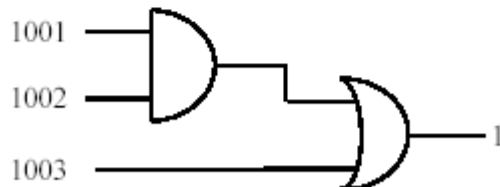
SOUT 1 = NOT(1001 AND 1002)



SOUT 1 = -1001 AND 1002



SOUT 1 = (1001 AND 1002) OR 1003



SOUT -1 = 1001 or SOUT 1 = -1001 or
SOUT 1 = NOT(1001)



STIM timer signal=input signal number, time

Назначение

Включает сигнал таймера, если указанный входной сигнал включен в течение заданного времени.

Параметры

1. Сигнал таймера

Выбирает сигнал для включения. Допустимые сигнальные номера 3001-3064.

2. Номер входного сигнала

Определяет в целых числах номер входного сигнала или логического выражения, чтобы контролировать условие для включения сигнала таймера. Значение не может превысить номера сигналов фактически установленных.

3. Время

Задаёт в действительных числах время (сек), в течение которого входной сигнал должен быть включен до включения сигнала таймера.

Пояснение

Контролируемый входной сигнал должен поступать непрерывно для того, чтобы сигнал таймера включился. Если входной сигнал пропадает до истечения заданного промежутка времени, происходит рестарт счета времени при появлении входного сигнала. Если входной сигнал не появился в данном промежутке времени, сигнал таймера не включается. Однако входной сигнал оказывает влияние на сигнал таймера, только в момент выполнения STIM инструкции.

Чтобы проверить состояние сигналов 3001-3128 используйте команду IO/E.

SETPICK time1, time2, time3, time4, ..., time8

SETPLACE time1, time2, time3, time4, ..., time8

Назначение

Устанавливает время для начала управления закрытием схвата (SETPICK) или открытием схвата (SETPLACE) для каждого из 8 фиксаторов.

Параметр

Время 1 - 8

Устанавливает управление временем для открытия/закрытия фиксаторов 1 - 8 в секундах.

Пояснение

См. CLAMP инструкцию для подробностей.

5.8. КОМАНДЫ ОТОБРАЖЕНИЯ ВЫРАЖЕНИЙ

PRINT	Отображает данные.
TYPE	Отображает данные.
IFPWPRINT	Отображает заданную символьную строку в окне отображения.

PRINT	device number: print data,....
TYPE	device number: print data,....

Назначение

Отображает на терминале данные печати, заданные в параметре.

Параметры

1.Номер внешнего устройства

Выбирает номер внешнего устройства для отображения данных

1: персональный компьютер

2: пульт ручного управления

Если параметр не задан, данные отображаются на подключенном, в данный момент, устройстве.

2.Данные печати

Устанавливает данные для печати. Разделите данные запятой, когда задаются несколько параметров.

(1) цепочка символов: т.е. "count="

(2) реальное выражение значения: т.е. count

(3) информация о формате: т.е. /D, /S

Если параметры не заданы, отображается пустая строка.

Пояснение

Если "2" введено для номера устройства, экран пульта ручного управления автоматически переходит в экран клавиатуры. Нажмите **NEXT PAGE** для того чтобы вернуться в обычный экран.

Следующие коды используются для задания выводного формата числового выражения. Пока не задан другой формат, используется одинаковый формат. В любом формате, если значение является слишком большим для изображения, звездочки заполняют все пространство. В этом случае необходимо ввести число символов, которые могут быть отображены. Максимальное число символов, помещающихся в одной строке 128. Отобразить более чем 128 символов в одной строке возможно, используя /S код.

Примечание

Если MEASSAGES переключатель выключен, сообщение не появляется.

Формат используемых кодов

/D

Использует заданный по умолчанию формат. Это тоже самое как формат /G15.8 за исключением того, что нули, следующие после числовых значений и пробелов, удалены.

/Em.n

Отображает числовое значение в экспоненциальном виде (в виде мантиссы и порядка, т.е. -1.234+02). "m" описывает общее число символов, отображаемых на экране, "n" – число десятичных разрядов. "m" должно быть больше чем "n", больше 6 и меньше 32.

/Fm.n

Отображает числовое значение с фиксированной запятой (-1.234). “m” описывает общее число символов, “n” – число знаков в дробной части.

/Gm.n

Если значение больше чем 0.01 и может быть отображено в Fm.n формате в пределах m цифр, значение отображается в том формате. В других случаях – в Em.n формате.

/Hn

Отображает значение как шестнадцатеричное число в n цифровом поле

/In

Отображает значение как десятичное число в n цифровом поле

Следующие параметры используются, чтобы вставить определенные символы внутри цепочки символов.

/Cn Вставка перевода строки n раз в месте, где этот код введен, либо спереди, либо после данных печати. Если этот код расположен внутри данных печати, n пустых строк вставятся.

/S Строка не продолжается

/Xn Вставка n пробелов


/Jn Отображает значение как шестнадцатеричное число в n цифровом поле. Нули отображаются на месте пробелов (опция)

/Kn Отображает значение как десятичное число в n цифровом поле. Нули отображаются на месте пробелов (опция)

/L То же самое как /D за исключением того, что все пробелы удаляются с этим кодом (опция)

Пример

В этом примере значение реальной переменной “i” есть 5, пятый элемент переменной массива “point” есть 12.66666.

```
>PRINT "point", i, "=", /F5.2, point [i] 
```


```
point 5 = 12.67
```

The display should look like this.

```
point 5 = *****
```

If the value of point[5] is 1000, the display should look like this. The value is too large to display (i.e. the number of digits is greater than 5).

В следующем примере код /S используется для того чтобы отобразить данные без изменения линий после данных

```
>PRINT "ABC"  
>PRINT/S, "DEF"  
>PRINT "GHI" 
```

На экране появится выражение

```
| ABC  
| DEFGHI      |
```

IFPWPRINT window, row, column,background color, label color= "character string",...

Назначение

Отображает заданную символьную цепочку в строке окна, установленного при помощи дополнительной функции 0509

Параметры

1.Окно

Выбирает окно, в котором отобразится последовательность символов. Выборка от 1 до 4 в дополнительной функции 0509.

2.Строка

Определяется строка в окне для отображения цепочки. Допустимое число от 1 до 16, однако, это зависит от размеров окна. Если не задано, подразумевается 1.

3.Колонка (столбец)

Определяется столбец в окне для отображения цепочки. Допустимое число от 1 до 43, однако, это зависит от размеров окна. Если не задано, подразумевается 1.

4.Цвет заднего фона

Определяется цвет заднего фона выбранного окна. Допустимое число от 1 до 15. Если не задано, задний фон белый.

5.Цвет символа

Определяется цвет символов. Допустимое число от 1 до 15. Если не задано, цвет черный.

6.Символьная строка

Задаёт цепочку символов для отображения. Все строки после первой строки отображаются в следующей строке, начиная с заданного столбца. Выполнение данной команды очищает все параграфы, за исключением заданной символьной цепочки, из заданного окна.

Пояснение

IFPWPRINT команда может использоваться только, когда панель интерфейса доступна для использования. Если параметры не определены, выбирается последняя установка

этого специфического окна (в течение первого использования, вышеупомянутые значения установлены по умолчанию). Если символьная строка не помещается в одной строке, ее переполнение отображается в следующей строке (выравнивая в выбранном столбце). Строки, которые превышают размеры окна, не отображаются. Управляющие символы в строке отображены как пробелы.

6.0 ПРОГРАММНЫЕ ИНСТРУКЦИИ

6.0 Программные инструкции

6.1 Инструкции движения

6.2 Инструкции управления скоростью и точностью

6.3 Инструкции управления фиксаторами

6.4 Инструкции конфигурации

6.5 Инструкции управления

6.6 Инструкции программных структур

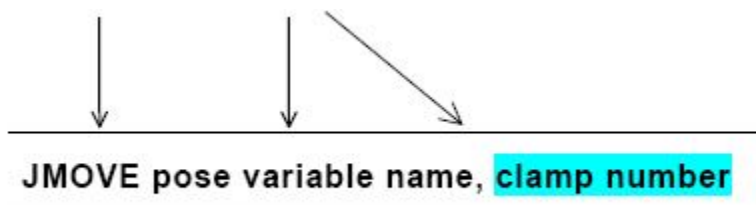
6.7 Инструкции бинарных сигналов

6.8 Инструкции управления сообщениями


6.9 Инструкции позиционной информации

6.10 Инструкции управления программой и данными

Ключевое слово **параметры**



Параметр, выделенный цветом, может быть опущен. Всегда вводите пробел между ключевым словом и параметрами

 в примерах представляет клавишу **Enter**.

6.1.ИНСТРУКЦИИ ДВИЖЕНИЯ

JMOVE	Перемещает робот в интерполированном угловом движении.
LMOVE	Перемещает робот в интерполированном прямолинейном движении
DELAY	Останавливает движение робота на определенное время.
STABLE совмещения осей.	Останавливает движение робота на определенное время после
JAPPRO движении.	Приближается к координате в интерполированном угловом
LAPPRO движении.	Приближается к координате в интерполированном прямолинейном
JDEPART движении.	Отходит от текущей позиции в интерполированном угловом
LDEPART движении.	Отходит от текущей позиции в интерполированном прямолинейном
HOME	Перемещается в позицию home.
DRIVE	Перемещается в направлении одной оси
DRAW	Перемещается на заданное расстояние в направлении X, Y, Z, осей базовой системы координат.
TDRAW	Перемещается на заданное расстояние в направлении X, Y, Z, осей инструментальной системы координат.
ALIGN	Располагает параллельно ось Z инструментальной системы координат с ближайшей осью базовой системы координат.
HMOVE	Перемещает робот в интерполированном прямолинейном движении (суставы запястья двигаются в интерполированном угловом движении)
XMOVE	Перемещается по прямолинейной траектории в заданную позицию.
C1MOVE	Перемещение с круговой интерполяцией
C2MOVE	Перемещение с круговой интерполяцией

JMOVE pose variable name, clamp number
LMOVE pose variable name, clamp number

Назначение

Перемещение робота в заданную позицию, с интерполированным угловым или прямолинейным движением

Параметры

1. Имя координаты (может быть задана в декартовых координатах, в угловых координатах, в сложных точках или функциями позиций)

2. Номер зажима

Определяет номер зажима для открытия или закрытия в точке назначения. Положительное число закрывает зажим, отрицательное – открывает. Любой номер фиксатора может быть установлен до максимума установкой номера через команду HSETCLAMP (или вспомогательную функцию 0605). Если опущено, фиксатор не открывается или не закрывается.

Пояснение

Робот перемещается по интерполированной криволинейной траектории, когда выполняется команда JMOVE. Робот перемещается так, чтобы соотношения между пройденным расстоянием и полным расстоянием, были одинаковы для всех суставов через все движение от начала позиции до конца позиции.

Робот перемещается в прямолинейном интерполированном движении, когда выполняется команда LMOVE. Начало инструментальной системы координат (TCP) движется по прямолинейной траектории.

JMOVE #pick

LMOVE ref+place

LMOVE #pick,1

DELAY time

Назначение

Задерживает движение робота на заданное время. Понимается как задание нулевой скорости.

Параметр

Время

Задаёт время в секундах.

Пояснение

В AS системе, DELAY инструкция понимается как инструкция движения, т.е. “движение в никуда”.

Даже если робот останавливается при помощи данной инструкции, выполнение программных шагов продолжается до следующей инструкции движения.

·
DELAY 2.5

STABLE time

Назначение

Задерживает выполнение следующих инструкций движения до истечения промежутка времени после совмещения осей. (Ожидает до тех пор, пока робот полностью не остановится)

Если для совмещения осей не хватает времени заданного в этой инструкции, робот останавливается и время вычисляется, когда оси совместятся снова.

Параметр

Задаёт время в секундах для останова движения.

Пояснение

Если совмещение осей не произошло, когда робот был остановлен этой инструкцией, время вычисляется, когда оси совместятся снова.

JAPPRO pose variable name, distance

LAPPRO pose variable name, distance

Назначение

Двигается по оси Z инструментальной системы координат, не доходя определенного расстояния до обученной точки. Движение происходит JAPPRO по криволинейной траектории, LAPPRO по прямолинейной траектории.

Параметр

1.Имя координаты

Задаёт точку назначения в декартовых или угловых значениях.

2.Расстояние

Задаёт расстояние между обученной точкой и положением в пространстве, которое достигает робот, двигаясь по оси Z инструментальной системы координат и не доходя до обученной точки на заданное расстояние. Если расстояние задано положительным числом, робот перемещается в отрицательном направлении оси Z, если отрицательным – перемещается в положительном направлении оси Z.

Пояснение

В этих командах, положение инструмента устанавливается в положении указанной позиции, и позиция установлена на указанном расстоянии от заданной позиции в направлении оси Z инструментальной системы координат.

JAPPRO place, 100

Перемещает робот в точку, находящуюся над точкой place, заданную как декартова координата, на расстоянии 100 мм, по направлению оси Z инструментальной системы координат

LAPPRO place, offset

JDEPART distance LDEPART distance

Назначение

Отходит от данной точки по оси Z инструментальной системы координат на определенное расстояние. Движение может быть криволинейным или прямолинейным

Параметр

Задаёт расстояние в миллиметрах между текущей позицией и позицией, расположенной на оси Z инструментальной системы координат, куда робот должен выйти. Если заданное расстояние положительное значение, робот движется по отрицательному направлению оси Z. Если заданное расстояние отрицательное значение, робот движется по положительному направлению оси Z.

JDEPART 80

LDEPART 2*offset

HOME home position number

Назначение

Двигается по криволинейной траектории в точку, имя координаты которой названы как HOME или HOME2.

Параметр

Номер положения HOME

Задаётся числами 1 или 2. Если параметр не задан, выбирается HOME 1.

Пояснение

Могут быть заданы две домашних позиции (позиции безопасности) (HOME1 и HOME2). Эта команда перемещает робот в одну из домашних позиций в интерполированном угловом движении. Домашняя позиция должна быть определена заранее, используя команды SETHOME или SET2HOME. Если домашняя позиция - не определена, нулевое начало координат (все суставы в 0 °) принято как домашняя позиция.

HOME

HOME 2

DRIVE joint number, displacement, speed

Назначение

Движение определенного сустава робота.

Параметры

1.Номер сустава

Задаёт номер сустава, для выполнения движения, в зависимости от конфигурации робота.

2.Смещение

Задаёт угол поворота для выполнения движения заданного сустава в положительных или отрицательных значениях. Единицы измерения для этой величины такие же, как и описывающие положение сустава. Если необходимо совершить вращательное движение, значение задается в градусах, если поступательное – в миллиметрах.

3.Скорость

Задаёт скорость перемещения. Как при задании обычной программной скорости, она задается в процентах от мониторной скорости. Если параметр не задан, устанавливается 100% мониторная скорость.

Пояснение

Эта инструкция двигает только один заданный сустав. Скорость движения для этой инструкции есть комбинация скорости, заданной в этой инструкции и мониторной скорости. Программная скорость, установленная в программе не эффективна для этой инструкции.

DRIVE 2,-10,75

Поворачивает 2 сустав на -10 градусов со скоростью 75% от 100 % мониторной скорости

DRAW X translation, Y translation, Z translation, X rotation, Y rotation, Z rotation, speed

TDRAW X translation, Y translation, Z translation, X rotation, Y rotation, Z rotation, speed

Назначение

Движение робота по прямолинейной траектории с заданной скоростью по осям координат X,Y,Z в декартовой системе координат (DRAW), вращение на заданное количество вокруг каждой из осей. TDRAW инструкция движения робота в инструментальной системе координат.

Параметр

X перемещение. Задаёт движение по оси X соответствующей системы координат в миллиметрах. Если параметр не задан, вводится нулевое перемещение.

Y перемещение. Задаёт движение по оси Y соответствующей системы координат в миллиметрах. Если параметр не задан, вводится нулевое перемещение.

Z перемещение. Задаёт движение по оси Z соответствующей системы координат в миллиметрах. Если параметр не задан, вводится нулевое перемещение.

X вращение. Задаёт вращение вокруг оси X в градусах. Допустимые пределы меньше чем +-180 градусов. Если параметр не задан, вводится нулевое вращение.

Y вращение. Задаёт вращение вокруг оси Y в градусах. Допустимые пределы меньше чем +-180 градусов. Если параметр не задан, вводится нулевое вращение.

Z вращение. Задаёт вращение вокруг оси Z в градусах. Допустимые пределы меньше чем +-180 градусов. Если параметр не задан, вводится нулевое вращение.

Скорость. Задаёт скорость в %, мм/сек, см/мин, сек. Если параметр не задан робот движется с программной скоростью.

Пояснение

Робот движется из текущего положения в заданную аппозицию по прямолинейной траектории.

`DRAW 50,-30`

Движется по прямолинейной траектории по оси X на 50 мм, по оси Z на -30 мм.

ALIGN

Назначение

Устанавливает ось Z инструментальной системы координат параллельно ближайшей оси декартовой системы координат.

Пояснение

Если желательное направление движения установлено вдоль направления оси Z инструмента для каждого применения, DO ALIGN разрешает удобное выравнивание во время обучения, совмещая направление оси Z с ближайшей осью базовой системы координат.

HMOVE pose variable name, clamp number

Назначение

Движение робота в заданную позицию. Движение происходит по гибридному типу: старшие суставы двигаются по линейной траектории (1,2,3), суставы запястья по криволинейной траектории.

Параметр

1. Имя заданного положения.

2. Номер фиксатора

Определяет номер фиксатора, чтобы открыться или закрыться в при достижении координаты. Положительный номер закрывает фиксатор, и отрицательный номер открывает его. Любой номер фиксатора может быть установлен в максимально установленный номер через команду HSETCLAMP (или вспомогательную функцию 0605). Если опущено, фиксатор не открывается и не закрывается.

Пояснение

Эта инструкция перемещает робот в прямолинейном интерполированном движении. Начало инструментальной системы координат движется по прямолинейной траектории. Однако суставы запястья двигаются в интерполированном угловом движении. Эта инструкция используется, когда робот должен быть перемещен в прямолинейном движении, но положении суставов запястья должно сильно изменяться между началом и концом движения.

XMOVE mode pose variable name TILL signal number

Назначение

Движение робота вперед к заданной позиции по линейной траектории, если в момент движения возникает заданное сигнальное состояние, движение в данную точку прерывается и происходит переход к следующему шагу программы.

Параметр

1.Режим

Контролирует нарастающий или убывающий фронт заданного входного сигнала. Положительное сигнальное число контролирует нарастающий фронт, отрицательное число – убывающий.

/ERR (опция)

указывает ошибочное состояние, если сигнальное условие было установлено до начала движения

/LVL (опция)

немедленно переходит на следующий шаг, если сигнальное условие было установлено до начала движения.

2.Имя позиции

Задаёт конечную цель движения робота.

3.Номер сигнала

Задаёт номер внешнего входного сигнала или внутреннего сигнала.

Внешние входные сигналы – 1001 до реально смонтированного сигнала

Внутренние сигналы – 2001 - 2256

(ПРИМЕЧАНИЕ)

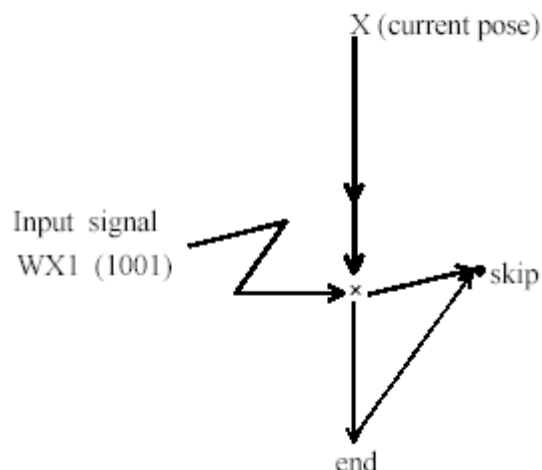
При контролировании нарастающего, понижающего фронтов сигнала, произойдет скачок только тогда, когда произойдет изменение сигнального состояния. Поэтому, если контролируется нарастающий фронт сигнала, и сигнал находится в состоянии ON во время выполнения инструкции XMOVE, программа не будет прервана до тех пор, пока сигнал не примет состояние OFF, а затем снова состояние ON. Входной сигнал должен быть устойчив в течение 50 мсек для точного контролирования.

Пример

XMOVE end TILL 1001

LMOVE skip

Движение из текущей позиции в позицию “end” по линейной траектории. Как только входной сигнал 1001 становится ON, программное выполнение переходит к следующему шагу, даже если робот не достиг позиции “end”.



C1MOVE pose variable name, clamp number
C2MOVE pose variable name, clamp number

Назначение

Движение в заданную позицию по круговой траектории.

Параметр

1.Имя позиции

Задаёт координату точки, в которую должен выйти робот, под определённым именем. Для того чтобы совершить круговое движение необходимо задать три обученных точки. Эти три точки различаются для инструкций C1MOVE и C2MOVE.

. Номер фиксатора

Определяет номер фиксатора, чтобы открыться или закрыться в при достижении координаты. Положительный номер закрывает фиксатор, и отрицательный номер открывает его. Любой номер фиксатора может быть установлен в максимально установленный номер через команду HSETCLAMP (или вспомогательную функцию 0605). Если опущено, фиксатор не открывается и не закрывается.

Пояснение

C1MOVE инструкция должна находиться посередине между двумя инструкциями движения. C2MOVE завершает движение по круговому типу.

C1MOVE

- 1.Обученная точка начала движения (может быть любой инструкцией движения)
- 2.Обученная точка, задаваемая как параметр данной инструкции
- 3.Обученная точка следующего движения (задается как параметр только для C1MOVE или C2MOVE инструкций)

C2MOVE

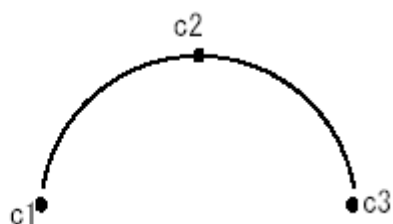
- 1.Точка, задаваемая как параметр в последней C1MOVE инструкции
- 2.Точка, задаваемая как параметр до выполнения последней C1MOVE инструкции
- 3.Точка, задаваемая как параметр C2MOVE инструкции, т.е. окончание круговой интерполяции

(ПРИМЕЧАНИЕ)

Следующие инструкции движения должны использоваться перед инструкцией C1MOVE: ALIGN, C1MOVE, C2MOVE, DELAY, DRAW, TDRAW, DRIVE, HOME, JMOVE, JAPPRO, JDEPART, LMOVE, LAPPRO, LDEPART, STABLE, XMOVE

C1MOVE инструкция должна следовать за C1MOVE или C2MOVE инструкцией. C1MOVE инструкция должна быть раньше C2MOVE инструкции

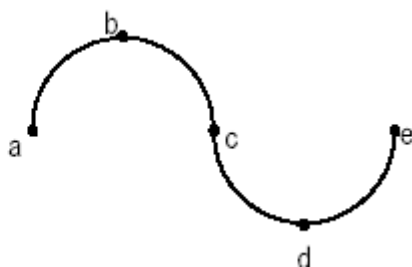
```
JMOVE c1
C1MOVE c2
C2MOVE c3
```



```
JMOVE #a
C1MOVE #b
C2MOVE #c
C1MOVE #d
C2MOVE #e
```

} arc a,b,c

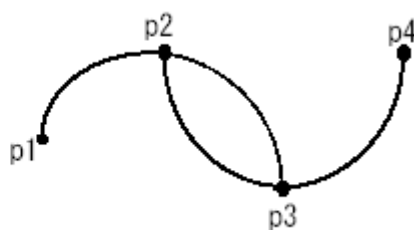
} arc c,d,e



```
LMOVE #p1
C1MOVE #p2
C1MOVE #p3
C2MOVE #p4
```

} arc p1,p2,p3

} arc p2,p3,p4



6.2. ИНСТРУКЦИИ УПРАВЛЕНИЯ СКОРОСТЬЮ И ТОЧНОСТЬЮ

SPEED	Устанавливает скорость движения (программную скорость).
ACCURACY	Устанавливает диапазон точности.
ACCEL	Устанавливает ускорение.
DECEL	Устанавливает замедление.
BREAK	Останавливает выполнение следующего шага до тех пор, пока текущее движение полностью не завершится
BRAKE	Останавливает текущее движение и переходит к следующему шагу.
BSPEED	Устанавливает скорость блочного программирования (опция)

SPEED speed, rotational speed, ALWAYS

Назначение

Задаёт скорость движения робота

Параметр

1. Скорость

Выбирается программная скорость. Обычно она задается в процентах от 0.01 до 100 %. Скорость движения может быть установлена заданием скорости в мм/сек, мм/мин, сек. Если единицы измерения скорости не заданы устанавливаются %.

2. Скорость вращения (опция)

3. Всегда

Если этот параметр введен, скорость остается действительной до выполнения следующей инструкции SPEED. Если параметр не введен, скорость эффективна только для следующей инструкции движения.

Пояснение

Действительная скорость движения робота определяется как результат взаимодействия мониторной скорости и скорости движения, заданной этой инструкцией (Monitor speed x Program speed). Однако, полная скорость движения не всегда вычисляется как показано выше, т.е. в случаях

- 1) когда расстояние между обученными точками очень мало
- 2) когда линейная скорость превышает максимальную скорость вращения осей.

Задаваемая скорость движения различается для криволинейного движения и прямолинейного движения. В криволинейном интерполированном движении скорость движения определяется как процент максимальной скорости для движения каждого сустава, в прямолинейном движении скорость движения определяется как процент максимальной скорости для движения начала инструментальной системы координат.

Когда скорость определена в единицах расстояния и времени или секундах, устанавливается скорость для прямолинейного движения начала инструментальной системы координат. При перемещении в интерполированном угловом движении, устанавливается скорость в процентах. (Даже если скорость установлена в абсолютной скорости или во время движения, робот не будет двигаться с установленной скоростью. Вместо этого, скорость обрабатывается как процент от данного значения к максимальной скорости.)

Абсолютная скорость выражается в значениях с ММ/ С и ММ/МІN, и время, задающее скорость, выражается в значении с S, описывает скорость, когда мониторная скорость 100 %. Если мониторная скорость уменьшается, эти скорости уменьшаются в той же самой пропорции.

(ПРИМЕЧАНИЕ)

Даже если результат программной скорости и скорости, установленной командой SPEED (мониторная скорость) превышает 100 % реальная скорость движения не будет превышать 100 %.

Скорость вращения не может быть установлена без опции управления скоростью вращения, которая должна находиться в состоянии ON. Если опция не включена произойдет ошибка.

Примеры

SPEED 50 Устанавливает скорость следующего движения 50%.

SPEED 100 Устанавливает скорость следующего движения 100%.

SPEED 200 Устанавливает скорость следующего движения 100%.(скорость свыше 100 % понимается как 100 %).

SPEED 20MM/S ALWAYS Скорость движения начала инструментальной системы координат (TCP) установлена 20mm/sec до тех пор, пока она не изменится другой SPEED инструкцией.(когда мониторинг скорости 100 %).

SPEED 6000 MM/MIN Устанавливает скорость следующего движения робота в 6000 mm / min. (Скорость движения начала инструментальной системы координат (TCP), когда мониторинг скорости 100%).

SPEED 5 S Устанавливает скорость следующего движения робота, так что расстояние проходится за 5 секунд (Скорость движения начала инструментальной системы координат (TCP), когда мониторинг скорости 100%).

SPEED 100 MM/S, 10 DEG/S Устанавливает скорость следующего движения. Установка скорости, которая требует больше времени , чтобы достигнуть координату, делается приоритетной.

ACCURACY distance ALWAYS

Назначение

Подход в точку с заданной точностью.

Параметр

1.Расстояние

Задаёт расстояние предела точности в миллиметрах. Т.е. при достижении этого предела робот прекращает движение в точку.

2.Всегда

Если этот параметр введен, точность остается действительной до выполнения следующей инструкции ACCURACY. Если параметр не введен, точность эффективна только для следующей инструкции движения.

Пояснение

Когда параметр ALWAYS введен, все выполняемые движения управляются при помощи точности, установленной этой инструкцией.

По умолчанию точность устанавливается 1 миллиметр.

Это предел эффективности установки точности, т.к. в AS системе проверка точности не начинается до тех пор, пока робот не начнет торможение для прихода в обученную точку. (см. 4.5.4)

(ПРИМЕЧАНИЕ)

Когда точность установлена 1 мм, робот определяет позицию в соответствии с каждой инструкцией движения, в наступающих паузах между сегментами движения. Чтобы обеспечить плавность при задании CP движений, задавайте пределы более большие.

Нельзя устанавливать более маленькие пределы точности, т.к. это может привести к рассогласованию осей.

Точность, задаваемая этой инструкцией, не является точностью повторяемости и позиционирования робота.

ACCURACY 10 ALWAYS

ACCEL acceleration	ALWAYS
DECEL acceleration	ALWAYS

Назначение

Задаёт ускорение (замедление) движений робота

Параметры

1. Ускорение (замедление)

Выбирает ускорение (замедление) в процентах от максимального ускорения (замедления). Допустимые границы от 0.01 до 100.

2. Всегда

Если этот параметр введен, точность остается действительной до выполнения следующей инструкции ACCEL (DECEL). Если параметр не введен, точность эффективна только для следующей инструкции движения.

Пояснение

ACCEL инструкция устанавливает ускорение, когда робот начинает движение, как процент от максимального ускорения. DECEL инструкция устанавливает замедление, когда робот заканчивает движение, как процент от максимального замедления.

ACCEL 80 ALWAYS

DECEL 50

BREAK

Назначение

Задерживает выполнение следующего шага программы, пока текущее движение робота незакончено полностью.

Пояснение

Эта функция имеет следующие два следствия

1. Задерживает выполнение программы, пока робот не достигнет назначения текущей инструкции движения.
2. Если CP движение от текущей позиции к следующей позиции сделать прерывистым, т.е. произвести задержку движения, не изменяя состояние CP системного переключателя.

BRAKE

Назначение

Прерывает текущее движение робота и переходит к следующему шагу программы.

BSPEED speed

Опция

Назначение

Устанавливает скорость движения роботов (блочную скорость). Скорость движения робота вычисляется при помощи $\text{monitor speed} \times \text{program speed} \times \text{block speed}$.

Параметр

Скорость

Устанавливает скорость (приемлемый диапазон: 1 - 1000%) . Скорость, установленная этой инструкцией действительна до тех пор, пока не выполнится следующая инструкция BSPEED.

Пояснение

Скорость движения робота вычисляется по формуле $\text{monitor speed} \times \text{program speed} \times \text{block speed}$. Однако, полная скорость не может превысить 100 %. Значения вплоть до 1000 могут быть введены для каждой скорости, но если полная скорость превышает 100%, она автоматически устанавливается в 100%. Для примера, если мониторинговая скорость 100% и программная скорость 50 %, скорость движения вычисляется как $100\% \times 50\% \times \text{block speed}$. Поэтому, если блочная скорость меньше чем 200%, скорость изменяется в зависимости от результата выше указанного выражения, но если скорость выше 200% , скорость движения всегда 100%

(ПРИМЕЧАНИЕ)

1. Когда программа выполнена, используя команду EXECUTE с пульта ручного управления, блоковая скорость установлена в значении по умолчанию 100 %. Когда программа, выбирается извне, блоковая скорость устанавливается как значение по умолчанию, если программа выбрана внешним сбросом программы, но не сигналами RPS и JUMP.

2. Обратите внимание, что робот не может двигаться с указанной программной скоростью если программа не выполняется с начала программы, или когда шаги пропущены. В примере ниже, робот останавливается в шаге 3, и движение продолжается после перехода в шаг 25. Тогда, блоковая скорость при шаге 25 будет скоростью блока 1.

Step 1	BSPEED	block1
Step 2	Joint	Speed 9.....
Step 3	Linear	Speed 9.....
Step 12	BSPEED	block2
Step 13	Joint	Speed 9.....
Step 14	Linear	Speed 9.....
Step 24	BSPEED	block3
Step 25	Joint	Speed 9.....
Step 26	Linear	Speed 9.....

Пример

Запишите программу следующим образом так, чтобы скорость была изменена на 4 бита от внешнего сигнала.

```
a=BITS(first signal for external speed selection,4)
BSPEED block1[a]
```

Следующая программа разрешает выбор скорости от внешнего устройства:

```
BSPEED block1 ;устанавливает значение по умолчанию.
IF SIG(External_speed ON)THEN ;определяет разрешен ли выбор от внешнего сигнала.
a=BITS(first signal for external speed selection,4) ; получает номер, используемый для
; внешнего сигнала
IF(a<11)THEN ;Установка невозможна, если a >11
BSPEEDblock11[a] ; устанавливает выбранный блок скорости
END
END
Joint Speed 9..... ; перемещается с выбранной блоковой скоростью.
Joint Speed 9.....
```

Переменная реального числа “block 1” должна быть определена заранее.

```
block1=50
block11[0]=10
block11[1]=20
block11[2]=30
block11[3]=40
```

Для примера, если первый сигнал внешнего программного выбора 1010, и сигналы введены как:

1010 · · · OFF

1011 · · · ON

1012 · · · OFF

1013 · · · OFF

При этом $a = 2$, поэтому выбирается блок 11[2] и скорость движения становится 30%.

6.3 ИНСТРУКЦИИ УПРАВЛЕНИЯ ФИКСАТОРАМИ

OPEN	Выводит сигнал открытия фиксатора, когда начинается следующая р
OPENI	Выводит сигнал открытия фиксатора, когда текущая инструкция движения завершается
CLOSE	Выводит сигнал закрытия фиксатора, когда начинается следующая инструкция движения.
CLOSEI	Выводит сигнал закрытия фиксатора, когда текущая инструкция движения завершается.
RELAX	Устанавливает OFF для сигнала фиксатора, когда начинается следующая инструкция движения.
RELAXI	Устанавливает OFF для сигнала фиксатора, когда завершается инструкция движения.
OPENS	Выводит сигнал открытия фиксатора во время выполнения инструкции движения (опция)
CLOSES	Выводит сигнал закрытия фиксатора во время выполнения инструкции движения (опция)
RELAXS	Устанавливает OFF для сигнала фиксатора во время выполнения инструкции движения (опция)
GUNON	Устанавливает ON для сигнала сварочным клещам и управляет клещами выходной синхронизацией при помощи расстояния. (опция)
GUNOFF	Устанавливает OFF для сигнала сварочным клещам и управляет цифровым выходом клещей при помощи расстояния. (опция)
GUNONTIMER	Управляет состоянием ON цифрового выхода клещей при помощи таймера (опция)
GUNOFFTIMER	Управляет состоянием OFF цифрового выхода клещей при помощи таймера (опция)

OPEN clamp number
OPENI clamp number

Назначение

Открывает фиксаторы робота (выводит сигнал открытия фиксатора)

Параметр

Выбирает номер фиксатора. Если параметр пропущен, выбирает первый фиксатор.

Пояснение

Эта инструкция выдает сигналы для контролирования клапанов пневматического схвата при открытии фиксатора.

С OPEN инструкцией сигнал не выйдет до тех пор, пока не начнется следующее движение.

Синхронизация для вывода сигнала, используя команду OPENI, следующая:

1. Если робот находится в настоящее время в движении, сигнал выводится после того, как движение закончено. Если робот двигается в режиме СР, движение СР временно приостанавливается.
2. Если робот не находится в движении, сигнал посылают немедленно клапану управления.

Пример

OPEN

OPENI 2

Сигнал открытия фиксатора послан для контроля клапана фиксатора 1, когда робот начинает следующее движение.

Сигнал открытия фиксатора послан для контроля клапана фиксатора 2, как только завершилось текущее движение.

CLOSE clamp number
CLOSEI clamp number

Назначение

Закрывает фиксаторы робота (выводит сигнал закрытия фиксатора)

Параметр

Выбирает номер фиксатора. Если параметр пропущен, выбирает первый фиксатор.

Пояснение

Эта инструкция выдает сигналы для контролирования клапанов пневматического схвата при закрытии фиксатора.

С CLOSE инструкцией сигнал не выйдет до тех пор, пока не начнется следующее движение.

Синхронизация для вывода сигнала, используя команду CLOSEI, следующая:

1. Если робот находится в настоящее время в движении, сигнал выводится после того, как движение закончено. Если робот двигается в режиме СР, движение СР временно приостанавливается.
2. Если робот не находится в движении, сигнал посылают немедленно клапану управления.

Пример
CLOSE 3

CLOSEI

Сигнал закрытия фиксатора послан для контроля клапана фиксатора 3, когда робот начинает следующее движение.

Сигнал закрытия фиксатора послан для контроля клапана фиксатора 1, как только завершилось текущее движение.

RELAX clamp number RELAXI clamp number

Назначение

Выключает пневматический электромагнитный клапан для OPEN, CLOSE (устанавливает сигналы фиксатора в положение OFF. Для двойного электромагнитного клапана, оба сигнала и открытия, и закрытия устанавливаются в состояние OFF).

Параметр

Номер фиксатора

Выбирает номер фиксатора. Если параметр пропущен, устанавливается первый фиксатор.

Пояснение

При использовании этой инструкции, сигнал не появится до тех пор, пока не начнется следующее движение

Синхронизация для вывода сигнала, используя команду RELAXI, следующая:

1. Если робот находится в настоящее время в движении, сигнал выводится после того, как движение закончено. Если робот двигается в режиме СР, движение СР временно приостанавливается.
2. Если робот не находится в движении, сигнал посылают немедленно клапану управления.

OPENS clamp number
CLOSES clamp number
RELAXS clamp number

Опция

Назначение

Включает ON/OFF для сигналов открытия, закрытия пневматических электромагнитных клапанов.

Параметр

Выбирает номер фиксатора. Если параметр пропущен, выбирается первый фиксатор.

Эта инструкция отличается от выше изложенных. Сигнал выводится немедленно после выполнения этих инструкций.

GUNON gun number , distance
GUNOFF gun number , distance

Опция

Назначение

Включает, выключает сигнал клещей и управляет цифровым выходом клещей при помощи выбранного расстояния.

Параметры

1.Номер клещей

Выбирает номер 1 или 2

2. Расстояние

Выбирает расстояние в миллиметрах, для того чтобы урегулировать ON/OFF синхронизацию для клещей. Отрицательное значение ускоряет синхронизацию, положительное значение задерживает синхронизацию. Если параметр не задан, устанавливается 0.

Пояснение

Сигнал клещей устанавливается в состояние ON/OFF, когда после выполнения соответствующих инструкций, выполняется команда движения. Синхронизация вывода определяется расстоянием, заданным в этой инструкции или временем, установленным в инструкциях GUNONTIMER/GUNOFFTIMER.

GUNONTIMER gun number , time
GUNOFFTIMER gun number , time

Опция

Назначение

Регулирует синхронизацию выхода клещей (промежуток, в котором клещи устанавливаются в состояние ON/OFF) при помощи заданного времени.

Параметры

Номер клещей

Выбирает номер 1 или 2

Время

Выбирает время в секундах, для того чтобы урегулировать ON/OFF синхронизацию для клещей. Отрицательное значение ускоряет синхронизацию, положительное значение задерживает синхронизацию. Если параметр не задан, устанавливается 0.

Пояснение

Время регулировки определено средой вокруг системы клещей (т.е. расстояние от клапана до конца клещей, тип наносимого материала, климат и т.д.) , так что установка синхронизации должна быть в начале программы. Всегда используйте переменную для этого параметра, для возможности изменить синхронизации снаружи программы (когда синхронизация должна быть изменена при смене наносимого материала).

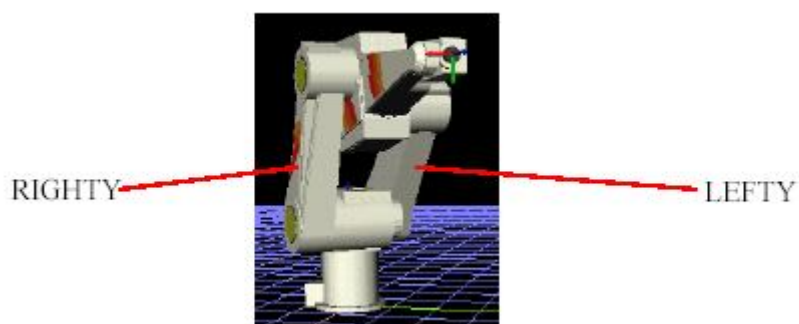
6.4. ИНСТРУКЦИИ КОНФИГУРАЦИИ РОБОТА

RIGHTY человека.	Изменяет конфигурацию так, что рука робота напоминает правую руку человека.
LEFTY человека.	Изменяет конфигурацию так, что рука робота напоминает левую руку человека.
ABOVE	Изменяет конфигурацию так что локтевой находится сверху.
BELOW	Изменяет конфигурацию так что локтевой находится снизу.
UWRIST значение.	Изменяет конфигурацию так, что угол 5 сустава имеет положительное значение.
DWRIST значение.	Изменяет конфигурацию так, что угол 5 сустава имеет отрицательное значение.

RIGHTY
LEFTY

Назначение

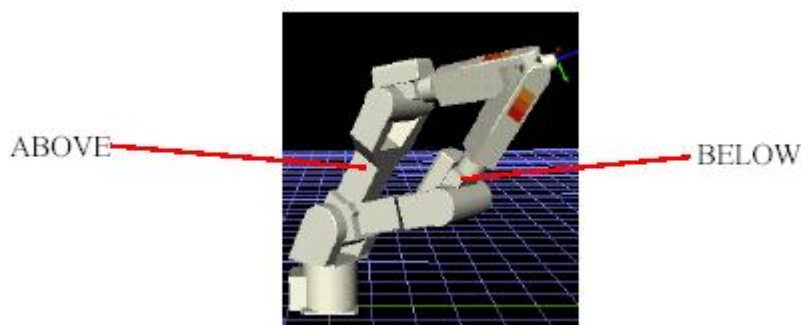
Вынуждает изменить конфигурацию руки робота во время следующего движения, так что рука робота будет напоминать правую или левую руку человека, т.е. происходит изменение положения первых трех суставов, а положение запястья в пространстве не изменяется. Конфигурация не может быть изменена во время линейного интерполированного движения, или когда точка назначения задана в угловых величинах.



ABOVE
BELOW

Назначение

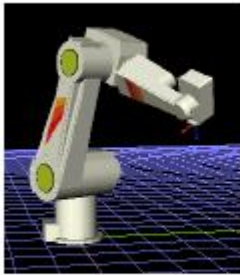
Вынуждает изменить конфигурацию руки робота во время следующего движения, так что локоть руки робота (третий сустав) будет напоминать локоть руки человека в положении локоть вверх (локоть вниз) относительно запястья, положение запястья в пространстве не изменяется. Конфигурация не может быть изменена во время линейного интерполированного движения, или когда точка назначения задана в угловых величинах.



UWRIST DWRIST

Назначение

Вынуждает изменить конфигурацию руки робота во время следующего движения, так что угол пятого сустава робота принимает положительное или отрицательное значение. Конфигурация не может быть изменена во время линейного интерполированного движения, или когда точка назначения задана в угловых величинах.



UWRIST
(Joint 5 is 90°)



DWRIST
(Joint 5 is -90°)*

6.5. ИНСТРУКЦИИ УПРАВЛЕНИЯ ПРОГРАММОЙ

GOTO	Переходит к определенной метке.
IF	Переходит к определенной метке, когда условие выполнено.
CALL	Переходит к подпрограмме.
RETURN	Возвращается в программу, которая вызвала подпрограмму.
WAIT	Помещает программное выполнение в готовность продолжить действие, как только состояние установится.
TWAIT	Помещает программное выполнение в готовность продолжить действие, как только установленное время пройдет
MVWAIT	Помещает программное выполнение в готовность продолжить действие, как только данное расстояние или время будет достигнуто.
LOCK	Изменяет приоритет управляющих программ робота.
PAUSE	Приостанавливает программное выполнение.
HALT	Прекращает программное выполнение. (Продолжение невозможно)
STOP	Останавливает цикл выполнения.
SCALL	Переходит к подпрограмме.
ONE	Вызывает программу, когда происходит ошибка.
RETURNE	Выполняет с шага, следующего за шагом, где произошла ошибка
CALLAUX	Отображает экран вспомогательных функций. (опция)

GOTO lable IF condition

Назначение

Переходит на программный шаг, в соответствии с меткой

Параметр

1. Метка

Задается метка программного шага для перехода к этому шагу. Метка задается целым числом от 0 до 32767

2. Условие

Задаёт условие для перехода, т.е. если условие выполнено происходит переход на шаг программы с заданной меткой. Если параметр не задан, переход происходит автоматически на заданную метку.

Следует различать метку и номер шага программы. Каждый шаг программы имеет свой номер, но может не иметь метки

Пояснение

Переходит к шагу, указанному меткой. Если состояние определено, программа делает переход, когда состояние установлено. Если состояние не установлено, выполнение продолжается к следующему шагу после этой инструкции.

Обратите внимание, что метка и номер шага отличаются. Номера шагов присваиваются всем шагам программы автоматически системой. Метки присваиваются выборочно в программные шаги и вводятся после номера шага.

Функции команды те же самые, как у команды IF GOTO, когда состояние определено.

```
GOTO 100
```

```
GOTO 200 IF n==3
```

IF condition GOTO lable

Назначение

Переходит к шагу с указанной меткой, когда данное состояние установлено.

Параметр

1. Состояние

Определяет условие в выражениях, например $n = 0$, $n > 3$, $m + n < 0$.

2. Метка

Определяет метку шага, в который надо перейти (не номер шага). Метка должна быть в пределах этой же программы.

Объяснение

Переходит к шагу, указанному меткой. Если состояние определено, программа делает переход, когда состояние установлено. Если состояние не установлено, выполнение

продолжается к следующему шагу после этой инструкции. Если указанная метка не существует, происходит ошибка.

Если значение целой переменной номера “n” больше чем 3, то программа переходит к шагу с меткой 100. Если n не больше чем 3, тогда шаг после этого шага выполняется. Если значение целой переменной - не 0, программа переходит к шагу с меткой 25. Если значение переменной равно 0, программа переходит на следующий шаг выполнен. Это то же самое как запись: IF flag <> 0 GOTO 25.

```
IF n>3 GOTO 100
```

```
IF flag GOTO 25
```

CALL program name

Назначение

Прекращает выполнение программы и переходит к выполнению другой программы или подпрограммы, начиная с первого шага. Когда выполнение подпрограммы закончено, происходит возврат в головную программу на шаг, следующий за этой инструкцией.

Параметр

Программное имя

Выбирает имя подпрограммы для выполнения.

Пояснение

Эта команда временно останавливает выполнение текущей программы и переходит к первому шагу указанной подпрограммы.

(ПРИМЕЧАНИЕ)

Одна и та же подпрограмма не может одновременно вызываться из управляющих программ робота и РС программ, т.к. подпрограмма не может вызвать сама себя. До 20 программ могут быть приостановлены, когда вызываются подпрограммы.

```
CALL sub1
```

Переходит в подпрограмму, названную “sub1”. Когда команда RETURN в “sub1” выполнена, программное выполнение возвращается к первоначальной программе, и выполняет программу с шага после этой инструкции CALL.

RETURN

Назначение

Заканчивает выполнение подпрограммы и возвращает на шаг головной программы, следующий за CALL инструкцией.

Пояснение

Эта инструкция заканчивает выполнение подпрограммы и возвращается в программу, которая вызвала подпрограмму. Если подпрограмма не вызывается другой программой (Если подпрограмма выполняется в мониторном режиме при помощи команды EXECUTE) программное выполнение заканчивается.

В конце подпрограммы программное выполнение возвращается в первоначальную программу, даже если нет инструкции RETURN. Однако инструкция RETURN должна быть записана как последний шаг подпрограммы (или в любом месте подпрограммы, для того чтобы ее завершить).

WAIT condition

Назначение

Заставляет ждать выполнение определенного состояния для продолжения выполнения программы

Параметр

Выбирает дублирующее условие. (Реальное числовое выражение)

Пояснение

Эта инструкция останавливает выполнение программы до выполнения определенного условия. CONTINUE NEXT команда продолжает программное выполнение, не дожидаясь установки условия. (перепрыгивает инструкцию WAIT)

WAIT SIG(1001, – 1003) приостанавливает выполнение программы до тех пор, пока внешний входной сигнал 1001 (WX1) не станет ON и сигнал 1003(WX3) не станет OFF.

WAIT TIMER(1)>10 приостанавливает выполнении программы, пока значение таймера не превысит 10 секунд.

WAIT n>100 приостанавливает выполнении программы, пока значение переменной “n” не превысит 100. (в этом примере предполагается. Что значение переменной “n” считается либо в РС программе, либо в программе прерывания.)

TWAIT time

Назначение

Останавливает программное выполнение до истечения определенного времени.

Параметр

Задается время в секундах для останова выполнения программы.

Пояснение

Эта инструкция приостанавливает программное выполнение до тех пор, пока не пройдет заданное время.

А TWAIT инструкция при выполнении может быть перепрыгнута при помощи команды CONTINUE NEXT.

WAIT инструкция может быть использована вместо TWAIT инструкции, получится тот самый результат.

```
TWAIT 0.5
```

```
TWAIT deltat
```

MVWAIT value

Назначение

Останавливает программное выполнение пока оставшееся расстояние или время текущего движения не становится меньше чем заданное расстояние или время.

Параметр

Значение

Задаёт расстояние в миллиметрах или время в секундах. Если единица измерения не задана, по умолчанию воспринимается как миллиметры.

Пояснение

Эта инструкция используется для одновременного выполнения программы и движения робота. Однако обратите внимание, что после того как инструкция отследит оставшееся расстояние (время), заданное в параметре, оно может отличаться от действительно оставшегося расстояния (времени) из-за задержки ответа. Когда робот совершает движение по криволинейной траектории, заданное расстояние и действительное могут значительно различаться. Если текущее движение завершено полностью во время выполнения этой инструкции, выполнение программы продолжается. CONTINUE NEXT команда может продолжить выполнение программы, пропуская эту инструкцию.

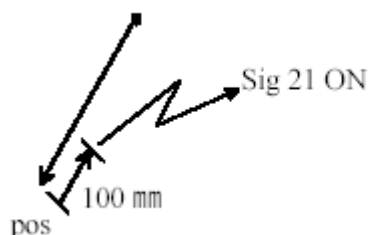
(ПРИМЕЧАНИЕ)

MVWAIT инструкция не может использоваться в PC программах, также она не является мониторной командой.

Пример

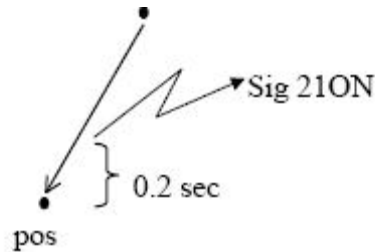
На диаграмме ниже, робот движется по направлению к позиции “pos”, и когда он войдет внутрь 100 мм до “pos”, выходной сигнал 21 включится. Это верно только тогда, когда сигнал считывается впереди стоящей функцией (PREFETCH.SIGINS) в состоянии ON и робот находится в диапазоне точности.

```
LMOVE Ic  
MVWAIT 100mm  
SIGNAL 21
```



На диаграмме ниже, робот движется по направлению к позиции “pos”, и когда требуемое время для достижения точки станет 0.2 секунды, выходной сигнал 21 включится. Это верно только тогда, когда сигнал считывается впереди стоящей функцией (PREFETCH.SIGINS) в состоянии ON и робот находится в диапазоне точности

```
LMOVE Ic  
MVWAIT 0.2S  
SIGNAL 21
```



LOCK priority

Назначение

Меняет приоритет программы, в настоящее время выбранной в стеке.

Параметр

Задаёт приоритет в реальных числах от 0 до 127

Пояснение

Нормально, когда приоритет программы робота 0. Приоритет может быть изменен при помощи этой инструкции. Чем больше число, тем выше приоритет.

Пример

```
LOCK 2
```

Изменяет приоритет программы на 2.

PAUSE

Назначение

Временно останавливает программное выполнение

Пояснение

Эта инструкция временно останавливает выполнение программы и отображает сообщение на терминале. Выполнение может быть продолжено CONTINUE командой.

Использование инструкции удобно в наладочном (проверочном) режиме. Значения переменных могут быть проверены во время программного останова инструкцией PAUSE.

HALT

Назначение

Останавливает программное выполнение. Программа не может быть продолжена после выполнения этой инструкции никакими командами, не взирая на оставшиеся шаги программы. На терминале появляется сообщение.

Пояснение

Останавливает выполнение программы, независимо от оставшихся шагов. На экране появляется сообщение.

Выполнение программы, прерванное этой инструкцией не может быть возобновлено использованием команды CONTINUE.

STOP

Назначение

Заканчивает текущее выполнение цикла.

Пояснение

Если есть циклы, которые должны быть завершены, выполнение возвращается на первый шаг, иначе выполнение заканчивается. Инструкция обозначает конец выполняемой части программы и отличается от HALT инструкции.

Если есть оставшиеся для выполнения циклы, выполнение продолжается с первого шага основной программы* (даже если STOP инструкция была выполнена во время выполнения подпрограммы или программы прерывания, выполнение возвращается в основную программу).

ПРИМЕЧАНИЕ* Основная программа – программа, выполняемая при помощи использования команд EXECUTE, STEP, PSEXECUTE

Подпрограмма – это программа, вызываемая из другой программы при помощи инструкций CALL, ON или ONI.

RETURN инструкция в основной программе функционирует тем же способом как STOP инструкция. Программное выполнение, остановленное инструкцией STOP не может быть продолжено командой CONTINUE.

SCALL string expression, variable

Назначение

Переходит к выполнению программы с именем, данным посредством строчного выражения

Параметры

1.Строчное выражение

Задаёт имя подпрограммы в виде строчного выражения.

2.Переменная

Если подпрограммный вызов выполняется нормально, значение 0 присваивается этой переменной. Если некоторая ненормальность произошла в течение вызова подпрограммы, назначается код ошибки (не 0). Если параметр опущен, выполнение приходит в останов по ошибке, когда неправильность происходит в вызванной подпрограмме.

Пояснение

Эта инструкция функционирует также как CALL инструкция за исключением того, что программное имя выражается строчным выражением.

Пример

```
$prog="sub1"
```

```
SCALL $prog
```

Переходит к подпрограмме с именем "sub1"

```
num=12
```

```
$temp1=$ENCODE(/I2,num)
```

```
$temp2=""
```

Преобразовывает в строчное выражение реальную величину, заданную переменной "num" и переходит к выполнению подпрограммы "sub12".

```
FOR i=1 to LEN($temp1)
```

```
$temp3=$MID($temp1,i,1)
```

```
IF $temp3<> "" THEN
```

```
$temp2=$temp2+$temp3
```

```
END
```

```
END
```

```
SCALL "sub"+$temp2
```

ONE program name

Назначение

Вызывает определенные программы, когда происходит ошибка.

Пояснение

Эта инструкция вызывает определенные программы, когда происходит ошибка. РС программа также может быть вызвана.

RETURN инструкция возвращает выполнение в шаг, где ошибка произошла. RETURNE инструкция возвращает выполнение на шаг после ошибки. Если ни та, ни другая инструкция не встречается внутри программы, выполнение цикла останавливается в конце вызванной программы.

Инструкции движения не могут быть использованы в программах, вызываемых инструкцией ONE.

Если ошибка происходит в программе, вызываемой ONE, программное выполнение останавливается на этом месте.

(ПРИМЕЧАНИЕ)

Пока основная программа, содержащая данную инструкцию, выполняется, эффективность ее распространяется и на подпрограммы, вызываемые в основной программе. Если вызов программы осуществляется ONE инструкцией, и ошибка произошла, лампочка ERROR не загорается.

RETURNE

Назначение

Возвращается в шаг, следующий за ошибкой.

Эта инструкция, как правило, парна с ONE инструкцией. С ONE инструкцией программа переходит в подпрограмму, когда происходит ошибка. Затем выполнение возвращается в шаг после ошибки в первоначальную программу, когда выполнится инструкция RETURNE в подпрограмме.

CALLAUX auxiliary function number

Назначение

Отображает указанный экран вспомогательных функций.

Параметр

Номер вспомогательных функций

Указывает номер вспомогательной функции от 1 до XXXX.

Пояснение

1. Разрешает от отображение и установку вспомогательной функции из программы.
2. Если программное выполнение останавливается, в то время как начинает выполняться инструкция CALLAUX , экран вспомогательных функций отключается. Если инструкция CALLAUX выполняется в нескольких программах, CALLAUX инструкция ожидает до тех пор, пока первая инструкция CALLAUX завершится.
3. Если задан неопределенный номер вспомогательной функции в этой инструкции , произойдет ошибка Error (P2030) Undefined function number. Задавайте номер от 1 до XXXX. (Обратите внимание, что не каждый номер присвоен функциям).

6.6. ИНСТРУКЦИИ ПОСТРОЕНИЯ ПРОГРАММНЫХ СТРУКТУР

IF.....THEN...ELSE.....END

WHILE.....DO.....END

DO.....UNTIL

FOR.....END

CASE.....OF.....VALUE.....ANY.....END

```
IF logical expression THEN  
  program instruction(1)  
ELSE  
  program instruction(2)  
END
```

Назначение

Выполняет группу программных шагов в соответствии с результатом логического выражения.

Параметры

1. Логическое выражение

Логическое выражение или реальное значение выражения. Проверяет если это значение TRUE (не 0) или FALSE (0).

2. Программные инструкции (1)

Программные инструкции, введенные здесь, выполняются, если вышеупомянутое логическое выражение истинно.

3. Программные инструкции (2)

Программные инструкции, введенные здесь, выполняются, если вышеупомянутое логическое выражение ложно.

Пояснение

1. Вычисляется логическое выражение и переходит к шагу 4, если выражение ложно или 0.

2. Вычисляется логическое выражение, и выполняются программные инструкции (1), если результат 1 или истинно.

3. Переход к шагу 5.

4. Если есть ELSE состояние, выполняются программные инструкции (2)

5. Продолжается выполнение программы с шага, следующего после END.

(ПРИМЕЧАНИЕ)

ELSE и END должны быть в строчке единственными словами. IF...THEN структура может заканчиваться только END состоянием.

В примере, показанном ниже, если n больше 5, скорость устанавливается 10%, если нет, скорость устанавливается 20%.

```
21    IF n>5 THEN  
22          sp=10  
23    ELSE  
24          sp=20  
25    END  
26    SPEED sp ALWAYS
```


В следующем примере, во-первых, проверяется значение переменной “m”. Если “m” не равно 0, программа проверяет внешний входной сигнал 1001 и отображает различные записи в зависимости от состояния сигнала. В данном случае внешняя IF структура не имеет состояния ELSE.

```
71    IF m THEN
72        IF SIG(1001) THEN
73            PRINT"Input signal is TRUE"
74        ELSE
75            PRINT"Input signal is FALSE"
76        END
77    END
```

WHILE condition DO
program instruction
END

Назначение

В то время, как определенное условие истинно, выполняются программные инструкции. Когда условие ложно, WHILE состояние перескакивается.

Параметры

1. Условие

Логическое условие или реальное значение условия. Проверяет если это значение TRUE (не 0) или FALSE (0).

2. Программные инструкции

Программные инструкции, введенные здесь, выполняются, если вышеупомянутое логическое выражение истинно.

Пояснение

1. Вычисляется логическое выражение и переходит к шагу 4, если выражение ложно или 0.
2. Вычисляется логическое выражение, и выполняются программные инструкции, если результат (1) или истинно.
3. Переход к шагу 1.
4. Продолжается выполнение программы с шага, следующего после END.

(ПРИМЕЧАНИЕ)

В отличие от структуры DO, если состояние FALSE (ложно), ни один программный шаг в структуре WHILE не выполняется.

Когда эта структура используется, в конечном итоге условие должно измениться из TRUE (истина) в FALSE (ложь).

В следующем примере контролируются внешние входные сигналы 1001 и 1002 и робот останавливает движение, основываясь на состоянии этих сигналов. Когда и тот и другой

сигналы от двух транспортеров изменяются в 0 (транспортеры пусты), робот останавливается, и выполнение программы продолжается с шага, идущего после состояния END.

Если один из транспортеров пуст во время начала выполнения структуры WHILE ни один из шагов структуры не выполняется, происходит скачок на шаг 27.

Пример

```
20 .
21 .
22 .
23 WHILE SIG(1001,1002) DO
24     CALL part1
25     CALL part2
26 END
27 .
28 .
29 .
30 .
```

DO **program instruction** **UNTIL logical expression**

Назначение

Создает цикл

Параметры

1. Программные инструкции

Эти инструкции повторяются до тех пор, пока логическое выражение ложно.

2. Логическое выражение

Логическое выражение или реальное значение выражения. Когда результат логического выражения меняется на истинное, выполнение программных инструкций в этой структуре завершается.

Пояснение

Эта структура управления технологическим процессом выполняет группу программных инструкций в то время как данное условие (логическое выражение) является ЛОЖНЫМ.

1. Выполняет программные инструкции.

2. Проверяет значение логического выражения и если результат ложный, шаг 1 повторяется. Если результат истинный, переходит к шагу 3.

3. Продолжает программное выполнение с шага, следующего за UNTIL состоянием.

Выполнение структуры DO осуществляется до тех пор, пока значение логического выражения изменится от FALSE к TRUE.

(ПРИМЕЧАНИЕ)

В отличие от структуры WHILE, программные инструкции в структуре DO выполняются по крайней мере один раз.

Программные инструкции между DO состоянием и UNTIL состоянием могут быть опущены. Если инструкций нет, логическое выражение после UNTIL оценивается повторно. Когда значение логического выражения изменяется в TRUE (истина), тогда заканчивается выполнение цикла и происходит переход на шаг после структуры UNTIL. DO структура должна заканчиваться состоянием UNTIL.

В примере, показанном ниже, DO структура управляет следующей задачей: деталь поднимается и переносится в буфер. Когда буфер заполняется, двоичный входной сигнал “buffer.full” принимает состояние ON. Робот останавливается, и начинаются другие операции.

Пример

```
10      .  
11      .  
12      .  
13      DO  
14          CALL get.part  
15          CALL put.part  
16      UNTIL SIG(buffer.full)  
17      .  
18      .  
19      .
```

FOR loop variable=start value TO end value STEP step value
program instruction
END

Назначение

Повторяет программное выполнение

Параметры

1.Переменная управления циклом

Переменная или реальное значение. Эта переменная первая и устанавливается как начальное значение, единица прибавляется все время, пока цикл выполняется.

2.Начальное значение

Реальное значение или выражение. Устанавливает первое значение для переменной управления циклом.

3.Конечное значение

Реальное значение или выражение. Это значение сравнивается с существующим значением переменной управления циклом и, если переменная управления циклом меньше или равна конечному значению, цикл выполняется, если больше цикл завершается.

4. Величина шага

Реальное значение или выражение, которое может быть пропущено. Это значение прибавляется или вычитается с переменной управления циклом после каждого цикла. Вводится эта команда с STEP состоянием. Если величина шага не задана, прибавляется 1.

Пояснение

Эта структура управления технологическим процессом повторяет выполнение программных инструкций между FOR и END состояниями. Переменная цикла увеличивается на значение заданного шага каждый раз, когда цикл повторяется.

1. Начальное значение присваивается переменной управления циклом.
2. Вычисляется конечное значение и величина шага.
3. Сравнивается значение переменной управления циклом с конечным значением.
 - А. Если величина шага положительна и переменная управления циклом больше чем конечное значение происходит переход на шаг 7.
 - Б. Если величина шага отрицательна и переменная управления циклом меньше чем конечное значение происходит переход на шаг 7.
- В других случаях переход на шаг 4.
4. Выполняются программные инструкции после FOR состояния.
5. Когда END состояние достигнуто, величина шага прибавляется к переменной управления циклом.
6. Возврат к шагу 3.
7. Выполняются программные инструкции после END состояния.

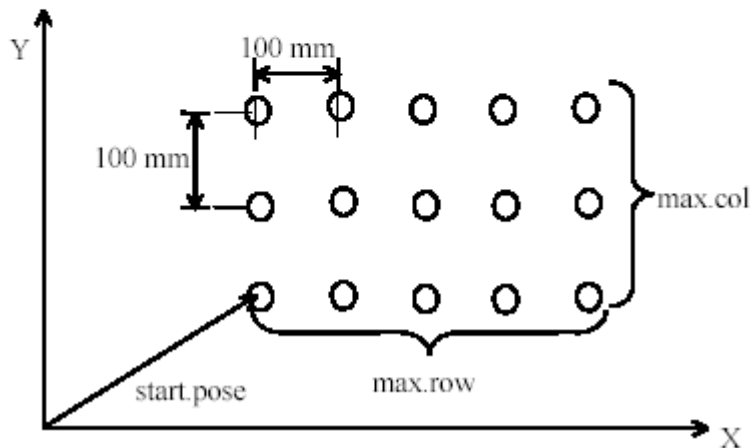
(ПРИМЕЧАНИЕ)

Должна быть инструкция END для каждой инструкции FOR.
Остерегайтесь того, что если переменная цикла больше чем конечное значение (меньше, если значение шага - отрицательное) в первый раз, когда происходит проверка, ни одна из инструкций программы между FOR и END не выполнится.
Значение для количества циклов (переменная цикла) не должна изменяться другими программными средствами (операторы, выражения, и т.д.) для FOR цикла.

В примере, показанном ниже, подпрограмма “pick.place” поднимает детали и располагает их в точках “hole”. Детали располагаются, как показано на рисунке. Паллета размещена параллельно осям X, Y рабочей системы координат и расстояние между деталями 100 мм.

Пример

```
FOR row = 1 TO max.row
POINT hole = SHIFT (start.pose BY (row-1)*100,0,0)
FOR col = 1 TO max.col
CALL pick.place
POINT hole = SHIFT(hole BY 0,100,0)
END
END
```



CASE index variable OF
VALUE case number 1,....
program instructions
VALUE case number 2,....
program instructions
.....
.....
VALUE case number n,....
program instructions
ANY :
program instructions
END

Назначение

Выполняет программу в соответствии с особенным случайным числом.

Параметры

1.Индексная переменная

Реальная переменная или выражение. Выбирает, какая CASE структура должна выполняться в соответствии со значением этой переменной.

2.Программные инструкции

Выполняет программные инструкции, когда значение индексной переменной равняется одному из значений после VALUE состояния.

Пояснение

Эта структура дает возможность программе выбрать из некоторого количества нескольких групп инструкций и обработать выбранную группу. Это - мощный инструмент в AS языке, который обеспечивает удобный метод для того, чтобы позволять несколько альтернатив в пределах программы.

- 1.Проверяет значение индексной переменной, введенной после CASE состояния.
- 2.Проверяет поочередно все VALUE шаги и находит первый шаг, который содержит в себе значение равное значению индексной переменной.
- 3.Выполняет инструкции после VALUE шага.
- 4.Переходит на инструкции после END состояния.

Если нет никакого значения, которое соответствует индексной переменной, выполняются программные инструкции после ANY состояния. Если нет состояния ANY, ни один из шагов в CASE структуре не выполняется.

(ПРИМЕЧАНИЕ)

ANY состояние может быть пропущено

ANY состояние может быть использовано только единожды в структуре, и должно заканчивать структуру.

Знак «:» после ANY состояния может быть пропущен. Когда знак вводится, всегда отделяйте его пробелом. Без пробела ANY: считается меткой.

И ANY и AND состояния должны вводиться по одному в линии

Рассмотрим следующий пример, если значение реальной переменной x отрицательно, программное выполнение останавливается после вывода сообщения на экран. Если значение положительно, программа обрабатывает соответственно три случая

1. Если число четное в пределах от 0 до 10
2. Если число нечетное в пределах от 1 до 9
3. Если число больше 10

IF $x < 0$ GOTO 10

CASE x OF

VALUE 0,2,4,6,8,10:

PRINT "The number x is EVEN"

VALUE 1,3,5,7,9:

PRINT "The number x is ODD"

ANY :

PRINT "The number x is larger than 10"

END

STOP

10 PRINT "Stopping because of negative value"
STOP

6.7. ПРОГРАММНЫЕ ИНСТРУКЦИИ ДВОИЧНЫХ СИГНАЛОВ

RESET	Выключает все внешние сигналы вывода.
SIGNAL внутренних сигналов.	Включает ON/OFF внешних сигналов ввода - вывода и
PULSE DLYSIG прошло.	Включает сигнал выхода для указанного времени. Устанавливает сигнал после того, как указанное время
RUNMASK	Задаёт маску сигналов.
BITS значению.	Устанавливают группу сигналов, равным указанному
SWAIT состояние не установится	Выполнение программы останавливается, пока указанное
EXTCALL	Вызывает программу, выбранную внешним сигналом.
ON	Устанавливает условия прерывания.
ONI	Устанавливает условия прерывания .
IGNORE	Отменяет ON или команду ONI.
SCNT	Выводит сигнал счётчика после прочтения заданного значения (опция)
SCNRESET	Очищает номер сигнала счётчика. (Опция)
SFLK	Устанавливает ON/OFF сигнала мерцания. (Опция)
SFLP	Устанавливает ON/OFF сигналов с SET/RESET сигналов (опция)
SOUT	Выводит сигнал, когда встречается определенное состояние. (опция)
STIM	Включает сигнал таймера, когда заданный сигнал становится ON (опция)
SETPICK	Устанавливает время для начала управления закрытием схвата. (опция)
SETPLACE	Устанавливает время для начала управления открытием схвата. (опция)
CLAMP	Управляет сигналами открытия/закрытия фиксаторов.

RESET

Назначение

Выключает все внешние выходные сигналы. Эта команда неэффективна для приоритетных сигналов, сигналов зажимных устройств и парадоксальных многофункциональных OX/WX.

При использовании дополнительной установки, сигналы, задействованные для экрана интерфейсной панели не затрагиваются этой инструкцией. (опция)

SIGNAL signal number,

Назначение

Включает или выключает заданные внешние выходные или внутренние сигналы.

Параметр

Сигнальный номер

Выбирает номер внешнего выходного или внутреннего сигнала. Выборка приоритетного сигнала приведет к ошибке.

Внешние выходные сигнальные номера – 1-32

Внутренние сигнальные номера – 2001-2256

Является мониторной командой. См. 5.7

PULSE signal number, time

Назначение

Включает заданный внешний выходной или внутренний сигнал на данный период времени.

Параметр

1. Сигнальный номер

Задаёт номер внешнего выходного или внутреннего сигнала. Задание приоритетного сигнала приведет к ошибке.

Внешние выходные сигнальные номера – 1-32

Внутренние сигнальные номера – 2001-2256

2. Время

Устанавливает, как долго сигнал включен (в секундах). Если параметр не задан, автоматически устанавливает 0.2 секунды.

Является мониторной командой. См. 5.7

DLYSIG signal number, time

Назначение

Выводит определенный сигнал после истечения заданного промежутка времени.

Параметры

1. Сигнальный номер

Задаёт номер внешнего выходного или внутреннего сигнала. Если сигнальный номер положительный, сигнал включается, если отрицательный – выключается. Задание приоритетного сигнала приведёт к ошибке.

Внешние выходные сигнальные номера – 1-32

Внутренние сигнальные номера – 2001-2256

2. Время

Задаёт время для задержки вывода сигнала в секундах.

Является мониторной командой. См. 5.7

RUNMASK starting signal number, number of signal

Назначение

Следующие сигналы будут включены только при выполнении программы. Сигналы могут быть включены, используя SIGNAL, PULSE, DLYSIG инструкции, но они выключаются, когда происходит останов программного выполнения. При продолжении выполнения программы состояние каналов выхода восстанавливается таким, каким оно было до останова.

Параметры

1. Начальный стартовый номер

Задаёт номер первого внешнего выходного или внутреннего сигнала в группе сигналов для наложения маски. Введение отрицательного числа отменяет маску, и сигнал не будет отключаться при программном останове.

Внешние выходные сигнальные номера – 1-32

Внутренние сигнальные номера – 2001-2256

2. Число сигналов

Задаёт, как много сигналов входит в маску. Если параметр не задан, устанавливается 1.

Для приоритетных сигналов команда неэффективна.

Пояснение

Сигналы, заданные этой инструкцией, устанавливаются в состояние OFF, когда программное выполнение останавливается. Однако для приоритетных сигналов данная инструкция не эффективна. Когда программа продолжается, при использовании команды CONTINUE, сигналы восстанавливают состояние, которое было до останова. Аналогично происходит с DO или STEP командами. Рестарт программы посредством EXECUTE команды аннулирует RUNMASK инструкцию.

Пример

RUNMASK 5,2

Накладывает маску на 5 и 6 выходные сигналы, заданных при помощи 2 бит. Во время выполнения программы эти сигналы могут быть включены при помощи инструкции SIGNAL, PULSE, DLSIG. Обнуление сигналов происходит при останове программы.

BITS starting signal number, number of signal=decimal value

Назначение

Классифицирует группу внешних выходных или внутренних сигналов в соответствии с двоичным выражением десятичного числа.

Параметры

1. Начальный сигнальный номер

Выбирает первый сигнал для установки сигнального состояния.

Внешние выходные сигнальные номера – 1-32

Внутренние сигнальные номера – 2001-2256

2. Количество сигналов

Задаёт количество сигналов для установки состояний ON/OFF. Максимальное разрешенное количество 16.

3. Десятичное значение

Выбирает десятичное значение, в соответствии с которым устанавливается состояние ON/OFF выбранной группы сигналов. Десятичное число переводится в двоичное выражение и каждый бит двоичного числа устанавливает сигнальное состояние, начиная с самого младшего двоичного разряда.

Является мониторной командой. См. 5.7

SWAIT signal number,

Назначение

Ожидает пока внешний I/O или внутренний сигнал приобретет заданное состояние.

Параметр

Сигнальный номер

Задаёт сигнальный номер для контроля состояния. Отрицательные числа указывают, что состояние удовлетворяется, когда сигналы выключены.

Внешние выходные сигнальные номера – 1-32

Внешние входные сигнальные номера – 1001 - 1032

Внутренние сигнальные номера – 2001-2256

SWAIT инструкция может быть прервана командой CONTINUE NEXT.

Тот же самый результат может быть получен, используя инструкцию WAIT.

Пример

SWAIT 1001,1002

SWAIT 1,-2001

Ожидает до тех пор, пока входные сигналы 1001 и 1002 не включатся.

Ожидает до тех пор, пока внешний выходной сигнал 1 не включится, внутренний сигнал 2001 не выключится

EXTCALL

Назначение

Вызывает программу, выбранную при помощи внешнего входного сигнала.

Пояснение

EXTCALL инструкция обработана в следующих процедурах:

1. Выводится JUMP-ST сигнал, позволяющий вход во внешнюю программу.
2. Ожидает для JUMP-ON сигнала ввода.
3. Когда JUMP-ON вводится, программный номер вводится при помощи считывания RPS-CODE. Если число есть 100 или выше, вызывается программа pgxxx, если число есть 99-10, вызывается программа pgxx, если число меньше равно 9, вызывается программа pg9.

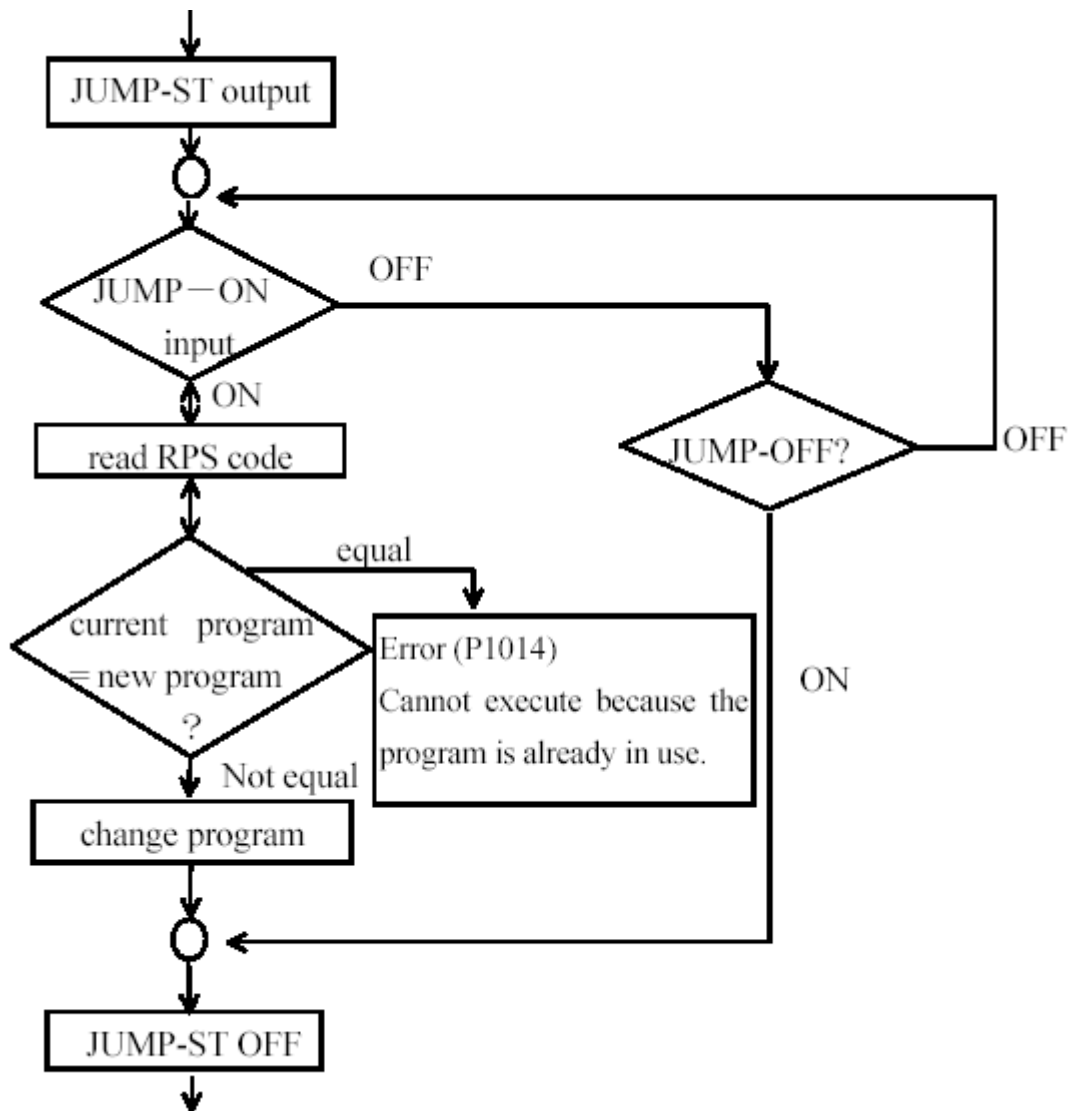
Примечание

Эта инструкция может быть пропущена введением CONTINUE NEXT командой, в процессе ожидания JUMP-ON сигнала.

Эта инструкция эффективна только при активизированном RPS режиме и установленном приоритетным сигналом RPS сигнала. Произойдет ошибка, если эта инструкция выполняется, а сигнал RPS не приоритетный

Если RPS режим запрещен, инструкция игнорируется.

EXTCALL используется, чтобы вызвать подпрограмму. После завершения этой подпрограммы (или когда в подпрограмме была обработана инструкция RETURN), выполнение возвращается в первоначальную программу.



ON mode signal number CALL program name, priority
ON mode signal number GOTO label, priority
ONI mode signal number CALL program name, priority
ONI mode signal number GOTO label, priority

Назначение

Контролирует определенный внешний входной или внутренний сигнал и при появлении сигнала переходит к выполнению подпрограммы, или перепрыгивает на заданную метку.

ONI прерывает текущую инструкцию движения, ON ожидает завершения текущей инструкции движения, только после этого переходит к подпрограмме или метке.

Параметры

1.Режим

/ERR – выдает ошибку, если состояние сигнала уже установлено до начала контроля

/LVL – немедленно переходит к подпрограмме или метке, если состояние сигнала уже установлено до начала контроля.

2. Сигнальный номер

Задаёт номер сигнала для контроля.

Если номер положительный, контролируется нарастающий фронт сигнала или изменение с OFF на ON.

Если номер отрицательный, контролируется затухающий фронт сигнала или изменение с ON на OFF.

Внешние входные сигнальные номера – 1001-1032

Внутренние сигнальные номера – 2001-2256

3. Программное имя

Задаёт имя подпрограммы для выполнения при получении заданного сигнала. Если параметр пропущен, программа переходит на следующий шаг и не переходит к подпрограмме.

4. Метка

Задаёт метку для перехода.

5. Приоритет

Выбирает приоритет программы, установленные рамки 1-127. Если параметр пропущен, устанавливается 1. Приоритет игнорируется, когда происходит переход к метке.

Пояснение

Если изменение обнаруживается в контролируемом сигнале, ON...CALL инструкция прерывает выполнение программы и переходит на выполнение подпрограммы. Если RETURN инструкция выполняется в вызванной подпрограмме, происходит возврат в основную программу на шаг, следующий за шагом, выполнявшимся в момент прерывания.

ONI инструкция может быть использована только в программах движения робота, но не может использоваться в РС программах.

Контролирование заданного сигнала отменяется в следующих случаях:

1. IGNORE инструкция, выполняемая для сигнала, заданного в ON или ONI инструкциях.
2. ON, ONI выполнены и произошёл переход в подпрограмму.
3. Новые инструкции задают тот же сигнал, что и более ранние инструкции.

(ПРИМЕЧАНИЕ)

1. Когда происходит контролирование нарастающего или затухающего фронтов сигнала, переход в подпрограмму происходит при изменении сигнального состояния. Поэтому, если верхний фронт должен быть обнаружен, переход не происходит, если этот сигнал уже включен, когда ON инструкция выполнена. Никакой переход не произойдет, пока сигнал не выключен и не включен снова.

2. Чтобы обнаружить изменение сигнала достаточно точно, необходима устойчивость сигнала в течение 50 мсек.

3. Контролирование начинается, как только инструкция ON (ONI) выполнена. Т.к. в AS системе, инструкции, не выполняющие движение, читаются и выполняются вместе с предыдущим движением, контролирование начинается в одно и тоже время с движением до выполнения инструкции ON (ONI).

4. Контроль сигналов не осуществляется, если программа не выполняется.

Пример

ONI -1001 CALL alarm

Контролирует внешний входной сигнал 1001. Как только этот сигнал изменяется от ON к OFF, движение останавливается и программа переходит к выполнению подпрограммы “alarm”.

ON test CALL delay

Контролирует сигнал, установленный в переменной “test”. Если сигнал изменяется как желательно, программа переходит к подпрограмме “delay” после выполнения текущего шага движения полностью. Возврат в головную программу осуществляется после полного выполнения подпрограммы “delay”.

IGNORE signal number

Назначение

Отменяет контролирование сигнала, заданного в ON (ONI) инструкциях

Параметр

Сигнальный номер

Выбирает номер сигнала для отмены контроля.

Внешние входные сигнальные номера – 1001-1032

Внутренние сигнальные номера – 2001-2256

Пояснение

Эта инструкция аннулирует эффект последней ON или ONI инструкции.

(ПРИМЕЧАНИЕ)

ON(ONI) контролирующая функция является эффективной только двоичных сигналов, реально установленных как входные сигналы.

Пример

IGNORE 1005

IGNORE test

Отменяет контролирование внешнего входного сигнала 1005

Отменяет контролирование сигнала, заданного переменной “test”.

SCNT counter signal number=count up signal, count down signal, counter clear signal, counter value

Назначение

Выводит сигнал счетчика, когда указанное значение счетчика достигнуто.

Параметры

Назначение

Выводит сигнал счетчика, когда заданное значение регистра достигнуто.

Параметры

1.Номер сигнала счетчика

Задается номером сигнала для вывода. Допустимые значения 3097-3128.

2.Сигнал для счета в прямом направлении

Задается сигнальным номером или логическим выражением. Всякий раз, когда сигнал изменяется от OFF к ON, счетчик подсчета в прямом направлении увеличивается на единицу.

3.Сигнал обратного счета

Задается сигнальным номером или логическим выражением. Всякий раз, когда сигнал изменяется от OFF к ON, счетчик подсчета в обратном направлении уменьшается на единицу.

4.Счетчик сигнала отбоя

Задается сигнальным номером или логическим выражением. Если этот сигнал становится ON, внутренний счетчик устанавливается в ноль.

5.Значение счетчика

Когда внутренний счетчик достигает этого значения, заданный сигнал выводится. Если дан «0» сигнал выключается.

Пояснение

Если сигнал счета в прямом направлении изменяется от OFF в ON, когда выполняется команда SCNT, тогда значение внутреннего счетчика увеличивается на единицу 1. Если сигнал обратного счета изменяется от OFF в ON, когда выполняется команда SCNT, тогда значение внутреннего счетчика уменьшается на единицу 1. Когда значение внутреннего счетчика достигнет значения, заданного в параметре (значение счетчика), выводится сигнал счетчика. Если выводится сигнал очистки счетчика, значение внутреннего счетчика становится 0. Каждый сигнал счетчика имеет свое индивидуальное значение счетчика. Для того чтобы принудительно сбросить внутренний счетчик в 0, используйте команду SCNTRESET.

Для того чтобы проверить состояние сигналов 3001 - 3128, используйте команду IO/E .
(опция)

SCNRESET counter signal number

Назначение

Возвращает в ноль значение внутреннего счетчика в соответствии с сигнальным номером счетчика.

Параметр

Номер сигнала счетчика

Задаёт номер сигнала счетчика для сброса. Установочный диапазон для номеров сигнала счетчика: 3097 - 3128.

SFLK signal number=time

Назначение

Устанавливает ON/OFF (мерцание) заданного сигнала в заданном цикле времени.

Параметры

1.Номер сигнала

Задаёт номер сигнала. Допустимые значения 3065-3096.

2.Время

Задаёт время цикла ON/OFF (реальное значение). Если значение отрицательное, мерцание прекращается.

Пояснение

Процесс ON/ OFF рассматривается в одном цикле, и цикл выполняется в определенное время.

SFLP output signal=set signal expression, reset signal expression

Назначение

Устанавливает ON/OFF выходного сигнала, используя установленный сигнал и сигнал сброса.

Параметры

1.Выходной сигнал

Задаёт номер выходного сигнала. Положительное число включает сигнал, отрицательное число выключает сигнал. Могут быть заданы только номера выходных сигналов.

2.выражение сигнала установки

Задаётся сигнальным номером или логическим выражением для установки выходного сигнала.

3. Выражение сигнала сброса

Задается сигнальным номером или логическим выражением для переустановки выходного сигнала.

Пояснение

Если установленный сигнал включен, выходной сигнал включается. Если переустановленный сигнал включен, выходной сигнал выключается. Выходной сигнал становится ON или OFF, когда выполняется команда SFLP, и нет, когда сигнал установки или сигнал сброса в ON.

SOUT signal number=signal expression

Назначение

Выводит заданный сигнал, когда устанавливается определенное состояние.

Параметры

1.Номер сигнала

Задаёт номер выходного сигнала

2.Сигнальное выражение

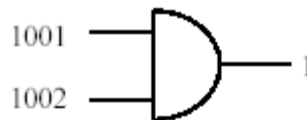
Задаёт номер сигнала или логического выражения.

Пояснение

Это команда – для логического вычисления сигналов. Используются логические выражения такие, как AND или OR. Определенный сигнал выводится, когда это состояние установится.

Example

SOUT 1 = 1001 AND 1002



SOUT 1 = 1001 OR 1002

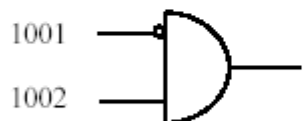


SOUT -1 = 1001 AND 1002

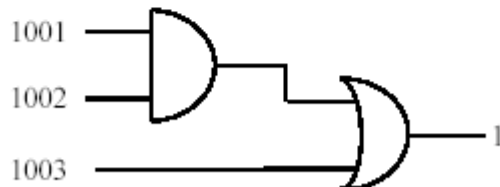
SOUT 1 = NOT(1001 AND 1002)



SOUT 1 = -1001 AND 1002



SOUT 1 = (1001 AND 1002) OR 1003



SOUT -1 = 1001 or SOUT 1 = -1001 or
SOUT 1 = NOT(1001)



STIM timer signal=input signal number, time

Назначение

Включает сигнал таймера, если указанный входной сигнал включен в течение заданного времени.

Параметры

1. Сигнал таймера

Выбирает сигнал для включения. Допустимые сигнальные номера 3001-3064.

2. Номер входного сигнала

Определяет в целых числах номер входного сигнала или логического выражения, чтобы контролировать условие для включения сигнала таймера. Значение не может превысить номера сигналов фактически установленных.

3. Время

Задаёт в действительных числах время (сек), в течение которого входной сигнал должен быть включен до включения сигнала таймера.

Пояснение

Контролируемый входной сигнал должен поступать непрерывно для того, чтобы сигнал таймера включился. Если входной сигнал пропадает до истечения заданного промежутка времени, происходит рестарт счета времени при появлении входного сигнала. Если входной сигнал не появился в данном промежутке времени, сигнал таймера не включается. Однако входной сигнал оказывает влияние на сигнал таймера, только в момент выполнения STIM инструкции.

Чтобы проверить состояние сигналов 3001-3128 используйте команду IO/E.

SETPICK time1, time2, time3, time4, ..., time8

SETPLACE time1, time2, time3, time4, ..., time8

Назначение

Устанавливает время для начала управления закрытием схвата (SETPICK) или открытием схвата (SETPLACE) для каждого из 8 фиксаторов.

Параметр

Время 1 - 8

Устанавливает управление временем для открытия/закрытия фиксаторов 1 - 8 в секундах.

Пояснение

См. CLAMP инструкцию для подробностей.

CLAMP clamp number 1,....., clamp number 8

Назначение

Выводит сигнал фиксатора для открытия/закрытия схвата, заданного параметром фиксатора n. Выбор времени устанавливается инструкциями SETPICK/SETPLACE, т.е. сигнал выводится x секунд до завершения полностью текущего движения.

Параметр

Номер фиксатора 1 – 8

Задаёт номер фиксатора. Если число положительное, схват открывается, если число отрицательное, схват закрывается.

Пояснение

Эта инструкция выдает сигналы для контроля открытия/закрытия клапанов пневматического схвата. Сигнал выводится немедленно, если робот не движется, или если остающееся время движения меньше времени, заданного SETPICK/SETPLACE инструкциями. Сигнал выводится, когда оси совпадают, если совмещение следующего движения начинается раньше времени, установленного SETPICK/SETPLACE инструкциями. Если сделана нелогичная установка такая, как “CLAMP 1, -1”, последний номер схвата будет действительным.

Пример

12 SETPICK 4, 3, 2, 1

13 SETPLACE 0.2, 0.4, 0.6, 0.8

14 LMOVE a

15 CLAMP -1, 2, 3, -4

Закрывает схват 2, в течение 3 секунд до достижения позиции a

Закрывает схват 3, в течение 2 секунд до достижения позиции a

Открывает схват 4, в течение 0.8 секунд до достижения позиции a

Открывает схват 1, в течение 0.2 секунд до достижения позиции a

6.8. ПРОГРАММНЫЕ ИНСТРУКЦИИ СООБЩЕНИЙ

PRINT	Отображает сообщение на терминале.
TYPE	Отображает сообщение на терминале.
PROMPT	Отображает сообщение на терминале и ожидает ввода с клавиатуры.
IFPWPRINT	Отображает заданную символьную строку в окне отображения.

PRINT device number: print data,....
TYPE device number: print data,....

Назначение

Отображает на терминале данные печати, заданные в параметре.

Параметры

1.Номер внешнего устройства

Выбирает номер внешнего устройства для отображения данных

1: персональный компьютер

2: пульт ручного управления

Если параметр не задан, данные отображаются на подключенном, в данный момент, устройстве.

2.Данные печати

Устанавливает данные для печати. Разделите данные запятой, когда задаются несколько параметров.

(1) цепочка символов: т.е. "count="

(2) реальное выражение значения: т.е. count

(3) информация о формате: т.е. /D, /S

Если параметры не заданы, отображается пустая строка.

Следующие коды используются для задания выводного формата числового выражения. Пока не задан другой формат, используется одинаковый формат. В любом формате, если значение является слишком большим для изображения, звездочки заполняют все пространство. В этом случае необходимо ввести число символов, которые могут быть отображены. Максимальное число символов, помещающихся в одной строке 128. Отобразить более чем 128 символов в одной строке возможно, используя /S код.

(Примечание)

Если MEASSAGES переключатель выключен, сообщение не появляется.

Формат используемых кодов

/D

Использует заданный по умолчанию формат. Это тоже самое как формат /G15.8 за исключением того, что нули, следующие после числовых значений и пробелов, удалены.

/Em.n

Отображает числовое значение в экспоненциальном виде (в виде мантиссы и порядка, т.е. -1.234+02). "m" описывает общее число символов, отображаемых на экране, "n" – число десятичных разрядов. "m" должно быть больше чем "n", больше 6 и меньше 32.

/Fm.n

Отображает числовое значение с фиксированной запятой (-1.234). "m" описывает общее число символов, "n" – число знаков в дробной части.

/Gm.n

Если значение больше чем 0.01 и может быть отображено в Fm.n формате в пределах m цифр, значение отображается в том формате. В других случаях – в Em.n формате.

/Hn

Отображает значение как шестнадцатиричное число в n цифровом поле

/In

Отображает значение как десятичное число в n цифровом поле

Следующие параметры используются, чтобы вставить определенные символы внутри цепочки символов.

/Cn Вставка перевода строки n раз в месте, где этот код введен, либо спереди, либо после данных печати. Если этот код расположен внутри данных печати, n пустых строк вставятся.

/S Строка не продолжается

/Xn Вставка n пробелов

IFPWPRINT window, row, column, background color, label color= “character string”,....

Назначение

Отображает заданную символьную цепочку в строке окна, установленного при помощи дополнительной функции 0509

Параметры

1.Окно

Выбирает окно, в котором отобразится последовательность символов. Выборка от 1 до 4 в дополнительной функции 0509.

2.Строка

Определяется строка в окне для отображения цепочки. Допустимое число от 1 до 16, однако, это зависит от размеров окна. Если не задано, подразумевается 1.

3.Колонка (столбец)

Определяется столбец в окне для отображения цепочки. Допустимое число от 1 до 43, однако, это зависит от размеров окна. Если не задано, подразумевается 1.

4.Цвет заднего фона

Определяется цвет заднего фона выбранного окна. Допустимое число от 1 до 15. Если не задано, задний фон белый.

5.Цвет символа

Определяется цвет символов. Допустимое число от 1 до 15. Если не задано, цвет черный.

6.Символьная строка

Задаёт цепочку символов для отображения. Все строки после первой строки отображаются в следующей строке, начиная с заданного столбца. Выполнение данной

команды очищает все параграфы, за исключением заданной символьной цепочки, из заданного окна.

PROMPT device number: character string, variable

Назначение

Отображает заданную символьную цепочку на терминале, сопровождаемую подсказкой “>”, и ожидает ввода от клавиатуры.

Параметры

1.Номер устройства

Выбирает устройство для отображения данных

1: Персональный компьютер

2: Пульт ручного управления

Если не задан, данные будут отображаться на текущем устройстве

2.Символьная цепочка

Задаёт символы для отображения на терминале.

3.Переменные

Задаёт на какие переменные изменяются данные, вводимые с клавиатуры. Это может быть последовательность реальных переменных или единичная строковая переменная.

Пояснение

Заданные символьные строки отображаются на терминале и ожидается ввод данных и нажатие **ENTER**, чтобы быть введенными с клавиатуры.

Данные ввода используются в одном из следующих случаев

1. Когда PROMPT используется, чтобы узнать о значениях для ряда переменных, система читает вводимую строку как ряд чисел, разделенных пробелами или запятыми. Каждое вводимое число преобразуется во внутренние выражения в соответствии с системой счисления, затем они назначаются переменными один за другим.
2. Если число величин ввода больше числа переменных, лишние величины игнорируются. Если число величин ввода меньше числа переменных, оставшимся переменным присваивается 0. Если данные ввода отличны от числовых значений, происходит ошибочная ситуация и программное выполнение останавливается. Чтобы избежать путаницы и ошибки, желательно использовать эту инструкцию для присвоения одного значения одной переменной.

Когда используют символ строчной переменной как переменный параметр для PROMPT, символы вводятся и читаются, как отдельная единица данных и все символы присваиваются символу строчной переменной.

В экране подсказки, если только клавиша **ENTER** или **CTRL** + **C** нажаты, “0” присваивается в реальные переменные, и в случае символьной строковой переменной, присваивается пустая строка.

Если “2” введен для номера устройства, экран пульта ручного управления автоматически переходит в экран клавиатуры. Нажмите **NEXT PAGE**, чтобы возвратиться к обычному экрану.

Пример

Символьная строка в кавычках отображается на терминале, и ожидает ввода данных. Когда данные (номер деталей) - введен, и клавиша **ENTER** нажата, введенное значение Присваивается в переменную “part.count”. Выполнение программы продолжается.

PROMPT "Enter the number of parts: ", part.count

Инструкция ниже ожидает ввода значений символьной строковой переменной, разнообразие алфавитно-цифровых символов могут быть вводом, не вызывая ошибку.

PROMPT "Enter the number of parts: ", \$input

6.9. ИНСТРУКЦИИ ПОЗИЦИОННОЙ ИНФОРМАЦИИ

HERE	Задаёт текущую позицию как позиционную переменную.
POINT	Задаёт позиционную переменную.
POINT/X	Устанавливает X значение позиционной переменной.
POINT/Y	Устанавливает Y значение позиционной переменной.
POINT/Z	Устанавливает Z значение позиционной переменной
POINT/OAT	Устанавливает OAT значения позиционной переменной
POINT/O	Устанавливает O значение позиционной переменной
POINT/A	Устанавливает A значение позиционной переменной
POINT/T	Устанавливает T значение позиционной переменной
POINT/7 переменной	Устанавливает значение седьмой координаты позиционной переменной
DECOMPOSE	Присваивает компоненты позиционной переменной в элементы массива переменных.
TOOL	Задаёт положение TCP.
BASE	Изменяет систему базовых координат робота.
LLIMIT	Устанавливает верхний предел для движения робота.
ULIMIT	Устанавливает нижний предел для движения робота.
TIMER	Устанавливает таймер.
UTIMER	Устанавливает таймерное значение пользовательского таймера.
ON	Устанавливает состояние ON системного переключателя
OFF	Устанавливает состояние OFF системного переключателя.
NCHON	Устанавливает состояние ON режекторного фильтра (опция)
NCHOFF	Устанавливает состояние OFF режекторного фильтра (опция)
WEIGHT	Устанавливает данные весовой нагрузки.
MC	Выполняет мониторинговую команду из РС программ.

PLCAOUT	Устанавливает реальные значения в выходные данные(опция)
TPLIGHT	Устанавливает подсветку пульта ручного управления.

HERE pose variable name

Назначение

Задаёт имя переменной координаты в текущей позиции. Позиция может задаваться как декартовая, угловая, сложная координата.

Параметр

Имя координаты точки

Присваивает координате обученной точки имя. Позиция может задаваться как декартовая, угловая, сложная координата.

(ПРИМЕЧАНИЕ)

Только крайне правая переменная сложной координаты задается. Если другие переменные, используемые в сложной координате, не заданы, произойдет ошибка при выполнении этой команды.

См. мониторинговую команду 5.5

POINT pose variable name1=pose variable name2, joint displacement values

Назначение

Присваивает координату точки, заданной в правой части равенства координате точки, заданной в левой части равенства.

Параметры

1. Определяет имя координаты, которую необходимо задать (декартовую, угловую, сложную)

2. Задаёт имя реально существующей координаты. Если параметр не задан, на дисплее появляются координаты точки pose variable name с запросом на изменение “Change?”. При этом можно изменить координату точки, поставив на соответствующие координаты необходимые значения. Выполнение команды и ее закрытие происходит путем нажатия клавиши (Enter).

3. Этот параметр должен быть установлен, когда в левой части равенства координата задается, как угловая, а первая координата правой части равенства, как декартовая. (если параметр слева не является угловой координатой, этот параметр не может быть установлен). Угловая координата правой части равенства определяет конфигурацию робота в пространстве в новой позиции. Если параметр, не задан текущая конфигурация будет использоваться, для того чтобы определить переменную позиции

Пояснение

Ошибка получится, если переменная позиции с правой стороны “=” не определена. Только крайне правая переменная сложной координаты задается. Если другие переменные, используемые в сложной координате, не заданы, произойдет ошибка при выполнении этой команды.

Примеры

>POINT #park

JT1	JT2	JT3	JT4	JT5	JT6
10.000	15.000	20.000	0.000	30.000	90.000

Change? (if not, hit RETURN only) (Enter)
,,,35

JT1	JT2	JT3	JT4	JT5	JT6
10.000	15.000	20.000	35.000	30.000	90.000

Change? (if not, hit RETURN only) (Enter)

>POINT pick1=pick (Enter)

присваивает декартовой координате pick1 значение декартовой координаты pick

>POINT pos0=#pos0 (Enter)

преобразует угловую координату #pos0 в декартовую и присваивает декартовой координате pos0 данное значение

>POINT #pos1=pos1,#pos2 (Enter)

преобразует декартовую координату “pos1” в угловую, используя конфигурацию робота, определенную позицией #pos2, присваивает это значение точке “#pos1”

Является мониторной командой см. 5.5

POINT/X transformation variable name1=transformation values2
POINT/Y transformation variable name1=transformation values2
POINT/Z transformation variable name1=transformation values2
POINT/OAT transformation variable name1=transformation values2
POINT/O transformation variable name1=transformation values2
POINT/A transformation variable name1=transformation values2
POINT/T transformation variable name1=transformation values2
POINT/7 transformation variable name1=transformation values2

Назначение

Присваивает элементы декартовой координаты, заданной в правой части равенства соответствующим элементам декартовой координаты, определенной в левой части равенства. Эти значения будут отображены на дисплее для редактирования.

Параметры

Имя декартовой координаты 1

Выбирает имя декартовой координаты для изменения координатного значения.

Имя декартовой координаты 2

Выбирает имя декартовой координаты, из которой будут взяты необходимые данные для изменения. Точка должна быть заранее обучена, она может быть простой, сложной точкой или функцией декартовой точки.

Если задана сложная точка в левой части равенства, эта инструкция определит только крайнее правое значение в этой сложной точке.

Пояснение

Ошибка получится, если переменная позиции с правой стороны “=” не определена.

Если сложная координата задана в левой части равенства, эта инструкция задает только крайнее правое значение сложной координаты. Если другие переменные, используемые в сложной координате, не заданы, произойдет ошибка при выполнении этой команды.

Пример

```
POINT/X temp=tempx
```

DECOMPOSE array variable name[element number]=pose variable name

Назначение

Записывает как элементы массива каждый компонент координаты заданной позиции (X,Y,Z,O,A,T, или JT1,JT2,JT3,JT4,JT5,JT6)

Параметр

1.Имя массива переменных

Задаёт имя массива переменных, в котором значения каждого компонента позиции будет записано.

2.Номер элемента

Задаёт первый элемент, с которого записываются компоненты.

3.Имя позиции (координаты)

Задаёт имя позиции, для которой выделяется каждый компонент.

Пояснение

Происходит присвоение элементов выбранной позиции элементам массива переменных.

При задании декартовой координаты шесть элементов координаты присваиваются шести элементам массива. При задании угловой координаты каждому элементу присваивается угловое значение соответствующих суставов робота.

Пример

```
DECOMPOSE X[0]=part
```

Присваивает координатные значения точки “part” первым шести элементам массива X.

```
DECOMPOSE angles[4]=#pick
```

Присваивает угловые значения точки #pick элементам массива angles, начиная с четвертого элемента и т.д.

Для примера, если значение #pick (10, 20, 30, 40, 50, 60), то массив будет

angle[4]=10	angle[7]=40
angle[5]=20	angle[8]=50
angle[6]=30	angle[9]=60

BASE transformation values

Назначение

Задаёт сдвиг начала координат (0,0,0,0,0,0) декартовой системы координат на определённые величины (X,Y,Z,O,A,T), заданные в параметре, создавая тем самым новую относительную систему координат.

Параметр

Имя точки (точка декартовая, сложная)

Определяет новую базовую систему координат. Точка здесь описывает положение начала координат новой системы по отношению к началу координат нулевой базовой системы координат. Если параметр не определен, на дисплее появляется информация об установленной системе координат в данный момент времени.

Пояснение

Команда BASE используется, как правило, когда есть неоднократно повторяющиеся траектории движения. Задаётся траектория движения в нулевой системе координат, затем, задав нужную базовую систему, без обучения точек, произойдет повторение траектории движения.

Следует учитывать, что координата точки (параметр) не может быть создана, путем обучения и запоминая, используя команду HERE. Координата точки создается аналитически (вычисляется) и вводится в память компьютера при помощи команды POINT.

Является мониторной командой

TOOL transformation values

Назначение

Определяет новую инструментальную систему координат, которая задается как сдвиг нулевой инструментальной системы на определенные расстояния, выраженные в координатах X,Y,Z,O,A,T

Параметр

Имя точки (точка декартовая, сложная)

Определяет новую инструментальную систему координат. Параметр здесь описывает положение начала координат новой системы по отношению к началу координат нулевой инструментальной системы координат. Если параметр не определен, на дисплее появляется информация об установленной инструментальной системе координат в данный момент времени.

Пояснение

Инструментальная система координат с началом 0,0,0,0,0 находится в центре фланца робота (6-ой сустав), плоскость XOY совпадает с плоскостью фланца, ось Z параллельна оси фланца.

Если установлена нулевая инструментальная система координат, вращение вокруг осей оставляет неподвижной точку начала инструментальной системы координат, но при наличии инструмента (все технологические операции требуют наличие какого-нибудь инструмента), конец инструмента перемещается в пространстве. При обучении и дальнейшем прохождении сложных траекторий, особенно, где требуется изменение ориентации на небольших расстояниях по ходу движения, могут возникнуть осложнения такие, как неравномерное движение от точки к точке, невозможность пройти по прямой траектории, во время сложного перехода возможно столкновение с обрабатываемой поверхностью. Чтобы избежать данных ситуаций используется команда TOOL. Начало координат переносится на конец инструмента и при вращении неподвижной остается рабочая часть инструмента.

Является мониторной командой

ULIMIT	joint displacement values
LLIMIT	joint displacement values

Назначение

Устанавливает и отображает верхний/нижний пределы границ движения робота.

Параметр

Имя точки в угловых координатах

Устанавливает программно новые пределы (верхний/нижний) границ движения робота в угловых координатах. Если параметр не задан, отображается текущее положение робота, которое можно изменить или не менять в зависимости от ответа на вопрос "Change?".

Пояснение

Если параметр задан, на дисплее появляются угловые координаты точки, которые можно изменить после вопроса "Change?", на необходимые пользователю.

Является мониторной инструкцией

TIMER timer number=time

Назначение

Задает время определенного таймера.

Параметры

1.Номер таймера. Допустимые номера от 1 до 9.

2.Время

Задает время в секундах для заданного таймера.

Пояснение

Когда эта инструкция выполняется, таймер немедленно устанавливается на указанное время. Проверить значение таймера можно, используя функцию TIMER.

В примере, показанном ниже? программное выполнение приостанавливается на время, заданное инструкцией TIMER и функцией TIMER.

Пример

```
TIMER 1=0  
WAIT TIMER(1)>delay
```

UTIMER @timer variable=timer value

Назначение

Устанавливает значение по умолчанию для пользовательского таймера. Пользовательский таймер может быть назван произвольно, используя переменные таймера.

Параметры

[1.@Переменная](#) таймера

Задаёт имя переменной или массива переменных, используемое как имя таймера.

2.Значение таймера

Задаёт по умолчанию значение для таймера. Допустимые числа от 0 до 2147483647 (секунды).

Switch name,... ON Switch name,...OFF

Назначение

Задавая имя системного переключателя, устанавливается состояние включения/выключения в соответствии с ON/OFF

Параметр

Имя переключателя

Устанавливает состояние ON/OFF для переключателя, заданного здесь. При задании более чем одного имени, они разделяются запятой.

Текущая установка системного переключателя может быть проверена при помощи SWITCH команды.

Является мониторинговой командой.

NCHON NCHOFF

Назначение

Включает/выключает фильтр-пробку в шагах движения в соответствии заданным шагам. NCHON разрешает фильтр-пробку, NCHOFF запрещает.

Пример

```
10    SPEED 100 ALWAYS
20    HOME
30    DRIVE5,90
40    NCHOFF
50    SPEED 10
60    DRAW 1000
70    NCHON
80    DRAW,-500
```

} Th

(ПРИМЕЧАНИЕ)

Нормально, когда используется NCHON.

NCHOFF используется, чтобы устранить небольшие вибрации, возникающие при медленных скоростях, или когда внешние силы влияют на устойчивость движения робота. Фильтр сбрасывается с выполнением команды EXECUTE. Он также сбрасывается, когда впервые выполняются команды PRIME, STEP, MSTEP. По умолчанию установка ON (режекторный фильтр разрешен).

WEIGHT load mass , center of gravity location X, center of gravity location Y, center of gravity location Z, inertia moment ab. X axis, inertia moment ab. Y axis, inertia moment ab. Z axis

Назначение

Устанавливает данные массы груза (вес инструмента, детали). Данные используются для определения оптимального ускорения/замедления робота.

Параметры

1. Масса груза

Масса инструмента и детали (в килограммах). Допустимые пределы от 0 до максимальной загрузки, заданной в спецификации робота.

2. Расположение центра тяжести (в миллиметрах)

X – значение x центра тяжести в инструментальной системе координат

Y - значение y центра тяжести в инструментальной системе координат

Z - значение z центра тяжести в инструментальной системе координат

3. Момент инерции вокруг Оси X, момент инерции вокруг Оси Y, момент инерции вокруг Оси Z (Опция)

Устанавливает момент инерции вокруг каждой оси. Единица измерения - $\text{kg} \cdot \text{m}^2$. Момент инерции вокруг каждой оси определяется, как момент инерции вокруг координатных осей параллельных нулевой инструментальной системе координат с центром вращения в центре силы тяжести инструмента.

Пояснение

При использовании WEIGHT как программной инструкции, если параметры не заданы, по умолчанию устанавливаются максимальные параметры для данной модели робота.



DANGER

Всегда устанавливайте правильную массу груза и расположение центра тяжести. Неправильные данные могут ослабить или сократить долговечность частей робота или причинить перегрузку /отклонения от установленных норм, что приведет к нарушению нормальной работоспособности робота.

MC monitor command

Назначение

Разрешает выполнение мониторных команд из PC программ. Мониторные команды, которые могут быть использованы с этой инструкцией следующие:
ABORT, CONTINUE, ERESET, HOLD, SPEED.

Пояснение

Эта инструкция используется в случаях, когда программа робота выполняется (EXECUTE команда) из программы AUTOSTART.PC. MC инструкция не может задаваться в обычных программах.

Например

Программа робота может быть запущена из AUTOSTART.PC, используя эту инструкцию. Однако, для того чтобы выполнить программу робота, используя эту инструкцию, после включения силового питания привода, вводится шаг, проверяющие включение силового питания.

```
autostart.pc()
1 10 IF SWITCH(POWER)==FALSE GO TO 10
2 MC EXECUTE pg1
.END
```

PLCAOUT data number=real value (опция)

Назначение

Устанавливает данное действительное значение числа на указанное число данных.

Параметры

Число данных

Выбирает число выходных данных в целых числах. Допустимые числа 1 – 32.

Реальное значение

Выбирает значение, которое будет установлено в выходные данные, введенное в десятичном представлении или в виде переменной. Допустимые числа 0 –65535.

(ПРИМЕЧАНИЕ)

Эта функция действительна только тогда, когда опция “Built-in Sequencer Function” включена в состояние ON. Если опция в состоянии OFF, появится следующее сообщение (E1102) Cannot execute, no option set up. - Check option specs.

TPLIGHT

Назначение

Включает на пульте ручного управления лампу подсветки.

Пояснение

Если лампа подсветки на пульте ручного управления выключена, эта инструкция включает подсветку для следующих 600 секунд.

6.10. ИНСТРУКЦИИ УПРАВЛЕНИЯ ПРОГРАММОЙ И ДАННЫМИ

DELETE	Удаляет программы и переменные из памяти робота. (опция)
DELETE/P	Удаляет программы из памяти робота. (опция)
ELETE/L	Удаляет переменные позиций из памяти робота. (опция)
DELETE/R	Удаляет реальные переменные из памяти робота. (опция)
DELETE/S	Удаляет строковые переменные из памяти робота. (опция)
TRACE	Устанавливает ON/OFF функции TRACE.

DELETE program name,
DELETE/P program name,
DELETE/L pose variable,
DELETE/R real variable [array elements] ,
DELETE/S string variable [array elements] ,

Опция

Назначение

Удаляет выбранные данные из памяти

Параметры

(/P) – удаляет программы

(/L) – удаляет точки

(/R) – удаляет реальные переменные

(/S) – удаляет строчные переменные

Мониторная команда

TRACE **stepper number:** ON/OFF

Назначение

Регистрирует и фиксирует фактические данные с начала выполнения программы.

Параметры

Номер повторителя

Тип программы выбирается, используя следующие целые числа

1: Robot program

1001: PC program 1

1004: PC program 4

1002: PC program 2

1005: PC program5

1003: PC program 3

Если тип не выбран, регистрируются все программы

ON/OFF

Начинает/заканчивает фиксирование.

7.0 СИСТЕМНЫЕ ПЕРЕКЛЮЧАТЕЛИ

Эта глава описывает назначение каждого системного переключателя. Для установки ON/OFF или проверки состояния переключателей, обратитесь к командам/инструкциям SWITCH, ON и OFF.

CP	Разрешает, запрещает функцию непрерывности движения.
CHECK.HOLD <u>HOLD/RUN</u>	Разрешает, запрещает ввод команд с клавиатуры, когда в положении HOLD.
CYCLE.STOP	Останавливает цикл по внешнему (останову) HOLD.
MESSAGES	Разрешает или запрещает вывод на терминал.
OX.PREOUT	Устанавливает синхронизацию генерации выходного сигнала.
PREFETCH.SIGINS	Разрешает или запрещает более раннюю обработку I/O сигналов в AS программах.
QTOOL	Разрешает или запрещает преобразование инструментальной системы координат во время блочного программирования.
RPS	Разрешает или запрещает случайный выбор программ.
SCREEN	Управляет отображением терминала.
REP_ONCE	Устанавливает повторение цикла: один раз или непрерывно.
STP_ONCE	Устанавливает выполнение одно шага за один раз или непрерывно.
AUTOSTART.PC	Разрешает PC программе стартовать автоматически по включению силового питания (опция)
ERRSTART.PC	Определяет, если выбранная PC программа выполняется, когда произошла ошибка.
TRIGGER	Отображает ON/OFF состояние переключателя <u>TRIGGER</u> .
CS	Отображает ON/OFF состояние переключателя <u>CYCLE START</u> .
POWER	Отображает ON/OFF состояние переключателя <u>MOTOR POWER</u> .
RGSO	Отображает ON/OFF состояние сервоуправления.
TEACH_LOCK	Отображает ON/OFF состояние переключателя <u>TEACH LOCK</u> .
ERROR	Отображает, находится ли программа в состоянии ошибки или нет.

REPEAT	Отображает состояние переключателя TEACH/REPEAT .
RUN	Отображает состояние переключателя HOLD/RUN .
DISPIO_01	Изменяет режим изображения IO команд.
HOLD.STEP	Разрешает отображение выполняемого шага, когда программа была приостановлена (опция)
WS_COMPOFF	Изменяет синхронизацию выхода WS сигнала (опция)
FLOWRATE (и наоборот). (опция)	Меняет режим управления расходом на режим скорости выхода
WS.ZERO (опция)	Изменяет процесс сварки, который происходит когда WS=0.
ABS.SPEED	Разрешает использование абсолютной скорости (опция)
SLOW_START	Разрешает или запрещает функцию замедленного старта (опция)
AFTER.WAIT.TMR	Устанавливает как таймер запускается в блочном программировании (опция)

Следует различать аппаратные переключатели и системные переключатели. Аппаратные переключатели расположены на операционной панели контроллера и пульте ручного управления и в документации имеют вид/...... Системные переключатели входят в состав программного обеспечения и могут управлять функциями аппаратных переключателей.

CP

Назначение

Разрешает, запрещает функцию непрерывного движения (CP функцию)

Пояснение

Этот переключатель используется для включения или выключения CP функции. Если переключатель меняется в программе, выполнение CP функции разрешается, запрещается, начиная со следующего шага движения. По умолчанию установка для этого переключателя ON.

Пример

CP OFF Запрещает CP функцию

CHECK/HOLD

Назначение

Разрешает, запрещает использование клавиатуры для ввода команд EXECUTE, DO, STEP, MSTEP, CONTINUE, когда переключатель **HOLD/RUN** на операционной панели находится в положении HOLD.

Пояснение

Если этот переключатель включен (ON), следующие команды, вводимые с клавиатуры, воспринимаются только тогда, когда переключатель **HOLD/RUN** находится в положении HOLD. Но при этом следует учитывать, что движение может произойти только тогда, когда аппаратный переключатель будет в положении RUN.

EXECUTE, DO, STEP, MSTEP, CONTINUE

Если системный переключатель выключен (OFF), команды, показанные выше, воспринимаются независимо от положения аппаратного переключателя **HOLD/RUN**. Операции не введенные с клавиатуры не затрагиваются этой командой. (если **HOLD/RUN** переключатель в позиции RUN, робот начинает движение как только команда введена.)

По умолчанию состояние переключателя OFF.

CYCLE.START

Назначение

Определяет продолжить или нет автоматическое выполнение цикла после применения HOLD (т.е. при выполнении программы переключатель **HOLD/RUN** был установлен в положение HOLD).

Пояснение

Если этот переключатель выключен (OFF), робот останавливается, когда получает внешний входной сигнал HOLD, но **CYCLE.START** остается включенным.

Поэтому, когда внешний входной сигнал становится OFF, программное выполнение продолжается.

Если переключатель включен (ON), робот останавливается, когда получает внешний входной сигнал HOLD и `CYCLE.START` выключается. Поэтому, даже если внешний входной сигнал выключается (OFF), робот не может продолжить программное выполнение. Продолжить программное выполнение возможно, сделав следующие процедуры: включить кнопку `CYCLE.START`.

Этот останов имеет эффект только для внешнего входного сигнала HOLD и неэффективен для переключателя `HOLD/RUN` на операционной панели.

Если переключатель `HOLD/RUN` на операционной панели находится в положении HOLD, робот останавливается, но `CYCLE.START` остается включенной независимо от состояния системного переключателя. Робот продолжает программное выполнение, когда `HOLD/RUN` на операционной панели принимает положение RUN.

По умолчанию состояние переключателя OFF.

MESSAGES

Назначение

Разрешает, запрещает вывод сообщений на терминал.

Пояснение

Если переключатель включен, сообщения выводятся на терминал, когда используются команды PRINT, TYPE. Если переключатель выключен, сообщения не выводятся на терминал.

По умолчанию состояние переключателя ON.

OX.PREOUT

Назначение

Определяет синхронизацию OX сигналов для переключения ON/OFF в шагах инструкций блочного программирования.

Пояснение

Когда рабочая программа обучена инструкциями блочного программирования и этот переключатель включен (ON), OX сигналы, заданные в какой-либо позиции, включаются, как только робот начинает движение в эту позицию.

Если переключатель выключен (OFF), сигнал не включится до тех пор, пока не произойдет совпадение осей с точкой, обученной в данной инструкции.

По умолчанию переключатель находится в состоянии ON.

PREFETCH.SIGINS

Назначение

Разрешает, запрещает более раннюю обработку команд сигналов входа, выхода. Функционирует аналогично OX.PREOUT, но эффективен только для AS программирования.

Пояснение

Если переключатель выключен (OFF), команды для сигналов входа, выхода и синхронизация, описанная выше, не выполняются до тех пор, пока не произойдет совпадение осей с точкой, обученной в текущей инструкции движения.

SWAIT, SIGNAL, TWAIT, PULSE, DLYSIG, RUNMASK, RESET, BITS

Если переключатель включен (ON), команды, указанные выше, выполняются, как только начинается движение в заданную точку, программа обрабатывается вплоть до следующей инструкции движения.

По умолчанию переключатель находится в состоянии OFF.

QTOOL

Назначение

Разрешает, запрещает использовать преобразования инструмента во время обучения в режиме блочного программирования.

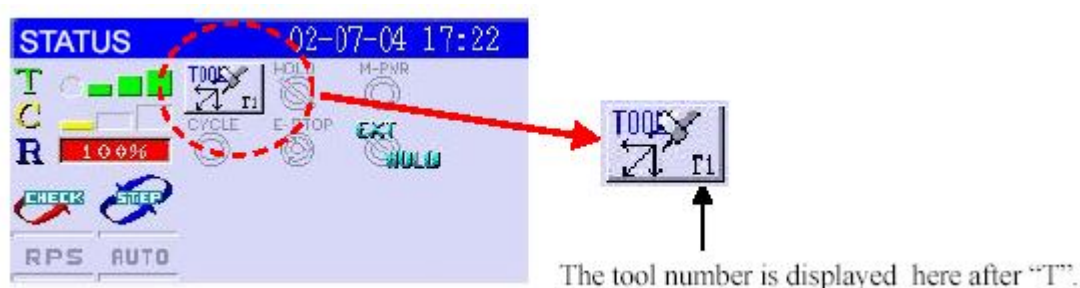
Пояснение

Если переключатель включен, следующие функции возможны во время обучения:

1. Автоматический выбор инструментальной системы координат.

- 1.1. Если дополнительные данные для инструмента задаются в текущем шаге, преобразования инструмента регистрируются как инструмент (TOOL) номер 1 – 9
- 1.2. Если AS команда для инструмента задана в текущем шаге, инструментальная система координат остается неизменной
- 1.3. Если ничего не задано в текущем шаге, а на пульте ручного управления отображена строка дополнительных данных, робот движется соответственно в той инструментальной системе, которая зарегистрирована в строке дополнительных данных в настоящий момент в соответствии с номером инструмента
- 1.4. Если ничего не задано в текущем шаге, и на пульте ручного управления нет строки дополнительных данных, инструментальная система координат остается неизменной

2. Реальный номер инструментальной системы координат отображается на дисплее пульта ручного управления в статус области в верхнем правом углу экрана



Если переключатель выключен, и используется инструментальная система, заданная в AS инструкции, номер инструмента на экране будет 0.

Если переключатель выключен (OFF), инструментальная система координат изменяется при выполнении команды TOOL или выполнении инструкции TOOL, или блочной инструкции внутри программы.

По умолчанию переключатель находится в состоянии ON.

RPS

Назначение

Разрешает, запрещает случайный выбор программ.

Пояснение

Если этот переключатель выключен (OFF), инструкция EXTCALL и дополнительные данные JUMP/END игнорируются

По умолчанию переключатель находится в состоянии OFF.

SCREEN

Назначение

Разрешает, запрещает пролистывание экрана.

Пояснение

Если переключатель включен (ON), пролистывание экрана останавливается, когда экран заполняется. Нажимая Spacebar, выводим следующую информацию.

По умолчанию переключатель находится в состоянии ON.

REP_ONCE

Назначение

Определяет, как выполняется программа, один раз или непрерывно.

Пояснение

Если переключатель включен (ON), программа выполняется один раз. Если выключен, непрерывно.

По умолчанию переключатель находится в положении OFF.

STP_ONCE

Назначение

Определяет, как выполняются шаги программы, по отдельным шагам или непрерывно.

Пояснение

Если переключатель включен (ON), программа выполняется по отдельным шагам. Если выключен, непрерывно.

По умолчанию переключатель находится в положении OFF.

AUTOSTART.PC
AUTOSTART2.PC
AUTOSTART3.PC
AUTOSTART4.PC
AUTOSTART5.PC

Опция

Назначение

Определяет автоматический старт выбранной PC программы по включению силового питания.

Пояснение

Если переключатель включен (ON), программа, названная AUTOSTART.PC стартует автоматически по включению силового питания. Пять различных PC программ могут быть выбраны для автоматического старта, каждая из них названа AUTOSTART.PC, AUTOSTART2.PC – AUTOSTART5.PC. Включение каждого переключателя запускает соответствующую программу.

По умолчанию переключатель находится в положении OFF.

(ПРИМЕЧАНИЕ)

Если этот переключатель находится в состоянии ON, но соответствующих программ не существует, произойдет ошибка и появится сообщение “Program does not exist”.

ERRSTART.PC

Опция

Назначение

Определяет выполнение программы под именем ERRSTART.PC во время прохождения ошибки.

Пояснение

Если переключатель включен (ON), происходит выполнение программы под именем ERRSTART.PC во время прохождения ошибки.

После выполнения программы ERRSTART.PC, переключатель автоматически выключается (OFF).

Возобновить автоматическое выполнение программы ERRSTART.PC для обработки следующей ошибки возможно последующим включением в программу переключателя в состоянии ON.

По умолчанию переключатель находится в положении OFF.

(ПРИМЕЧАНИЕ)

Если этот переключатель находится в состоянии ON, но соответствующей программы не существует, произойдет ошибка и появится сообщение “Program does not exist”.

SWITCH (TRIGGER)

Назначение

Отображает на экране пульта ручного управления, включен или не включен переключатель **TRIGGER(DEADMAN)**.

Эта функция не устанавливает включение или выключение триггера.

Используя с переключателем SWITCH функцию, -1 задается, если **TRIGGER** переключатель включен, 0, если выключен.

SWITCH (CS)

Назначение

Отображает состояние включения или выключения кнопки **CYCLE.START**.

Эта функция не устанавливает включение или выключение кнопки.

Используя с переключателем SWITCH функцию, -1 задается, если **CYCLE.START** переключатель включен, 0, если выключен.

SWITCH (POWER)

Отображает состояние переключателя **MOTOR POWER** при включении, выключении.

Эта функция не устанавливает включение или выключение триггера.

Используя с переключателем SWITCH функцию, -1 задается, если **MOTOR POWER** переключатель включен, 0, если выключен.

SWITCH (RGSO)

Назначение

Отображает состояние ON, OFF серво двигателя.

Эта функция не устанавливает включение или выключение серво двигателя.

Используя с переключателем SWITCH функцию, -1 задается, если серво мотор включен, 0, если выключен.

SWITCH (TEACH_LOCK)

Назначение

Отображает состояние ON, OFF аппаратного переключателя **TEACH_LOCK**.

Эта функция не устанавливает включение или выключение аппаратного переключателя. Используя с переключателем SWITCH функцию, -1 задается, если переключатель включен, 0, если выключен.

SWITCH (ERROR)

Назначение

Показывает, действительно ли случилась ошибка в данный момент времени.

Эта функция не устанавливает включение или выключение аппаратного переключателя.

Используя с переключателем SWITCH функцию, -1 задается, если ошибка произошла, 0, если ошибка не произошла.

SWITCH (REPEAT)

Назначение

Отображает положение аппаратного переключателя **TEACH/REPEAT** на операционной панели (состояние TEACH или REPEAT)

Эта функция не устанавливает включение или выключение аппаратного переключателя.

Используя с переключателем SWITCH функцию, -1 задается, если переключатель в позиции TEACH, 0, если в позиции REPEAT.

SWITCH (RUN)

Назначение

Отображает положение аппаратного переключателя **RUN/HOLD** на операционной панели (состояние RUN или HOLD).

Эта функция не устанавливает включение или выключение аппаратного переключателя.

Используя с переключателем SWITCH функцию, -1 задается, если переключатель в позиции RUN, 0, если в позиции HOLD.

DISPIO_01

Назначение

Меняет вид отображения внешних I/O сигналов и внутренних сигналов при использовании команды IO.

Пояснение

Если системный переключатель в состоянии OFF, “o” показывает включенные сигналы, “x” показывает выключенные сигнала. Приоритетные сигналы показываются соответственно заглавными буквами (“O”, “X”).

Если системный переключатель в состоянии ON, “1” показывает включенные сигналы, “0” показывает выключенные сигналы, “-” показывает сигналы, которые не установлены.

По умолчанию установка переключателя OFF. (см. 5.6)

Пример

DISPIO_01 в состоянии OFF

>IO 

32 - 1	xxxx	xxxx	xxxx	xxXX	xxxx	XXXX	XXXO	XXXO
1032 - 1001	xxxx	xxxx	xxxx	xxXX	xxxx	XXXX	XXXX	OXXX
2032 - 2001	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx

>

DISPIO_01 в состоянии ON

>IO 

32- 0	0000	0000	0000	0000	0000	0000	0001	0001
1032-1001	0000	0000	0000	0000	0000	0000	0000	1000
2032-2001	0000	0000	0000	0000	0000	0000	0000	0000

>

HOLD.STEP

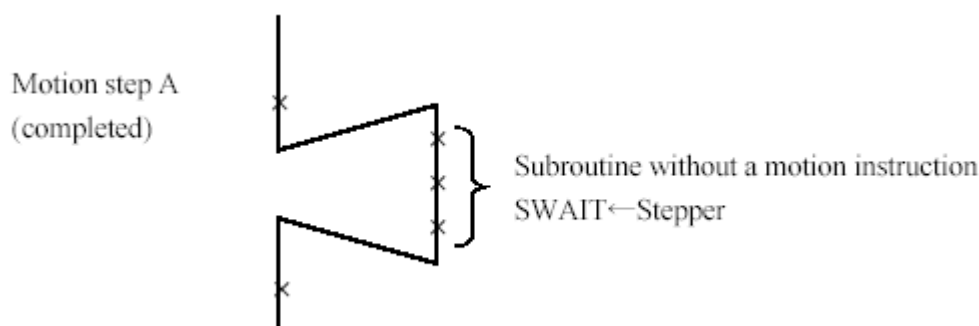
Назначение

Выбирает шаг, показывающий, где программное выполнение приостановлено.

Пояснение

Если системный переключатель в состоянии ON и программное выполнение приостанавливается в момент выполнения инструкции, не содержащей движение, шаг, выполненный в настоящее время, отображается вместо инструкции движения, только что законченной.

Для примера на рисунке ниже, если переключатель в состоянии ON, отображается шаг SWAIT (SWAIT может быть перепрыгнут), если в состоянии OFF, отображается шаг движения A.



По умолчанию установка OFF.

WS_COMPOFF

Опция

Назначение

Меняет выходное состояние сварочного сигнала (WS).

Пояснение

Если переключатель в состоянии ON, синхронизация выходных сварочных сигналов меняется с момента изменения памяти до момента, когда принят ввод законченного сварочного сигнала.

Если переключатель в состоянии OFF, сварочные сигналы выводятся в каждом изменении памяти.

По умолчанию состояние переключателя OFF.

FLOWRATE

Опция

Назначение

Переключается между режимом управления расходом и режимом вывода скоростей

Пояснение

Если этот переключатель в состоянии ON, выбирается режим управления расходом.

Если этот переключатель в состоянии OFF, выбирается режим вывода скоростей.

По умолчанию установка OFF.

WS.ZERO

Опция

Назначение

Меняет совершаемую операцию, когда сварочный сигнал устанавливается в 0.

Пояснение

Если переключатель в состоянии ON, сварка совершается, когда WS=0, также как когда WS не равно 0. (Герметизация и сварка).

Если переключатель в состоянии OFF, сварка не совершается, когда WS=0. (Только герметизация).

По умолчанию состояние переключателя устанавливается OFF.

ABS.SPEED

Назначение

Разрешает, запрещает использование абсолютной скорости. Эта функция разрешает выполнение шагов движения при низкой предопределенной установке скорости, эффективной для полной программы и принимающей приоритет над установленной мониторной скоростью.

Пояснение

Если этот переключатель в состоянии ON, робот движется с абсолютной скоростью, заданной в программе, когда следующее состояние верно.

$$\text{Maximum speed} \times \text{Monitor Speed} > \text{Program Speed}$$

Пример

Если максимальная скорость 2400 mm/s, мониторная скорость 10%, программная скорость 100 mm/s

$$2400 \times 0.1 > 100$$

Робот движется со скоростью 100 mm/s.

Если максимальная скорость 100%, мониторная скорость 10%, программная скорость 5%, то

$$100 \times 0.1 > 5$$

Поэтому, робот движется с программной скоростью 5%.

Если максимальная скорость 2400 mm/s, мониторная скорость 2%, программная скорость 100 mm/s

$$2400 \times 0.02 < 100$$

Поэтому робот движется со скоростью 48 mm/s.

SLOW_START

Назначение

Разрешает, запрещает функцию замедленного старта. Если разрешена, первая инструкция движения выполняется с медленной скоростью.

Если программа остановлена и перезапущена, первый шаг движения будет выполнен с медленной скоростью.

AFTER.WAIT.TIMR

Назначение

Устанавливает синхронизацию для запуска таймеров в инструкциях блочного программирования.

Пояснение

Если переключатель в положении OFF, таймер запускается после совпадения осей. Если переключатель в состоянии ON, таймер запускается, когда оси совпадут и все установочные состояния (WX, WAIT, RPS ON) завершены.

По умолчанию состояние переключателя OFF.

8.0 ОПЕРАТОРЫ

Эта глава описывает, как работают функции в AS языке. Эти операторы используются в соединении с мониторными командами и программными инструкциями.

8.1 Арифметические операторы

8.2 Относительные операторы

8.3 Логические операторы

8.4 Бинарные операторы

8.5 Операторы векторной алгебры

8.6 Строковые операторы

8.1 АРИФМЕТИЧЕСКИЕ ОПЕРАТОРЫ

Арифметические операторы используются для вычисления математических выражений.

Operator	Function	Example
+	Addition	$i = i + 1$
-	Subtraction	$j = i - 1$
*	Multiplication	$i = i * 3$
/	Division	$i = i / 2$
MOD	Remainder	$i = i \text{ MOD } 2$
^	Power	$i = i^3$

MOD – остаток

^ - возведение в степень

Примеры

$i = i + 1$ Значение i плюс 1 присваивается в i . т.е. когда i есть 5, 6 будет присвоено в i как результат выражения $i + 1$, .

$i = i \text{ MOD } 2$ Когда $i = 5$, оператор вычисляет $5 \div 2$ и присваивает остаток 1 в i .

$i = i^3$ Значение i^3 присваивается в i . Когда $i=2$, 8 присваивается в i с левой стороны инструкции.

При делении и вычислении остатка использование 0. как крайне правого значения выражения приведет к ошибке.

Пример

$i = i / 0$

$i = i \text{ MOD } 0$

8.2 ОТНОСИТЕЛЬНЫЕ ОПЕРАТОРЫ (ОПЕРАТОРЫ СРАВНЕНИЯ)

Относительные операторы используются с инструкциями такими, как IF, WAIT для проверки установленного состояния

Operator	Function	Example
<	TRUE (-1) when left side value is less than right side	$i < j$
>	TRUE (-1) when left side value is greater than right side	$i > j$
<=	TRUE (-1) when left side value is less than or equal to right side	$i <= j$
=<	Same as above	$i = < j$
>=	TRUE (-1) when left side value is greater than or equal to right side	$i >= j$
=>	Same as above	$i = > j$
==	TRUE (-1) when the two sides are equal	$i == j$
<>	TRUE (-1) when the two sides are not equal	$i < > j$

Примеры

Example

IF $i < j$ GOTO 10 Когда j больше чем i , (т.е. инструкция $i < j$ истинна), программа переходит на шаг с меткой 10, если нет, то программа обрабатывает следующий шаг.

WAIT $t == 5$ Когда t есть 5 ($t == 5$ истинно), программа переходит для выполнения на следующий шаг, если нет программа ожидает до тех пор, пока состояние не установится.

F $i + j > 100$ GOTO 20 Когда $i + j$ больше чем 100 (т.е. выражение $i + j > 100$ истинно), программа переходит на шаг с меткой 20. Если нет, программа переходит для обработки следующего шага.

IF \$a == "abc" GOTO 20 Когда \$a есть "abc" (т.е. $\$a == \text{"abc"}$ истинно), программа переходит на шаг с меткой 20. Если нет, программа переходит для обработки следующего шага.

8.3 ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Логические операторы используются в операциях Булевой алгебры, $0+1=1$, $1+1=1$, $0+0=0$ (логическое ИЛИ), $0 \times 1=0$, $1 \times 1=1$, $0 \times 0=0$ (логическое И). В AS программном языке существует два типа логических операторов: логические и бинарные операторы.

Логические операторы используются не для вычисления числовых значений, а для определения истинности, ложности значения или выражения. Если числовое значение «0», считается FALSE (OFF), все ненулевые значения считаются TRUE (ON). Однако, заметьте, что при вычислении логического выражения, устанавливается -1 как TRUE (ON).

Operator	Function	Example
AND	Logical AND	i AND j
OR	Logical OR	i OR j
XOR	Exclusive logical OR	i XOR j
NOT	Logical complement	NOT i

Например
i AND j

Вычисляет логическое AND между i и j. Переменные i и j обычно логические величины, но они могут быть и реальными числовыми значениями. В этом случае, все реальные числовые значения, отличные от нуля рассматриваются как ON (TRUE).

i	j	Result
0	0	0 (OFF)
0	not 0	0 (OFF)
not 0	0	0 (OFF)
not 0	not 0	-1 (ON)

Результат истинен, когда оба значения истинны.

i OR j

i	j	Result
0	0	0 (OFF)
0	not 0	-1 (ON)
not 0	0	-1 (ON)
not 0	not 0	-1 (ON)

Результат истинен, когда оба или одно из значений истинны.

i XOR j

i	j	Result
0	0	0 (OFF)
0	not 0	-1 (ON)
not 0	0	-1 (ON)
not 0	not 0	0 (OFF)

Результат истинен, когда одно из двух значений истинно.

NOT i

i	Result
0	-1 (ON)
not 0	0 (OFF)

Вычисляет логическое дополнение i.

В AS, логическое состояние значения или выражения может быть выражено следующим образом:

Истинно: not 0, ON, TRUE

Ложно: 0, OFF, FALSE

8.4 БИНАРНЫЕ ОПЕРАТОРЫ

Бинарные логические операции выполняются для каждого соответствующего бита двоичного числового значения. Например, если число состоит из четырех битов, значения, которые должны вычисляться, могут быть следующими 0000, 0001, 0010, 0011,, 1111 (В AS, числовые значения составляют 32 бита).

:

Binary expression	Decimal expression
0000	0
0001	1
0010	2
0011	3
⋮	⋮
1111	15

В AS числовое значение составляет из 32 битов.

Operator	Function	Example
BOR	Binary OR	i BOR j
BAND	Binary AND	i BAND j
BXOR	Binary XOR	i BXOR j
COM	Binary complement	COM i

Например
i BOR j

$$\begin{array}{r} i=5 \quad 0101 \\ j=9 \quad 1001 \\ \hline 1101 \Rightarrow 13 \end{array}$$

i BAND j

If i=5, j=9, then the result is 1.

$$\begin{array}{r} i=5 \quad 0101 \\ j=9 \quad 1001 \\ \hline 0001 \Rightarrow 1 \end{array}$$

i BXOR j

If i=5, j=9, then the result is 12.

$$\begin{array}{r} i=5 \quad 0101 \\ j=9 \quad 1001 \\ \hline 1100 \Rightarrow 12 \end{array}$$

COM i

If i=5 then the result is -6.

$$\begin{array}{r} i=5 \quad 0\dots \quad 0101 \\ 1\dots \quad 1010 \Rightarrow -6 \end{array}$$

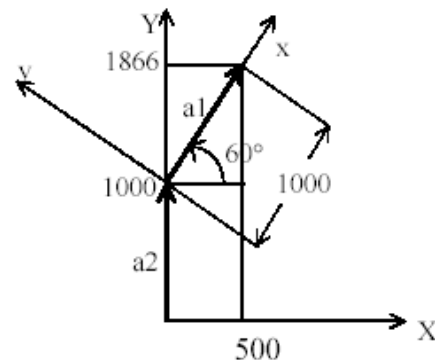
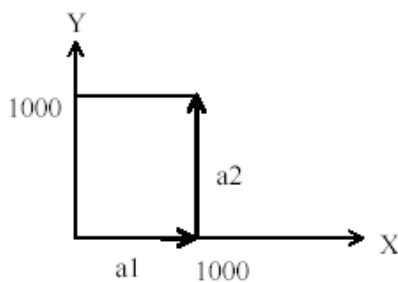
8.5 ОПЕРАТОРЫ ВЕКТОРНОЙ АЛГЕБРЫ

В AS системе операторы векторной алгебры используются при вычислениях и создании сложных точек. В AS системе операторы $+$ и $-$ используются, для того чтобы определить значение сложной координаты (XYZOAT значения). Однако, обратите внимание, что в отличие от обычных сложения и вычитания, переместительный закон не верен для операций векторной алгебры. Арифметическое выражение “ $a + b$ ” и “ $b + a$ ” имеют одинаковый результат, но “ $\text{pose } a + \text{pose } b$ ” не обязательно будет равен “ $\text{pose } b + \text{pose } a$ ”. Это происходит потому что в операциях преобразований, положение осей принимается во внимание. Как пример этого, показано ниже:

$$\begin{aligned} a1 &= (1000, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0) \\ a2 &= (\quad 0, 1000, \quad 0, \quad 60, \quad 0, \quad 0) \end{aligned}$$

$$a1+a2 = (1000, 1000, \quad 0, 60, \quad 0, \quad 0)$$

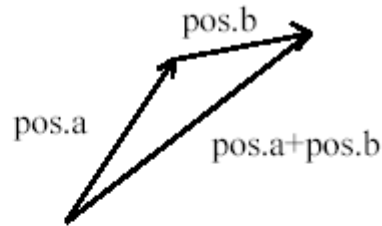
$$a2+a1 = (500, 1866, \quad 0, 60, \quad 0, \quad 0)$$



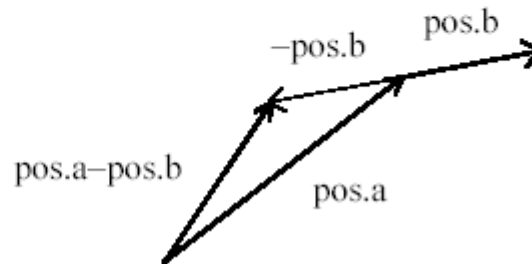
Operator	Function	Example
+	Addition of two transformation values	$\text{pos.a}+\text{pos.b}$
-	Subtraction of two transformation values	$\text{pos.a}-\text{pos.b}$

Например

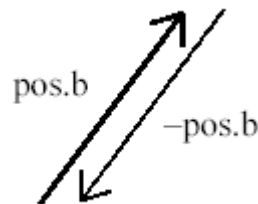
$\text{pos.a} + \text{pos.b}$



$\text{pos.a} - \text{pos.b}$

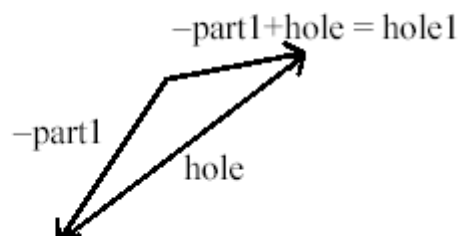


Векторный оператор “-” используется для однозначного задания обратного значения вектора pos.b . Например, когда вектор pos.b определяет положение объекта В относительно объекта А, вектор $-\text{pos.b}$ определяет положение объекта А относительно объекта В.



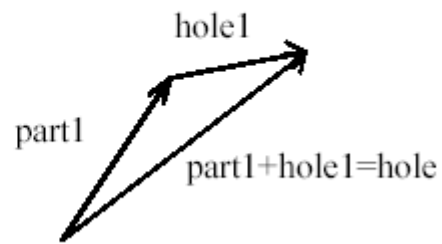
В примере, показанном ниже, “hole1” должно быть определено относительно “part1”, обученной ранее. Это может быть сделано при помощи задания сложной точки $\text{part1} + \text{hole}$. Переместить робот в позицию, которая будет определять “hole1” и обучить эту точку как “hole”, используя команду HERE. “hole1” может быть задана следующим образом:

`POINT hole1 = - (part1)+hole`



Или

POINT part1+hole1 = hole



СТРОКОВЫЕ ОПЕРАТОРЫ

Operator	Function	Example
+	Combines two strings	\$a = \$b + \$c

Например

\$a=\$b+\$c

Объединяет \$b+\$c и присваивает это значение \$a. Если \$b=abc, \$c=123, то \$a будет “abc123”.

9.0 ФУНКЦИИ

Эта глава описывает функции, используемые в AS системе. Функции в основном используются в комбинации с мониторными командами и программными инструкциями. Формат функции следующий: это ключевое слово и параметры, заключенные в круглые скобки. Ключевое слово задает функцию, параметр определяет значение

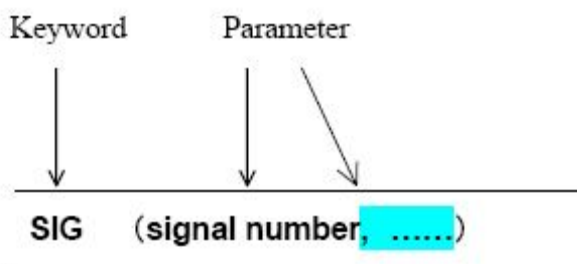
9.1 Функции реальных значений

9.2 Функции значений позиции

9.3 Математические функции

9.4 Строковые функции

EXAMPLE



Parameters marked by can be omitted.

Always enter a space (blank) after the keyword.

9.1 ФУНКЦИИ РЕАЛЬНЫХ ЗНАЧЕНИЙ

SIG	Выдает логическое AND определенного сигнала.
BITS	Выдает значение, соответствующее битовому образцу сигнала.
TIMER	Выдает текущее значение таймера.
DISTANCE	Выдает расстояние между двумя точками
DX, DY, DZ	Выдает значение декартовых координат (X, Y, Z)
DEXT	Выдает заданный компонент позиции (опция)
ASC	Выдает значение символов ASCII.
LEN	Выдает длину строки.
TRUE, ON	Выдает значение -1.0, которое представляет TRUE.
FALSE, OFF	Выдает значение 0.0, которое представляет FALSE.
VAL	Выдает реальное значение в данной строке.
INSTR	Выдает значение для начальной точки определенной строки.
MAXVAL	Сравнивает данные значения.
MINVAL	Сравнивает данные значения .
INT	Выдает значение в целом числе.
SWITCH	Выдает состояние системного переключателя.
TASK	Выдает состояние выполнения программы.
ERROR	Выдает номер ошибки.
PLCAIN	Выдает количество данных (опция)
PRIORITY	Выдает приоритет программ робота (опция)
UTIMER	Выдает текущее значение таймера.
MSPEED	Выдает мониторинговую скорость робота 1. (опция)
MSPEED2	Выдает мониторинговую скорость робота 2. (опция)
INRANGE SYSDATA	Проверяет находится ли позиция в диапазоне движения (опция) Выдает параметры AS (опция)

SIG(signal number)

Назначение

Выдает результат логического оператора AND заданного бинарного состояния сигналов.

Параметр

Задает число внешних или внутренних I/O сигналов.

Пояснение

Вычисляет логическое AND всех заданных бинарных состояний сигналов и выдает получающееся значение. Если все заданные сигнальные состояния истинны, выдается -1, если ложны, выдается 0.

Допустимые сигнальные номера

Внешние выходные сигналы – 1 – реально смонтированные

Внешние входные сигналы - 1001 – реально смонтированные

Внутренние сигналы – 2001 – 2256

Сигналы, заданные положительным числом, считаются TRUE, когда они включены, сигналы, заданные отрицательным числом считаются TRUE, когда они выключены.

Никакой сигнал не соответствует «0» номеру, так что он всегда рассматривается как TRUE.

(ПРИМЕЧАНИЕ)

Есть ограничение синхронизации при обработке более чем одного сигнала в одно и тоже время. Когда происходит оценка более чем одного сигнала в одно и тоже время, обратите внимание, что есть 2 миллисекундное различие в стабилизации времени каждого сигнала.

Например

Если бинарные I/O сигналы имеют состояние 1001=ON, 1004=ON, 20=OFF, то

	Results	
SIG(1001)	-1	TRUE
SIG(1004)	0	FALSE
SIG(-1004)	-1	TRUE
SIG(1001,1004)	0	FALSE
SIG(1001,-1004)	-1	TRUE
SIG(1001,-1004,-20)	-1	TRUE

BITS (starting signal number, number of signal)

Назначение

Считывает последовательные двоичные сигналы и выдает десятичное значение в соответствии с битовой комбинацией заданных двоичных сигналов.

Параметры

1. Стартовый сигнальный номер

Задаёт первый сигнал для считывания

2. Количество сигналов

Задаёт количество сигналов для считывания, максимальное число 16.

Пояснение

Эта функция возвращает десятичное значение, соответствующее разрядному образцу указанных сигналов.

В двоичном выражении значения, возвращенного этой функцией, наименьшему значащему биту соответствует стартовый номер сигнала.

Допустимые сигнальные номера

Внешние выходные сигналы – 1 – реально смонтированные

Внешние входные сигналы - 1001 – реально смонтированные

Внутренние сигналы – 2001 – 2256

(ПРИМЕЧАНИЕ)

Есть ограничение синхронизации при обработке более чем одного сигнала в одно и тоже время. Когда происходит оценка более чем одного сигнала в одно и тоже время, обратите внимание, что есть 2 миллисекундное различие в стабилизации времени каждого сигнала.

Пример

`x= BITS(1003,4)`

Задаётся считывание четырех сигналов, начиная с 1003 (т.е. 1003, 1004, 1005, 1006). Если битовая комбинация будет следующей

Signals:	1008	1007	<u>1006</u>	<u>1005</u>	<u>1004</u>	<u>1003</u>	1002	1001
State:	1	1	0	1	0	1	0	0

т.е. 0101 – битовая комбинация, x присвоится десятичное значение 5.

TIMER (timer number)

Назначение

Выдает результирующее значение заданного таймера в секундах. Выражает таймерное значение момента, когда TIMER функция выполнялась.

Параметр

Задаёт, какой таймер считывать. Допустимые границы 1 – 10.

Пояснение

При использовании функции `TIMER`, значение таймера может быть считано в любой момент времени. Считывание и выдача результирующего времени (в секундах) происходит исходя из значений, предварительно заданных инструкцией `TIMER`. Если значение не было установлено программной инструкцией, значение функции `TIMER` выдаётся 0.

Номер 0 таймера устанавливается для системного счетчика времени. Значение, выдаваемое при задании этого таймера, есть общее затраченное время с момента первоначального запуска контроллера.

Пример

```
TIMER (1)=0  
CALL test.routine  
PRINT "Elapsed time=", TIMER(1),"seconds"
```

В данном примере показано, как используется инструкция и функция `TIMER` для показа времени, израсходованного при выполнении подпрограммы.

DISTANCE (transformation value, transformation value)

Назначение

Вычисляет расстояние между двумя точками, заданными в декартовых координатах.

Параметр

Декартовая координата, декартовая координата

Задаёт имена точек, между которыми необходимо рассчитать расстояние.

Пояснение

Выдаёт расстояние между двумя позициями в мм. (Две позиции могут вводиться в любом порядке)

Пример

```
k=DISTANCE(HERE,part)
```

Вычисляет расстояние между текущим TCP и позицией “part”, и присваивает результат в k.

DX (transformation value)
DY (transformation value)
DZ (transformation value)

Назначение

Выдает декартовые значения X, Y, Z заданной позиции

Параметр

Декартовая координата

Задаёт имя позиции, из которой присваиваются значения.

Пояснение

Эти три функции выдают X, Y, или Z компоненты определенной позиции.

(ПРИМЕЧАНИЕ)

Каждый компонент декартовых значений может быть также получен, используя инструкцию DECOMPOSE. Значения для O, A, и T получаются, используя инструкцию DECOMPOSE.

Пример

Если точка с именем “start” имеет координаты

X	Y	Z	O	A	T
125,	250,	–50,	135,	50,	75

x=DX(start)	DX function returns x = 125.00
y=DY(start)	DY function returns y = 250.00
Z=DZ(start)	DZ function returns z = –50.00

DEXT (pose variable name, element number)

Опция

Назначение

Присваивает указанный элемент указанной позиции.

Параметры

1. Задаёт позицию в угловых или декартовых координатах
2. Задаёт выводимый элемент реальным числом

Element number	Pose	
	Transformation values	Joint displacement values
1	X component	JT1
2	Y component	JT2
3	Z component	JT3
4	O component	JT4
5	A component	JT5
6	T component	JT6
7	JT7	JT7

Пример

Если декартовые координаты для точки “aa” 0, 0, 0,-160, 0, 0, 300, затем вводится функция как

```
type DEXT(aa, 7)
```

300 устанавливается как значение седьмого сустава переменной “aa”.

ASC (string, character number)

Назначение

Присваивает ASCII значение заданного символа в строчном выражении.

Параметры

1.Строка

Задаёт строку, которая содержит символы, для которых ASCII значения необходимы. Если строка есть пустая строка или число, заданное для параметра «число символов» превышает реальное число символов в строке, выдается -1.

2.Количество символов

Задаёт количество символов, отсчитываемых от начала строки. Если параметр не задан или задан 0, или 1, выдается ASCII значение первого символа строки.

Пояснение

ASCII значение возвращается в реальных значениях.

Пример

```
ASC("sample", 2)
```

Выдаёт ASCII значение для символа «a»

```
ASC($name)
```

Выдаёт ASCII значение для первого символа строки “\$name”

`ASC($system, i)`

Выдает ASCII значение для i-го символа в строковой переменной “\$system”

LEN(string)

Назначение

Выдает число символов в заданной строке

Параметр

Строка

Задаёт символьную строку, переменную символьной строки, строчное выражение

Пример

`LEN("sample")`

Выдает число символов строки “sample”, которое равно 6.

...TRUE...

...ON...

Назначение

Выдает логическое значение для TRUE (-1).

Пояснение

Эта функция удобна, когда необходимо определить логическое состояние TRUE. Результат функций TRUE и ON одинаковы. Выбирается функция, которая лучше подходит к нуждам программы.

...FALSE...

...OFF...

Назначение

Выдает логическое значение для FALSE (0).

Пояснение

Эта функция удобна, когда необходимо определить логическое состояние FALSE. Результат функций FALSE и OFF одинаковы. Выбирается функция, которая лучше подходит к нуждам программы.

VAL(string, code)

Назначение

Выдает действительное значение в заданной строке

Параметры

Строка

Задаёт символьную строку, переменную символьной строки, строчное выражение.

Код

Выражается в реальном значении или выражении, задаёт систему исчисления числа. Если задано числом отличным от 1,2,0, присваивается 0 (десятичная система).

0 – десятичная система

1 – двоичная система

2 – шестнадцатеричная система

(ПРИМЕЧАНИЕ)

Экспоненциальное выражение может быть использовано в строке.

Коды, которые задаются в специальном выражении (т.е.. ^B и ^H) могут быть добавлены в начале строки.

Все символы, которые не читаются как значение числа или кода для исчисления интерпретируются, как символы, указывающие конец строки.

VAL("123 ")	Returns the real value 123.
VAL("123abc ")	Returns the real value 123.
VAL("12 ab 3 ")	Returns the real value 12.
VAL("1.2E-5")	Returns the real value 0.00001.
VAL("^HFF")	Returns the real value 255. (^H means hexadecimal notation. 16×15+15=255)

INSTR(starting point, string1, string2)

Назначение

Отображает место (в действительном значении), где начинается заданная строка в данной строке.

Параметры

1. Начальная точка

Задаёт откуда в строке 1 искать строку 2. Если параметр не задан, поиск начинается с начала строки 1.

2. Строка 1

Выражает в символьной цепочке, переменной символьной цепочки, строчным выражением, задает строку, где ищется строка 2.

3. Строка 2

Выражается в символьной цепочке, переменной символьной цепочки, строчным выражением, задает строку, которая ищется в строке 1.

Пояснение

Эта функция отображает значение начальной точки строки 2 в строке 1, если строка 2 включена в строку 1, если не включена, устанавливается 0. Если значение, заданное как начальная точка, равно или меньше 1, поиск начинается с начала строки 1. Если значение начальной точки больше чем число символов 1 строки, устанавливается 0. Заглавные и строчные буквы не различаются.

Пример

INSTR("file.ext", ".")	Real value 5 is returned.
INSTR("file", ".")	Real value 0 is returned.
INSTR("abcdefgh", "DE")	Real value 4 is returned.
INSTR(5, "1-2-3-4", "-")	Real value 6 is returned.

MAXVAL (real value1, real value 2,)

Назначение

Сравнивает данные действительные значения и отображает наибольшее из них.

Параметр

Реальное значение 1, реальное значение 2
Задаёт данные для сравнения.

MINVAL ((real value1, real value 2,)

Назначение

Сравнивает данные действительные значения и отображает наименьшее из них.

Параметр

Реальное значение 1, реальное значение 2
Задаёт данные для сравнения.

INT (numerical expression)

Назначение

Выделяет целое число заданного числового выражения.

Параметр

Числовое выражение

Пояснение

Возвращает целое число с левой стороны до десятичной точки (со значениями не в экспоненциальном формате). Отрицательный признак остается с целым числом, если целое число не 0.

Пример

INT(0.123)	0 is returned.
INT(10.8)	10 is returned.
INT(-5.462)	-5 is returned.
INT(1.3125E+2)	131 is returned.
INT(cost+0.5)	The value of "cost"

Значение "cost" округляется с понижением до ближайшего целого числа.

SWITCH (switch name)

Назначение

Отражает текущее состояние заданного системного переключателя.

Пояснение

-1 – состояние переключателя ON
0 – состояние переключателя OFF.

TASK (task number)

Назначение

Отражает состояние выполнения программы, заданной номером задачи.

Параметр

Номер задачи

1: робот 1	
2: робот 2	
1001: PC программа 1	1004: PC программа 4
1002: PC программа 2	1005: PC программа 5
1003: PC программа 3	

Пояснение

Эта функция выдает состояние выполнения программы

Для примера, эта функция может быть использована для контроля состояния выполнения РС программы из программы управления роботом. Тогда действия робота могут быть согласованы в соответствии с состоянием РС программы.

Значения, отображаемые при использовании этой функции

0 – не выполняется

1 – программа выполняется

2 – программное выполнение приостановлено

3 – выполнение шага завершено, ожидание в течение завершения движения робота

ERROR

Назначение

Отображает код текущей ошибки.

Пояснение

Выдает код ошибки, которая произошла в настоящее время

Значение 0 выдается, когда ошибка не произошла.

Пример

```
TYPE $ERROR(ERROR)
```

TYPE инструкция выводит на экран сообщение об ошибке с кодом ошибки, определенном при помощи функции ERROR.

PLCAIN (data number)

Опция

Назначение

Отображает значение числа указанных данных в целых числах.

Параметр

Количество данных

Задаёт число вводимых данных в целых числах. Допустимые границы 1 – 32.

(ПРИМЕЧАНИЕ)

Эта функция действительна только тогда, когда опция “Built-in Sequencer Function” в состоянии ON. Если опция отключена OFF, следующее сообщение появляется на экране (E1102) Cannot execute, no option set up. - Check option specs.

Пример

```
aa=PLCAIN(12)
```

Отображает значение 12-ой вводимой данной в целых числах.

PRIORITY

Опция

Назначение

Отображает приоритетный номер текущей программы робота.

Пояснение

Отображает приоритетный номер (в реальной величине) текущей программы робота, в настоящее время выбранной в стеке.

Нет никакого урегулирования установки приоритета среди РС программ.

По умолчанию для программ робота приоритет устанавливается 0. Приоритетный номер может быть изменен при помощи инструкции LOCK.

UTIMER (@ timer variable name)

Назначение

Отображает текущее значение переменной @timer variable, установленной при помощи инструкции UTIMER.

Параметр

@ timer variable name

Определяет имя переменной таймера, заданного инструкцией UTIMER. Символ @ добавляется в начало имени переменной так, чтобы можно было определить имя целого числа переменной.

MSPEED MSPEED2

Опция

Назначение

Отображает мониторинговую скорость, установленную в данный момент (0 – 100%). MSPEED для робота 1, MSPEED2 для робота 2.

INRANGE (pose variable, joint displacement values)

Назначение

Проверяет, находится ли позиция внутри границ движения робота и отображает величину, зависящую от результата этой проверки.

Параметры

1. Переменная положения

Задаёт, какая точка проверяется (декартова, угловая, сложная)

2. Величина перемещения сустава

Этот параметр вводится только тогда, когда указанная позиция задается как точка с декартовыми координатами или сложная. Конфигурация робота вычисляется при помощи данного параметра. Если параметр не задан, используется текущая конфигурация.

Пояснение

Значения, получаемые при использовании этой функции следующие:

- 0 – вне пределов движения
- 1 – первый сустав вне пределов движения
- 2 – второй сустав вне пределов движения
- 4 – третий сустав вне пределов движения
- 8 – четвертый сустав вне пределов движения
- 16 – пятый сустав вне пределов движения
- 32 – шестой сустав вне пределов движения
- 16384 – проверка диапазона вне зоны столкновения
- 32768 – вне досягаемости руки робота.

(ПРИМЕЧАНИЕ)

Эта функция проверяет, находится ли позиция в диапазоне движения, но не проверяет, находится ли траектория, которой принадлежит позиция, внутри диапазона движения.

Пример

```
IF INRANGE(pos1, #p) GOTO ERR STOP
:
ERR_STOP:
TYPE "pose pos1is out of motion range."
PAUSE
```

Происходит переход на метку ERR_STOP, если точка pos1 вне пределов движения, выводится сообщение и робот останавливается.

SYSDATA (keyword, opt1, opt2)

Опция

Назначение

Выдает указанные параметры в AS системе соответственно данному ключевому слову

Параметры

Ключевое слово, opt1, opt2

M.SPEED

Выдает мониторинговую скорость в %. Если никакой шаг движения не выполняется, выдается -1.

Opt1: Номер робота. Если не указан, присваивается 1.

Opt2: Не используется.

MSTEP

Выдает номер шага текущего шага движения или последнего выполненного шага движения. Если ни один шаг не выполняется, выдается -1.

Opt1: Номер робота. Если не указан, присваивается 1.
Opt2: Не используется.

STEP

Выдает номер шага текущего шага движения или последнего выполненного шага движения. Если ни один шаг не выполняется, выдается –1.

Opt1: Номер робота или номер РС задачи (1001 до номера РС программы). Если не указан, присваивается 1.
Opt2: Не используется.

P.SPEED

Выдает скорость текущего движения в % или скорость следующего движения. Если скорость установлена в секундах, выдается –1.

Opt1: Номер робота. Если не указан, присваивается 1.
Opt2: Не используется.

P.SPEED.M

Выдает скорость текущего движения в mm/s или скорость следующего движения. Если скорость установлена в секундах, выдается –1.

Opt1: Номер робота. Если не указан, присваивается 1.
Opt2: Не используется.

9.2 ФУНКЦИИ ПОЗИЦИОННЫХ ЗНАЧЕНИЙ

DEST	Выдает позицию назначения как декартовую координату.
#DEST	Выдает позицию назначения как угловую координату.
FRAME	Выдает значения декартовых координат для координат плоскости рамки.
NULL	Выдает ноль значений декартовых координат.
HERE	Выдает значение декартовых координат текущей позиции.
#HERE	Выдает значение угловых координат текущей позиции.
TRANS	Выдает значение декартовых координат, составленных из данных компонентов.
RX, RY, RZ	Выдает значение преобразования, выраженное вращением вокруг осей.
#PPOINT	Выдает значение угловых координат, составленных из данных компонентов.
SHIFT	Выдает значение декартовых координат, преобразованных при помощи сдвига начальной позиции
AVE_TRANS	Выдает значение декартовой координаты средней позиции между двух позиций.
BASE	Выдает значение декартовой координаты для смещения базовой системы координат.
TOOL	Выдает значение декартовой координаты для смещения инструментальной системы координат.
TRADD	Выдает значение X компоненты при добавлении значения компоненты перемещающейся оси. (опция)
TRSUB	Выдает значение X компоненты при отнимании значения компоненты перемещающейся оси. (опция)
#HOME	Выдает значение домашней позиции в угловых координатах (опция)
CCENTER	Выдает значение декартовой координаты для центра окружности, описанной заданными позициями (опция)
CSHIFT	Выдает значение декартовой координаты положения, сдвинутого по направлению к центру окружности, описанной заданными позициями (опция)

DEST

#DEST

Назначение

DEST: Выдает декартовые координатные значения точки назначения текущего движения робота.

#DEST: Выдает угловые координатные значения точки назначения текущего движения робота.

Пояснение

При использовании этих функций, точка назначения может быть распознана после того, как движение робота прервалось по некоторым причинам. Эти функции могут использоваться со всеми движениями робота.

(Примечание)

Позиция, где робот останавливается и позиция, заданная при помощи DEST/#DEST не всегда тоже самое. Для примера, если переключатель **RUN/HOLD** на операционной панели поворачивается в положение HOLD, робот останавливается мгновенно, но позиция, заданная при помощи DEST/#DEST, описывает позицию, которую робот достигал в тот момент.

Пример

DEST/#DEST функции удобны, когда необходимо возобновить движение, прерванное при помощи инструкции ONI...CALL. Включите следующие инструкции в подпрограмму, как показано ниже, для того чтобы робот мог завершить незаконченное движение.

POINT save=HERE	Записывает текущую позицию как "save"
POINT old=DEST	Записывает позицию назначения как "old"
JDEPART 50.0	Отходит назад на 50 мм по оси Z инструмента
:	
:	
JAPPRO save, 50.0	Приближается к позиции "save", не доходя до нее 50 мм
LMOVE save	Двигается в точку "save"
LMOVE old	Двигается в точку "old"

FRAME (transformation values1, transformation values2, transformation values3, transformation values4)

Задаёт относительную координатную плоскость в базовой системе координат. Заметьте, что только поступательные компоненты используются, как позиционная информация, для определения координатной плоскости.

Параметры

Точка 1, точка 2 (в декартовых координатах)

Задают направление оси X. Ось X, задаваемой координатной плоскости, устанавливается так, что она проходит через эти две точки.

Точка 3 (в декартовых координатах)

Задаёт направление оси Y. Ось Y координатной плоскости устанавливается так, что три точки: точка 1, точка 2, точка 3 образуют XY плоскость, точка 3 задаёт положительное направление оси Y.

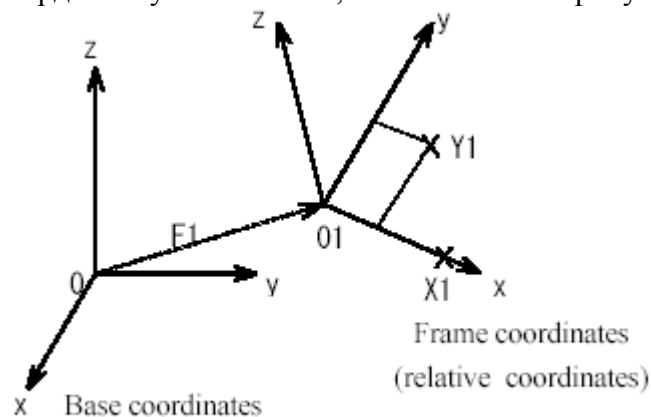
Точка 4

Задаёт начало координатной плоскости, которое равно значению, установленному данной функцией.

Пояснение

```
POINT F1=FRAME(O1, X1, Y1, O1)
```

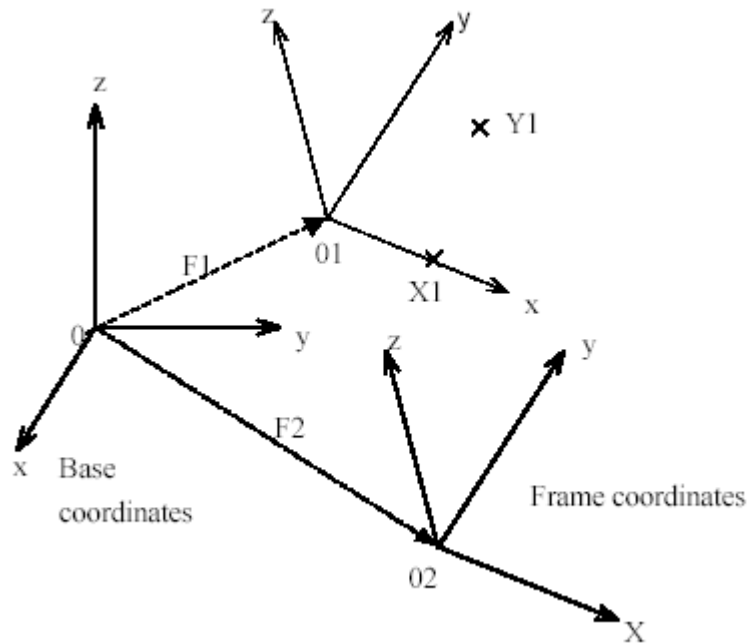
Устанавливает координатную плоскость, как показано на рисунке ниже.



Если позиция в координатной плоскости обучена как F1+A, затем только F1 нуждается в переобучении, если координаты обученной траектории меняют свое положение в пространстве.

```
POINT F2=FRAME(O1, X1, Y1, O2)
```

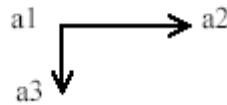
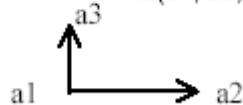
Устанавливает координатную плоскость, как показано ниже



(ПРИМЕЧАНИЕ)

Обратите внимание на следующие точки, когда определяется координатная плоскость.

POINT F=FRAME(a1, a2, a3, a1)



Обратите внимание, что оси Y, Z обеих координатных систем повернуты в противоположные стороны в зависимости от того, как обучена точка a3. Поэтому, когда a3 обучена близко к оси X, направление оси Z может не иметь желательное направление, так что проверяйте координатную плоскость перед использованием координат в любых программах.

Три точки a1, a2, a3 определяются положением начала инструментальной системы координат. Когда переопределяется координатная плоскость, преобразования инструмента должны быть такими же, как при первоначальном обучении a1, a2, a3.

Для более лучшей точности, обучайте три точки a1, a2, a3 настолько далеко друг от друга, как только возможно. Особенно a3, a3 может быть точкой вблизи оси Y, но как можно дальше от оси X.

При обучении a1, a2, a3 начало инструментальной системы координат должно быть определено в точке, которую легко увидеть, например в наконечнике инструмента. С некоторыми инструментами очень трудно определить начало инструментальной системы координат, когда оно перемещено заданным преобразованием инструмента. В этом случае, a1, a2, a3 обучается без инструмента, но обратите внимание, что в этом случае начало инструментальной системы координат находится в центре фланца робота.

NULL

Назначение

Возвращает нулевое значение преобразований

Пояснение

Возвращает значениям преобразования (декартовым координатам), в которых все поступательные и вращательные элементы устанавливаются в 0.

$X=Y=Z=0$, $O=A=T=0$. Эта функция удобна, когда используется с SHIFT функцией, допускающей легко переопределить координату точки. Могут быть сдвинуты только координаты поступательного движения без любых изменений компонентов вращения (OAT).

Пример

```
POINT new=SHIFT(NULL BY x.shift,y.shift,z.shift)+old
```

Определяет переменную “new” при помощи сдвига координат точки, обученной как “old” на определенные расстояния вдоль базовых координат.

```
dist=DISTANCE(NULL, test.pos)
```

Вычисляет расстояние между точкой “test.pos” и началом базовой системы координат, присваивает эту величину переменной dist.

HERE

#HERE

Назначение

HERE: Выдает декартовые координатные значения, которые описывают текущее положение инструментальной системы координат.

#HERE: Выдает угловые координатные значения, которые описывают текущее положение инструментальной системы координат.

Пояснение

В момент выполнения этой функции считываются значения энкодера, поэтому, заметьте, что значения, полученные при помощи этой функции, представляют положение робота, когда эта функция была выполнена.

Имя “here” не может быть использовано как программное имя или имя переменной.

Пример

```
dist=DISTANCE(HERE, pos1)
```

Вычисляет расстояние между точкой “pos1” и текущей позицией и присваивает это значение “dist”.

TRANS (X component, Y component, Z component, O component, A component, T component)

Назначение

Выдает результат координатных значений, которые имеют поступательные и вращательные компоненты.

Параметры

X, Y, Z

Задаёт поступательные компоненты. Если параметр не задан, присваивается 0.

O, A, T

Задаёт вращательные компоненты. Если параметр не задан, присваивается 0.

Пояснение

Координаты точки вычисляются из значений, заданных в каждом параметре. Новые значения могут быть использованы затем для определения позиции сложной точки или программной инструкции движения. Эта функция удобна при использовании с инструкцией DECOMPOSE.

Пример

```
POINT temp.pos=TRANS(v[0], v[1]+100, v[2], v[3], v[4], v[5])
```

Array variable v[0] –

RX (angle)

RY (angle)

RZ (angle)

Назначение

Выдает результат координатного значения, которое представлено вращением вокруг заданной оси.

Параметр

Угол

Задаёт значение вращения в градусах.

Пояснение

X, Y, Z, в этой функции, представляют координатные оси. Выдаются значения вращения вокруг осей. Поступательные компоненты не используются с этой функцией (X,Y,Z=0).

Пример

```
POINT x_rev =RX(30)
```

Выдает результат координатного значения, которое представляет вращение на 30 градусов вокруг оси X и присваивает его величине “x_rev”.

#POINT (jt1, jt2, jt3, jt4, jt5, jt6)

Назначение

Выдает результат определенных угловых значений.

Параметр

Определяет значение каждого угла (в градусах для каждого вращающегося сустава). Если параметр не задан, присваивается 0.

Пояснение

Эта функция выдает определенные угловые значения. Эти значения представляют смещения каждого сустава, от первого до последнего (необязательно шесть)

Пример

В примере, данном ниже, суставы 2 и 3 шести степенного робота перемещаются на определенную величину от текущей позиции.

```
HERE #ref
```

```
DECOMPOSE x[0]=#ref
```

```
JMOVE #PPOINT (X[0], x[1]+a, x[2]-a/2, x[3], x[4], x[5])
```

Записывает текущую позицию в память (#ref). Раскладывает каждый компонент в элементы массива переменных x[0],...,x[5].

То же самое действие можно сделать следующим образом

```
DRIVE 2, a, 100
```

```
DRIVE 3, -a/2, 100
```

SHIFT (transformation values BY X shift, Y shift, Z shift)

Назначение

Выдает результат координатных значений позиции, сдвинутой на определенное расстояние вдоль каждой базовой оси (X, Y, Z) от позиции, описанной декартовой координатой

Параметр

1. Декартовая координата

Задаёт имя точки, которую необходимо сдвинуть.

2. X shift, Y shift, Z shift

Задаёт величину сдвига вдоль соответствующих осей.

Пояснение

X, Y, Z сдвиги добавляются к соответствующим координатам данной точки. Результатом будет новая точка.

Пример

Если координата x (200, 150, 100, 10, 20, 30)

```
POINT y=SHIFT(x BY 5, -5, 10)
```

Точка x сдвигается на определенные значения вдоль осей X, Y, Z. Новые полученные значения присваиваются точке y (205, 145, 110, 10, 20, 30).

AVE_TRANS (transformation values 1, transformation values 2)

Назначение

Выдает средние значения двух координатных значений.

Параметр

Декартова координата 1, Декартова координата 2
Имена точек.

Пояснение

Эта функция вычисляет координатные значения позиции, которая задается серединой каждого компонента точек 1 и 2.

Обычно эта функция используется, чтобы получить среднее значение позиции от проверочного датчика.

(ПРИМЕЧАНИЕ)

Для XYZ компонентов, среднее значение получается сложением каждого компонента и делением на 2. Однако для ОАТ компонентов среднее значение не всегда вычисляется подобным способом

Пример

```
POINT x = AVE_TRANS(p,q)  
JMOVE AVE TRANS(p,q)
```

BASE

Назначение

Выдает текущее декартовое значение базовой системы координат

Пример

```
point a = BASE
```

Присваивает переменной «а» текущее значение базовой системы координат.

TOOL

Назначение

Выдает текущее значение инструментальной системы координат.

Пример

```
point aa = TOOL
```

Присваивает переменной «aa» текущее значение инструментальной системы координат.

TRADD (transformation values)

Опция

Назначение

Выдает сумму, полученную от сложения перемещающейся оси и X координаты заданной точки.

Параметр

Значение декартовой координаты

Задаёт значение декартовой координаты, для того чтобы добавить к значению перемещающейся оси.

TRSUB (transformation values)

Опция

Назначение

Выдает разность, полученную от вычитания значения перемещающейся оси из X координаты заданной точки.

Параметр

Значение декартовой координаты

Задаёт значение декартовой координаты, для того чтобы вычесть значение перемещающейся оси.

#HOME (home position number)

Опция

Назначение

Выдает позицию HOME, установленную в настоящее время.

Параметр

Номер домашней позиции

1 – выбирает позицию HOME 1, установленную командой SETHOME

2 – выбирает позицию HOME 2, установленную командой SET2HOME.

Если параметр не задан, по умолчанию устанавливается 1.

Пояснение

Выдает значение домашней позиции (обычно позиции безопасности) в угловых координатах

(ПРИМЕЧАНИЕ)

Функция может обрабатывать только угловые значения. Поэтому всегда ставьте знак # перед функцией, чтобы не путать с инструкцией HOME

Пример: в программе, указанной ниже, робот движется не по заданной траектории, а сначала перемещается на ту же высоту (по оси Z), что и высота позиции безопасности, и только затем перемещается в позицию безопасности.

```
POINT homepos = #HOME(1)
IF DZ(homepos) > DZ(HERE) THEN
  HERE tmp
  POINT/Z tmp = homepos
  LMOVE tmp
END
HOME
```

CCENTER (transformation values 1, transformation values 2, transformation values 3, configuration transformation values)

Опция

Назначение

Вычисляет центр дуги, созданной тремя точками.

Параметры

1. Значение декартовой координаты 1, значение декартовой координаты 2, значение декартовой координаты 3

Задаются три точки, определяющие дугу, и точка, определяющая конфигурацию робота.

2. Значение декартовой координаты для задания конфигурации

Задается точка, определяющая конфигурацию робота.

CSHIFT (transformation values 1, transformation values 2, transformation values 3, object transformation values BY shift amount)

Опция

Назначение

Вычисляет положение, которое было сдвинуто на определенную величину от положения объекта. Робот сдвигается по отношению к центру окружности, созданной при помощи трех определенных точек.

Параметры

1. Значение декартовой координаты 1, значение декартовой координаты 2, значение декартовой координаты 3

Задаются три точки, определяющие окружность.

2. Значение декартовой координаты для задания конфигурации

Задается положение объекта в координатных значениях.

3. Величина сдвига

Задается величина сдвига в действительных числах.

9.3 МАТЕМАТИЧЕСКИЕ ФУНКЦИИ

ABS	Вычисляет абсолютное значение числового выражения.
SQRT	Вычисляет квадратный корень числового выражения .
PI	Выдает константу π .
SIN	Вычисляет значение SIN (синус).
COS	Вычисляет значение COS (косинус).
ATAN2	Вычисляет значение Tg (тангенс)
RANDOM	Выдает случайное число.

ABS (real value)
SQRT (real value)
PI
SIN (real value)
COS (real value)
ATAN2 ((real value 1, real value 2)
RANDOM

Функция	Назначение	Пример
ABS	Выдает модуль числового выражения	
SQRT	Выдает корень квадратный	
PI	Выдает константу пи (3.1415...)	
SIN	Выдает синус данного угла	
COS	Выдает косинус данного угла	
ATAN2	Выдает тангенс угла	
RANDOM	Выдает случайное число от 0.0 до 1.0	

Примеры

`x=ABS(y)`

`x=SQRT(y)`

`en=2*PI*r`

`Z=(SIN(x) ^ 2)+(COS(y) ^ 2)`

`slope=ATAN2(rise,run)`

`r=RANDOM*10`

9.4 СТРОКОВЫЕ ФУНКЦИИ

\$CHR	Выдает ASCII символы указанного значения.
\$SPACE	Выдает заданное количество пробелов.
\$LEFT	Выдает левосторонние символы в строке.
\$RIGHT	Выдает правосторонние символы в строке.
\$MID	Выдает заданное количество символов.
\$DECODE	Выделяет символы, разделенные заданными символами.
\$ENCODE	Выдает строку, созданную при помощи данных печати.
\$ERRORS	Выдает сообщение об ошибке с специальным кодом ошибки.
\$ERROR ошибки.	Выдает сообщение об ошибке с отрицательным специальным кодом ошибки.
\$DATE	Выдает системную дату.
\$TIME	Выдает системное время в строке.

\$CHR (real value)

Назначение

Выдает ASCII цепочку символов соответственно определенным ASCII значениям.

Параметр

Реальное значение или числовое выражение.

Задаёт значение для того чтобы изменить его в ASCII код.

Допустимые пределы 0 – 255.

Пример

\$CHR(65)	Returns "A" for ASCII value 65.
\$CHR(^H61)	Returns "a" for ASCII value 97 (16×6+1)

\$SPACE (number of blanks)

Назначение

Выдает определенное число пробелов

Параметр

Количество пробелов

Задаёт число пробелов. 0 или положительное число

Пример

Type "a" + \$SPACE(1) + "dog"

На экране появится надпись "a dog".

\$LEFT (string, number of characters)

Назначение

Выдает определенное число символов, начиная с крайнего левого символа заданной цепочки.

Параметры

1. Строка

Символьная цепочка, строчная переменная или строчное выражение

2. Число символов

Определяет, как много символов нужно отсчитать и показать, начиная с крайнего левого, или первого символа введенной строки. Если ноль или отрицательное число заданы, вводятся пробелы. Если задано число, большее чем число символов строки, выводится полностью вся строка.

Пример

<code>\$LEFT("abcdefgh",3)</code>	Returns the string "abc".
<code>\$LEFT("*1*2*3*4*5",15)</code>	Returns "*"1*2*3*4*5" (the whole string).

\$RIGHT (string, number of characters)

Назначение

Выдает определенное число символов, начиная с крайнего правого символа заданной цепочки.

Параметры

1. Строка

Символьная цепочка, строчная переменная или строчное выражение

2. Число символов

Определяет, как много символов нужно отсчитать и показать, начиная с крайнего правого, или последнего символа введенной строки. Если ноль или отрицательное число заданы, вводятся пробелы. Если задано число, большее чем число символов строки, выводится полностью вся строка.

Выдает определенное число символов, начиная с крайнего левого символа заданной цепочки.

Пример

<code>\$RIGHT("abcdefgh",3)</code>	Returns the string "fgh".
------------------------------------	---------------------------

\$MID (string, real value, number of characters)

Назначение

Выдает заданное число символов из заданной строки.

Параметры

1. Строка

Символьная цепочка, строчная переменная или строчное выражение

2. Действительная величина

Задаёт позицию в строке, начиная с которой производится вывод информации.

3. Число символов

Задаёт число символов для извлечения.

Пояснение

Если начальная позиция не задана или задана 1 или меньшим числом, извлечение символов происходит с первого символа строки. Если начальная позиция задана нулем или отрицательным числом, или число символов больше символов строки, выводится

пробел. Если число символов не задано, или когда оно больше числа символов строки, извлечение начинается с позиции (real value) и до конца строки.

Пример

```
$substring=$MID("abcdef",3,2)
```

Функция \$MID выводит "cd" символы (два символа, начиная с третьего из строки) из строки "abcdef", затем результат присваивается строковой переменной \$substring

\$DECODE (string variable, separator character, mode)

Назначение

Выдает строку, отделенную «разделительным символом».

Параметры

1. Строчная переменная

Определяет строку, из которой берутся символы. Символы, извлеченные при помощи этой функции, перемещаются из этой строки.

2. Разделительный символ

Определяет символ, который читается как разделительный знак. Любой символ в строке может быть определен как разделительный знак.

3. Режим

Определяет реальное число для действий, осуществляемых этой функцией.

Если число отрицательное или 0, или не задано, символы начинаются от первого символа в строчной переменной до разделительного знака. Выделенная строка перемещается из строчной переменной. Если число положительное, первый разделительный знак, который появляется в строке, выделяется. Выделенный разделительный знак перемещается из значения строчной переменной. Если более одного разделительного знака, символы появляются в строке постепенно, все разделительные знаки выделяются и перемещаются из строчной переменной.

Пояснение

Эта функция ищет указанную строку для разделительного символа и извлекает символы от начала строки до разделительного знака. Извлеченные символы выделяются, как результат этой функции и в то же самое время они перемещаются из первоначальной строки.

Строка, выделенная как результат функции (строка, удаленная из первоначальной строки) могла быть или символами перед разделителем или разделителем непосредственно.

(ПРИМЕЧАНИЕ)

Эта функция изменяет первоначальную строку, в то время, когда она выделяет символы. Символ разделителя не зависит от регистра.

Пример

В примере, показанном ниже, числа, разделенные запятыми или пробелами, перемещаются из строки "\$input". Первая инструкция в DO структуре перемещает от первых установленных символов \$input и заменяет их затем в переменной "\$temp". Следующая функция VAL изменяет строку, увеличенную в предыдущей инструкции в реальное значение. Реальное значение затем заменяется в массиве переменных "value". Затем программное выполнение переходит к следующей функции \$DECODE и разыскивает следующий разделительный знак.

```
i=0                                ;Resets counter
DO
  $temp=$DECODE($input,"",0)      ;Extracts characters up to the separator","
  value[i]=VAL($temp)             ;Converts the characters to real values.
  if $input == " " GOTO 100
  $temp = $DECODE($input,"",1)    ;Extracts the separator ","
  i=i+1                            ;Counter increment
  UNTIL $input=="                 ;Continues program execution until there are no
  100 TYPE "END"                  more characters
```

Если значения \$input выглядят, как следующие

1234, 93465.2, .4358, 3458103,

value[0]	1234.0
value[1]	93465.2
value[2]	0.4358
value[3]	3458103.0

Как результат выполнения программы, значение строковой переменной \$input становятся "" (или пробелом).

\$ENCODE (print data, print data,)

Назначение

Выдает строку, созданную из данных печати, определенных в параметрах. Строка, созданная таким образом, аналогична, как и при выполнении команды TYPE.

Параметр

Данные печати

Выбирается один или более. Разделяются данные запятой, когда задано больше одного.

- (1) символьная строка
- (2) реальное значение выражения (значение вычисляется и отображается)
- (3) Информация форматирования (управляет форматом вывода сообщения)

Пояснение

Эта функция разрешает создание строк внутри программы, используя те же самые данные, что и в команде TYPE. В отличие от TYPE, \$ENCODE функция не отображает созданные строки, но вместо этого использует результаты в программе как величины.

Следующие коды используются, чтобы задать выходной формат для числовых выражений. Используется один и тот же формат, до последующего изменения. В любом формате, если значение гораздо больше, чем может быть отображено, в строке появляются (*).

/D – используется по умолчанию. Это то же самое, что задание формата /G15.8, за исключением того, что нули, следующие за числовыми выражениями и все пробелы между числовыми значениями удаляются.

/Em.n – выражает числовое значение в экспоненциальном виде (в виде мантиссы и порядка т.е. –1.234E+02). “m” описывает общее число символов, показанных на терминале, “n” – число десятичных разрядов. “m” должно быть больше “n” на шесть или более и меньше чем 32.

/Fm.n – выражает числовое значение с фиксированной точкой (т.е. –1.234). “m” – описывает общее количество символов, “n” – число десятичных разрядов.

/Gm.n – если значение больше чем 0.01 и может быть выражено в формате Fm.n внутри “m” цифр, величина выражается в данном формате. Иначе величина выражается в Em.n формате.

/Hn – значение выражается шестнадцатичным числом в “n” цифровом поле.

/In – значение выражается десятичным числом в “n” цифровом поле.

Следующие параметры используются для вставки определенных символов между символьными цепочками.

/Cn – вставляет перевод строки n раз в месте, где этот код введен в начале или после данных печати. Если этот код расположен внутри данных печати, n-1 пустых строк вставляются.

/S – линия не вводится

/Xn – вставляется n пробелов.

/Jn – выражается величина в шестнадцатичном виде в n цифровом поле. Ноли используются в месте пробелов (опция).

/Kn – выражается величина в десятичном виде в n цифровом поле. Ноли используются в месте пробелов (опция).

/L – это то же самое что /D за исключением, что все пробелы перемещаются с этим кодом (опция).

Пример

```
$output = $output + $ENCODE(/F6.2,count)
```

Величина действительной переменной “count” преобразуется в строку в формате, заданном /F6.2 и добавляется в конце строки “\$output”. Затем комбинированная строка заменяется назад в строковой переменной “\$output”.

\$ERRORS (error code)

Назначение

Выдает сообщение об ошибке с определенным кодом ошибки. Код ошибки выдается, как символьная цепочка с сообщением об ошибке.

Параметр

Код ошибки

Определяет код ошибки в следующем формате: Rxxxx, Wxxxx, Exxxx, Dxxxx.

\$ERROR (error number)

Назначение

Выдает сообщение об ошибке с определенным кодом ошибки.

Параметр

Номер ошибки

Задаёт номер ошибки отрицательным числом. Коды ошибки преобразуются в отрицательные номера ошибок.

```
Dxxxx : -4xxxx  
Exxxx : -3xxxx  
Wxxxx : -2xxxx  
Rxxxx : -1xxxx
```

\$DATE (date form)

Назначение

Выдает системную дату в определенном строчном формате.

Параметр

Форма даты

Задаётся номерами 1 – 3, данные формата для вывода.

1 - \$DATE(1) mm/dd/yyyy

Если дата 10 июля 2002 года, значение выводится в виде 07/10/2002

2 - \$DATE(2) dd/mmm/yyyy

Значение выводится в виде 10/JUL/2002

3 - \$DATE(3) yyyy/mm/dd

Значение выводится в виде 2002/07/10

\$TIME

Назначение

Выводит системное время в следующем формате строки
hh:mm:ss

Пример

18:27:50

Часы выражаются в 24 часах

10. ПРОГРАММЫ УПРАВЛЕНИЯ ПРОИЗВОДСТВЕННЫМ ПРОЦЕССОМ

Эта глава описывает мониторные команды и программные инструкции, используемые с программами управления производственным процессом (РС программами). В скобках справа М – указывает мониторную команду, Р – программную инструкцию. Поэтому и М и Р могут быть использованы либо как команды, либо как инструкции.

PCSTATUS	Отображает статус определенной РС программы. (М)
PCEXECUTE	Выполняет заданную РС программу. (М, Р)
PCABORT (М, Р)	Мгновенно останавливает выполнение заданной РС программы.
PCEND	Заканчивает выполнение заданной РС программы. (М, Р)
PCCONTINUE	Продолжает выполнение прерванной РС программы. (М)
PSKILL	Инициализирует стек выполнения РС программ. (М)
PCSTEP	Выполняет один шаг РС программы (М)
PCSCAN	Задаёт время обработки РС программы (опция) (Р)

Пример

Ключевое слово	параметр
PCSTATUS	program number

Параметр, выделенный цветом можно опустить.
Всегда вставляйте пробел между ключевым словом и параметром

PCSTATUS PC program number:

Назначение

Отображает состояние PC программ (мониторная команда) (M)

Параметр

Номер PC программы

Выбирает номер программы для отображения. Допустимые границы от 1 до 5. Если параметр не задан, устанавливается 1.

Пояснение

Состояние PC программ отображается в следующем формате.

```
(1) ..... PC status:      Program is not running
           ..... Execution cycles:
(2) ..... Completed cycles: 11
(3) ..... Remaining cycles: Infinite
(4) ..... Program name      Prio  Step No.
(5) ..... pc_test0          1      PRINT "step1"
```

(1) Состояние программы описывает следующее:

Программа не выполняется. Программа, в настоящее время, не выполняется.

Программа выполняется. Программа, в настоящее время выполняется.

Программа ожидает. Программа выполняется, но ожидает наступления условия, заданного WAIT командой.

(2) Завершенные циклы.

Отображает число выполненных циклов.

(3) Оставшиеся циклы.

Отображает количество циклов до сих пор не выполненных. Если выполнение циклов задано отрицательным числом (-1) в PCEXECUTE команде, на дисплее будет надпись "infinite"

(4) Программное имя

(5) Шаг

Отображает номер выполняемого шага и выполняемую в настоящее время инструкцию.

PCEXECUTE PC program number: program name, execution cycle, step number

Назначение

Выполняет PC программы (M, P)

Параметры

1. Номер PC программы

Задаёт номер программы для выполнения. Допустимые пределы 1 – 5. Если параметр не задан, устанавливается 1. Вплоть до 5 программ могут быть выполнены в одно и то же время.

2. Имя программы

Задаёт имя программы для выполнения в номере РС программы. Если параметр не задан, выполняется последняя программа, заданная этой инструкцией.

3. Выполнение цикла

Задаёт, сколько раз выполняется РС программа. Если параметр не задан, выполняется один раз. Если задана –1 выполняется бесконечное число раз.

4. Номер шага

Задаёт шаг, с которого начинается программное выполнение. Если параметр не задан, выполнение начинается с первого шага программы.

Пояснение

Эта команда идентична мониторной команде EXECUTE, за исключением того, что она запускает выполнение РС программ вместо управляющих программ робота.

Отображаемое мерцание звездочки «*» в конце имени, говорит о том, что выполняется РС программа.

PCEXECUTE может быть использована, как мониторная команда, так и как программная инструкция.

Пример

```
PCEXECUTE control, -1
```

Программа “control” выполняется непрерывно до тех пор, пока не выполнится команда PCABORT, или инструкции PAUSE, HALT изнутри программы, или пока не произойдет ошибка.

PCABORT PC program number

Назначение

Прерывает выполнение РС программы. (M,P)

Параметр

Номер РС программы

Задаёт номер РС программы, для ее прерывания. Допустимые пределы от 1 до 5. Если параметр не задан, устанавливается 1.

Пояснение

PCABORT идентична команде ABORT, за исключением того, что она прерывает выполнение РС программ вместо управляющих программ робота.

Программное выполнение может быть продолжено при помощи команды PCCONTINUE.

PCABORT может использоваться и как мониторная команда, и как программная инструкция.

PSKILL PC program number

Назначение

Инициализирует стек PC программ (мониторная команда)

Параметр

Номер PC программы

Задаёт номер PC программы для инициализации. Допустимые пределы 1 – 5.

Если параметр не задан, устанавливается 1.

Пояснение

Эта команда инициализирует стек PC программ

Когда программа временно прерывается командами PAUSE или PCABORT, или ошибкой, программа сохраняется в программном стеке. До тех пор пока программа находится в стеке, ее невозможно будет удалить. В этом случае, при первом использовании PSKILL, переместите программу из стека.

PCEND PC program number: task number

Назначение

Прекращает выполнение PC программы, в настоящее время запущенной, при выполнении следующей STOP инструкции в этой программе (используется как мониторная команда, так и программная инструкция).

Параметры

1. Номер PC программы

Задаёт номер PC программы для прекращения выполнения. Допустимые пределы 1 – 5.

Если параметр не задан, устанавливается 1.

2. Номер задачи

Определяет 1 или – 1. Если параметр не задан, устанавливается 1.

Пояснение

Если номер задачи не определен или задан как 1, программное выполнение останавливается, как только выполняются следующие инструкции STOP, RETURN, не обращая внимания на оставшиеся циклы. Оставшиеся циклы могут выполняться при использовании PCCONTINUE.

Если –1 определена, как номер задачи, команда PCEND, введенная предварительно, отменяется. Когда программный цикл совершается или программа выполняется бесконечно без использования инструкции STOP, PCEND неэффективна и должна быть отменена при помощи PCEND –1. Для отмены цикла должна быть использована команда PCABORT.

PCEND может использоваться и как мониторная команда, и как программная инструкция.

PCCONTINUE PC program number NEXT

Назначение

Возобновляет выполнение приостановленной PC программы. Или перескакивает инструкцию WAIT в PC программе.

Параметры

1. Номер PC программы

Задаёт номер PC программы для продолжения выполнения. Допустимые пределы 1 – 5. Если параметр не задан, устанавливается 1.

2. Следующий

Если параметр задан, выполнение возобновляется с шага, следующего после шага, где программа была прервана. Если параметр не задан, выполнение программы возобновляется с шага прерывания. Эта команда с параметром NEXT может быть использована для перепрыгивания инструкции WAIT в приостановленной в настоящее время PC программе, и возобновления выполнения этой программы.

Пояснение

PCCONTINUE идентичнао CONTINUE команде, за исключением того, что эта команда используется для продолжения PC программы, а не управляющих программ робота. Выполнение продолжается с шага, где выполнение было остановлено при помощи команд PAUSE или PCABORT, или из-за ошибки, и со следующего шага, если задан параметр NEXT.

PCSTEP PC program number: program name, execution cycles, step number

Назначение

Выполняет единичный шаг PC программы (мониторная инструкция).

Параметры

1. Номер PC программы

Задаёт номер PC программы, содержащей желательный шаг. Допустимые пределы от 1 до 5. Если параметр не задан, устанавливается 1.

2. Программное имя

Задаёт имя программы для выполнения в этом номере PC программы. Если параметр не задан, выбирается текущая программа или выбранная последней.

3. Выполнение цикла

Задаёт, как много раз программный шаг будет выполняться. Если параметр не задан, выполняется один раз.

4. Номер шага

Задаёт номер программного шага для выполнения. Если параметр не задан, выполняется первый шаг.


Если никакой параметр не задан, выполняется следующий шаг последней программы.

Пояснение

PCSTEP команда, подобно команде PCCONTINUE, может использоваться без параметров в следующих случаях:

1. Когда PCSTEP команда была использована в последнем выполненном шаге
2. После инструкции PAUSE
3. Когда программа была приостановлена по другим причинам, чем ошибочное состояние.

Пример

```
>PCSTEP sequence,,23 
```

Выполняет шаг 23 PC программы №1, названной “sequence” один раз.

Вводя PCSTEP  после этого, выполняется следующий шаг (24).

PCSCAN time

Опция

Назначение

Устанавливает время цикла для выполнения PC программы (программная инструкция).

Параметр

Время

Устанавливает, как долго повторяются программные циклы. Время задается в секундах от 0 и более.

Пояснение

Эта команда используется для выполнения PC программы в определенном временном цикле. Если время выполнения больше заданного времени, время, заданное здесь игнорируется.

Пример

```
program  
  PCSCAN 1  
  IF sig(1) THEN  
    SIGNAL -1  
  ELSE  
    SIGNAL 1  
  END
```

Если программа выполняется непрерывно, используя команду PCEXECUTE (выполняемый цикл –1), выходной сигнал 1 включается, выключается каждую секунду.

11. ПРИМЕРЫ ПРОГРАММ

Эта глава показывает примеры некоторых программ, используемых в системе AS языка.

11.1 Первоначальные установки для программ

11.2 Паллетирование

11.3 Внешнее взаимодействие

11.4 Преобразование инструментальной системы координат

11.5 Относительные позиции

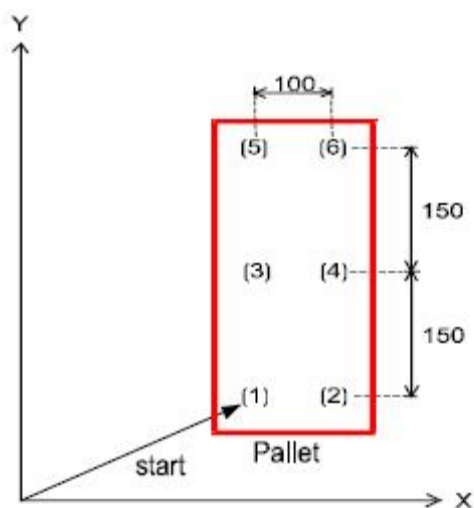
11.6 Использование относительных позиций с функцией FRAME

11.7 Установка конфигураций робота

11.1 ПЕРВОНАЧАЛЬНЫЕ УСТАНОВКИ ДЛЯ ПРОГРАММ

Для более простого программирования, параметры настройки, показанные ниже, делаются до выполнения любой функции на роботе.

- Перемещают робот в домашнюю позу (позиция и конфигурация).
- Определяют необходимые переменные для каждой задачи (например для паллетирования, устанавливают количество деталей на поддоне)
- Инициализируют счетчик, флажок, и т.д.
- Устанавливают инструментальную систему координат, чтобы использоваться в этой задаче.
- Устанавливают базовую систему координат, чтобы использоваться в этой задаче.



В примере, показанном выше, детали укладываются на поддон в порядке от (1) до (6). В этом случае программа, подобная показанной ниже будет использоваться для первоначальных установок. В этом примере, поддон параллелен базовым координатам робота.

1 BASE NULL	;задает нулевую базовую систему координат робота (NULL)
2 TOOL tool1	;задает инструментальную систему координат (tool1)*
3 row.max=3	;3 ряда
4 col.max=2	;2 столбца
5 xs=100	;устанавливает известное расстояние по координатной оси X ($\Delta X=100\text{mm}$)
6 ys=150	;устанавливает известное расстояние по координатной оси Y ($\Delta Y=150\text{mm}$)
7 POINT put=start	;присваивает значение позиции (1) в переменную put.
8 OPENI	;открывает схват инструмента
9 HOME	;перемещается в позицию home**

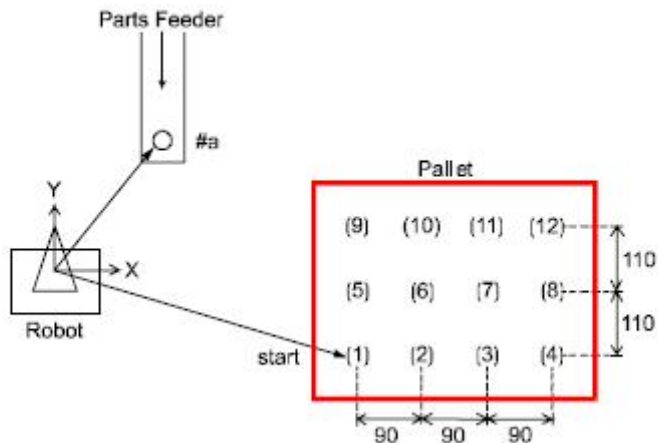
ПРИМЕЧАНИЕ* Значение координаты (tool 1) должны быть задано

ПРИМЕЧАНИЕ** Положение HOME должно быть предварительно задано

11.2 ПАЛЛЕТИРОВАНИЕ

В примере, показанном здесь, детали берутся с подающего лотка и помещаются в поддон в три ряда (на расстоянии 110 мм) и четыре столбца (на расстоянии 90 мм). Для того чтобы упростить объяснение, и поддон и детали, помещаемые в поддон, установлены параллельно базовой системе координат робота.

Также, процедуры синхронизации подающего лотка и робота, при использовании внешних сигналов ввода - вывода (команда SWAIT, команда SIGNAL, и т.д.) опущены.



- Паллета установлена параллельно XY плоскости базовой системы координат.
- Позиция #a (Деталь на подающем лотке) и позиция “start” (где кладется первая деталь) обучаются заранее до выполнения программы.

Пример программы

```
.PROGRAM palletize
;      initial setting (3 rows, 4 columns, ΔX=90, ΔY=110, etc.)
row.max=3
col.max=4
xs=90
ys=110
SPEED 100 ALWAYS
ACCURACY 100 ALWAYS
POINT put=start
OPENI
```

```
;
;      Start palletizing
FOR row=1 TO row.max
FOR col=1 TO col.max
JAPPRO #a,100
SPEED 30
ACCURACY 1
LMOVE #a
CLOSEI
LDEPART 200
;
JAPPRO put, 200
SPEED 30
ACCURACY 1
LMOVE put
OPENI
LDEPART 200
;

;      Calculate the pose of part in the next row.
POINT put=SHIFT(put BY xs, 0,0)
END
;

;      Calculate the pose of part in the next column.
POINT put=SHIFT(start by 0,ys*row, 0)
END
.END
```

} Берет деталь с подающего лотка

} Кладет деталь на паллету

11.3 ВНЕШНЕЕ ВЗАИМОДЕЙСТВИЕ

Этот пример демонстрирует работу, выполняемую синхронно с внешними устройствами. Эта программа использует команды: SIGNAL, IF, SWAIT, ONI, IGNORE.

1. Два типа деталей, А и В, установлены на подающем лотке в случайном порядке.
(Входной сигнал для завершения установки детали на подающем лотке: IN1)
2. Робот поднимает деталь с подающего лотка и устанавливает ее в станцию тестирования
(Выходной сигнал для начала тестирования: OUT1)
3. На станции тестирования, детали классифицируются в деталь А, деталь В или в другую, отличную от А и В.

Входной сигнал для завершения тестирования: IN2

Входной сигнал для классификации деталей: IN3, IN4

(IN3, IN4) = (1, 0): деталь А

(IN3, IN4) = (0, 1): деталь В

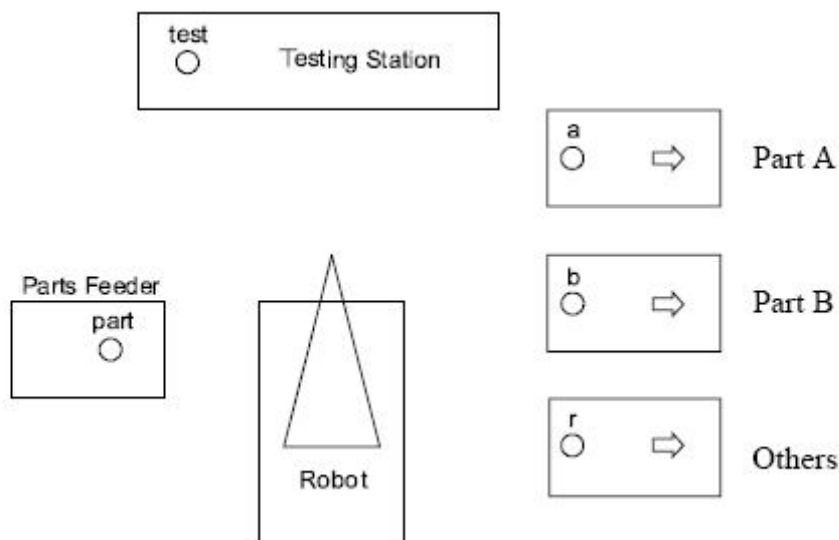
(IN3, IN4) = (0, 0) или (1, 1): Другие

4. Робот размещает детали согласно классификации каждой детали.

Если любая неисправность возникает со станцией тестирования, в то время как робот поднимает деталь с подающего лотка и переносит ее к станции, программа немедленно останавливается и выполняет переход к подпрограмме обработки неисправности.

Внешний входной сигнал при возникновении неисправности - IN7. Входной сигнал IN6 поступает, когда неисправность устранена, и робот, как только получает этот сигнал, продолжает выполнение программы.

Главная программа для выполнения вышеупомянутой задачи называется MAIN, подпрограмма обработки неисправностей называется EMERGENCY



Пример программы

```

; Define Variables
set.end =1001                ; деталь на месте взятия на подающем лотке
test.end =1002               ; тестирование детали завершено
a.part =1003                 ; деталь а
b.part =1004                 ; деталь в
retry =1006                  ; неисправность устранена
fault=1007                   ; возникновение неисправности
test.start= 1                ; сигнал начала тестирования

.PROGRAM main()
OPENI
10 JAPPRO part,100
ONI fault CALL emergency     ; контроль сигнала неисправности и переход к подпрограмме
                              ; обработки неисправности

SWAIT set.end                ; деталь на месте взятия на подающем лотке
LMOVE      part              ; Движение к детали (на лотке)
CLOSEI
LDEPART    100
JAPPRO     test,100          ; перенос детали к станции тестирования
LMOVE      test
BREAK
;
IGNORE     fault ;stops ;    ; останавливает контроль сигнала неисправности
SIGNAL     test.start        ; включает сигнал тестирования детали
TWAIT 1.0
                              ; ожидает завершения тестирования
SWAIT test.end
JDEPART    100
SIGNAL     -test.start       ; выключает сигнал тестирования
IF SIG(a.part,-b.part) GOTO 20 ; классифицирует детали и переходит к определенной метке, если
IF SIG(-a.part,b.part) GOTO 30 ; деталь А – к метке 20, если деталь В - к метке 30
POINT n=r
GOTO 40                ; если деталь ни А, ни В переход к метке 40
20 POINT    n=a         ; задание точки, для того чтобы положить деталь А
GOTO 40
30 POINT    n=b         ; задание точки, для того чтобы положить деталь В
40 JAPPRO    n,100       ; переход к позиции над местом расположения детали

```

```
LMOVE      n  
OPENI
```

```
LDEPART    100  
GOTO 10
```

```
.END
```

```
.PROGRAM emergency()
```

```
PRINT "***ERROR***"
```

```
;выводит сообщение об ошибке на экран  
;ожидает сигнал о сбросе неисправности
```

```
SWAIT retry
```

```
ONI  fault CALL emergency
```

```
; устанавливает контроль наличия неисправности  
;возврат в головную программу
```

```
RETURN
```

```
.END
```

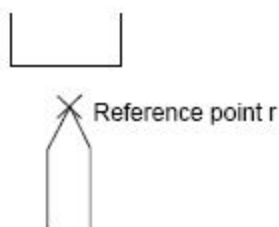
11.4 ПРЕОБРАЗОВАНИЕ ИНСТРУМЕНТАЛЬНОЙ СИСТЕМЫ КООРДИНАТ

Эта глава описывает, как получить значения для преобразования инструментальной системы координат и как создавать программы, которые их используют.

11.4.1 ЗНАЧЕНИЕ ПРЕОБРАЗОВАНИЯ ИНСТРУМЕНТАЛЬНОЙ СИСТЕМЫ КООРДИНАТ 1 (КОГДА РАЗМЕР ИНСТРУМЕНТА НЕИЗВЕСТЕН)

Когда размер инструмента неизвестен из-за сложности инструмента, значения для преобразования инструментальной системы координат могут быть рассчитаны, как показано ниже. Ось Z базовых координат установлена перпендикулярно к основанию {земле}.

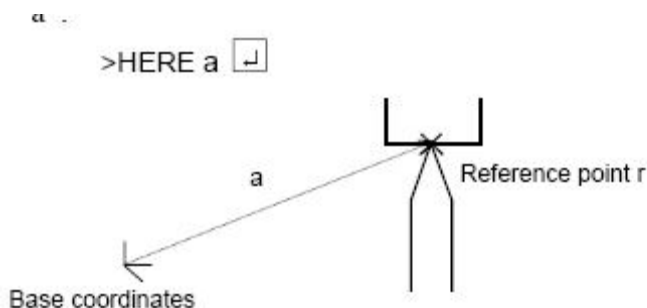
1. Выберите объект с острым наконечником (точкой). Установите верхушку объекта острием вертикально вверх по отношению к основанию. Это будет контрольная точка "r".



2. Переместите робот, так чтобы поверхность монтажного фланца робота была повернута вниз, лицом по отношению основания. Затем в мониторном режиме введите следующие команды:

```
>SPEED 10 ☐
>TOOL NULL ☐ ;устанавливает инструментальную систему координат как
                     ;нулевую
>DO ALIGN ☐ ;выравнивает ось Z с осью Z базовой системы координат
```

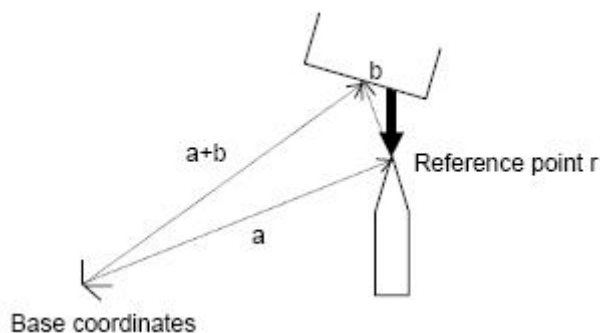
3. Используя режим базовой системы координат на пульте ручного управления переместите робот так, чтобы центр фланца стал перпендикулярен контрольной точке. Перемещайте робот только вдоль осей X, Y, Z базовой системы координат. Затем введите команду, как показано ниже, чтобы присвоить координатное значение позиции в переменную "a" (присвоить имя координате):



4. Установите инструмент на фланце, и переместите центральную точку инструмента (TCP) в контрольную точку так, чтобы ось Z новой инструментальной системы координат была перпендикулярна базовым координатам X, Y.

Для того чтобы обучить значение преобразования в позиции как сложную координату “a+b”, введите следующую команду :

>HERE a+b



5. Из этого значения сложной координаты, значение для преобразования инструментальной системы координат может быть найдено как “-b”.

Введите:

>POINT t=-b

Эта команда присваивает значение $-b$ в значение переменной t .

6. Задайте преобразование инструментальной системы координат как t .

>TOOL t

7. Для того чтобы проверить, введите следующее:

>DO JMOVE r

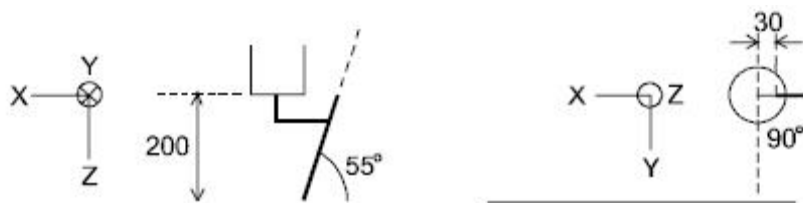
Точка центра инструментальной системы координат должна подойти к контрольной точке r .

После задания новой инструментальной системы координат, все действия основываются на этом преобразовании инструмента, пока инструмент не будет заменен.

11.4.1 ЗНАЧЕНИЕ ПРЕОБРАЗОВАНИЯ ИНСТРУМЕНТАЛЬНОЙ СИСТЕМЫ КООРДИНАТ 2 (КОГДА РАЗМЕР ИНСТРУМЕНТА ИЗВЕСТЕН)

Когда размер инструмента известен, значения для преобразования инструментальной системы координат могут быть рассчитаны, как показано ниже.

Значения, определенные этой процедурой, вообще более точны, чем полученные в прежней процедуре. (См. 11.4.1)



XYZ оси, в вышеупомянутом рисунке, выражают нулевую инструментальную систему координат. Следующая процедура устанавливает начало инструментальной системы координат в верхушке сварочной горелки и оси Z в том же самом направлении, что и направление горелки.

(1) Определите переменную преобразования инструмента "torch", используя команду POINT:

```
>POINT torch 
X      Y      Z      O      A      T
0      0      0      0      0      0

Change
>-30, 0, 200, 0, 35, 0 
X      Y      Z      O      A      T
-30    0      200    0      35      0

Change 
>
```

(2) Установите значение преобразования инструментальной системы координат, используя переменную "torch".

```
>TOOL torch 
```

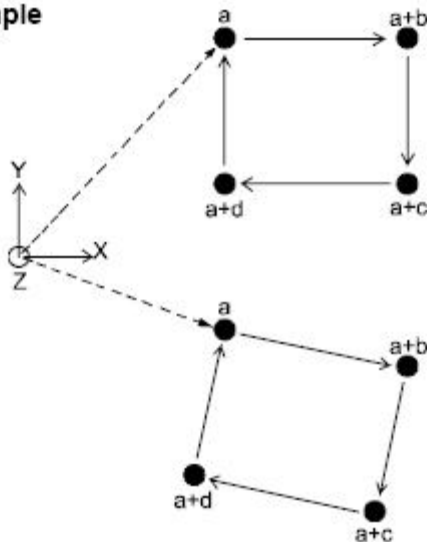
11.5 ОТНОСИТЕЛЬНЫЕ ПОЗИЦИИ

11.5.1 ИСПОЛЬЗОВАНИЕ ОТНОСИТЕЛЬНЫХ ПОЗИЦИЙ

Позиция может быть определена относительно контрольной точки. Когда определено таким способом, отношение между этой позицией и контрольной точкой остается постоянным, даже если контрольная точка переопределяется.

Например, когда обучаются четыре угла поверхности, позиционная зависимость между роботом и поверхностью изменяются в зависимости от того, куда они помещены, но пока форма поверхности остается той же самой, зависимость между этими четырьмя углами не изменяется. Поэтому, если один из углов обучается, как контрольная точка для того, чтобы определить абсолютную позиционную зависимость между роботом и поверхностью, и другие три угла обучаются относительно первого угла, тогда, когда поверхность перемещена, только контрольная точка должна быть переопределена.

Example



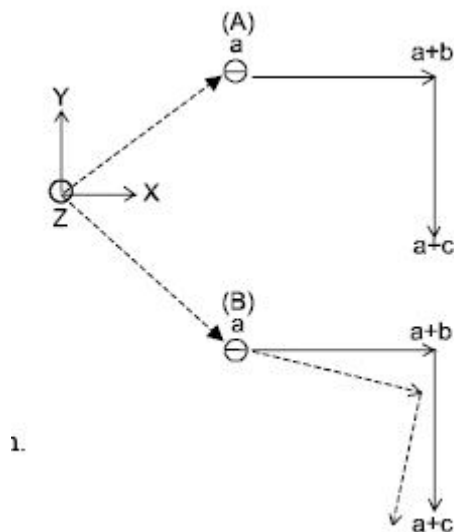
Teaching

>HERE a ☐
>HERE a+b ☐
>HERE a+c ☐
>HERE a+d ☐

Program

```
JMOVE a
LMOVE a+b
LMOVE a+c
LMOVE a+d
LMOVE a
```

(A)



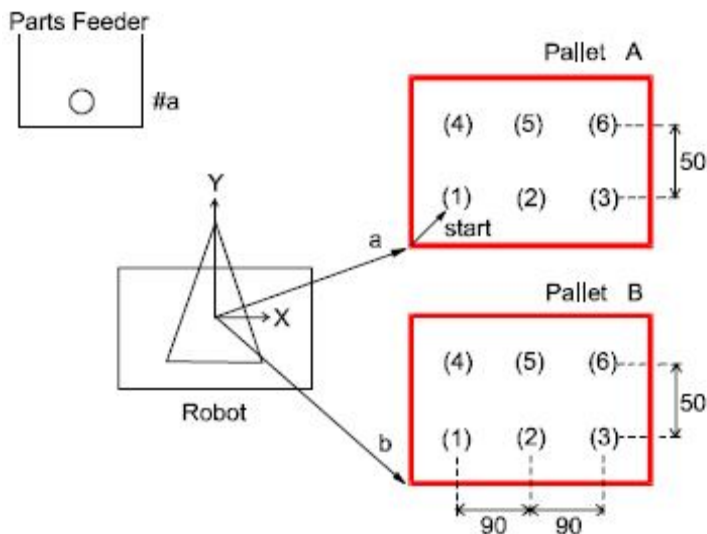
1.

На рисунке (А) вверху, обучаются контрольная точка а, и сложные координаты для других углов. Затем, на рисунке (В) контрольная точка переопределяется. Если конфигурация запястья робота вновь не установлена (то есть сохраняется в том же самом положении, как это было в (А)), робот будет двигаться по траектории, показанной сплошной линией. Если робот, как предполагается, проходит пунктир, необходимо переопределить конфигурацию так же как позицию.

11.5.2 ПРИМЕР ПРОГРАММЫ ПРИ ИСПОЛЬЗОВАНИИ ОТНОСИТЕЛЬНЫХ ПОЗИЦИЙ

В этом примере, детали раскладываются как в предыдущем примере, за исключением, что используется два поддона.

Поддоны помещены отдельно, но зависимость между контрольной точкой и местами, где детали должны быть размещены, одинаковы на любом из поддонов. Эта программы выполняет установку детали с подающего лотка на поддон А. После того, как шесть деталей установлены, робот продолжает делать то же самое, устанавливая детали на поддон В. (процедура синхронизации с подающим лотком опущена).



Позиции для обучения

#a : позиция, где робот берет детали с подающего лотка

a : контрольная точка на поддоне А

b : контрольная точка на поддоне В

start : позиция первой детали на поддоне по отношению к контрольной точке

Пример программы

```
.PROGRAM relative.test
      ; начальные установки (2 ряда, 3 столбца, ΔX=90, ΔY=50, etc.)
row.max=2
col.max=3
xs=90
ys=50
OPENI
flg=0 ; flg=0 : Pallet A, flg=1 : Pallet B
POINT pallet=a
      ; начало пакетирования деталей
10 POINT put=start
FOR row=1 TO row.max
FOR col=1 TO col.max
JAPPRO #a,100
LMOVE #a                      берет деталь с подающего лотка
CLOSEI
LDEPART 100
;
POINT put_pt=pallet+put
JAPPRO put_pt,200
LMOVE                          кладет деталь на поддон
OPENI
LDEPART 200
;
POINT put=SHIFT(put BY xs,0,0)  определяет место для детали в следующем столбце
END
;
POINT put=SHIFT(start BY 0,ys*row,0) ; определяет место для детали в следующей строке
END
;
IF flg<>0 GOTO 30                ; переходит к финишной процедуре, когда поддон B заполнен
                                  (flg=1)

flg=1
POINT pallet=b                  ;определяет контрольную точку на поддоне B
GOTO 10
30 TYPE "*** end ***"
STOP
.END
```

11.6 ИСПОЛЬЗОВАНИЕ ОТНОСИТЕЛЬНЫХ ПОЗИЦИЙ С ФУНКЦИЕЙ FRAME

В примере в 11.5.1, положение запястья должно было быть исправлено при переопределении контрольной позиции. Это не является необходимым, если используется функция FRAME. Обучите четыре точки (b, c, d, e), чтобы определить значения преобразования рамки a; b и c определяют направление X оси, третья точка d определяет координатную плоскость XY, и точка e начало координат. После того, как точки обучены, введите следующую команду:

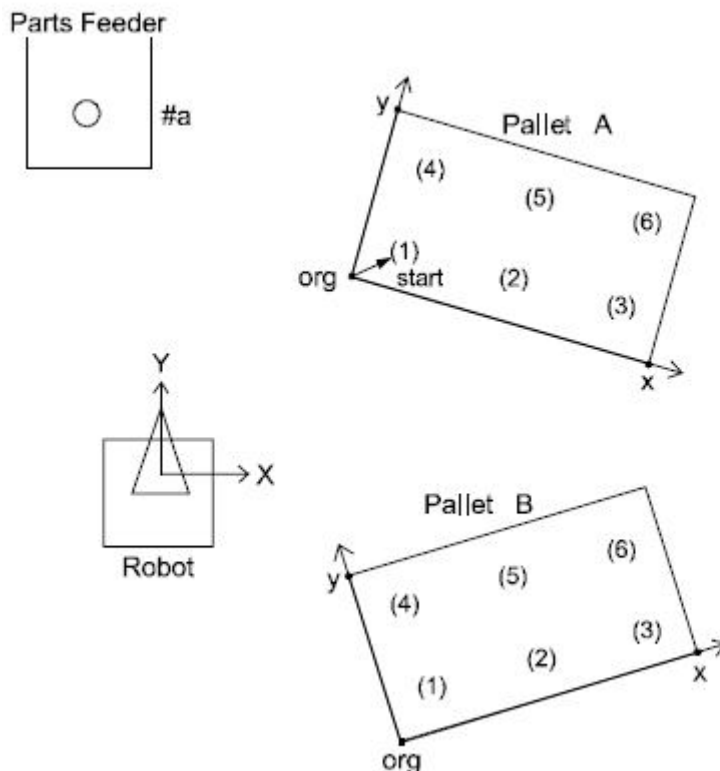
```
POINT a=FRAME (b, c, d, e)
```

Тогда, относительные координаты определятся, как переменная "a". Значения XYZ показывают положение начала относительных координат, и значение OAT показывает конфигурацию относительных координат в пространстве.

После этого, все позиции по отношению к относительным координатам могут быть выражены как позиция a + \square . Если изменится расположение поддона, переобучите b, c, d, e снова в контрольных точках (которые можно пометить при первом обучении) и переопределите тем же самым способом как выше.

Относительные координаты задаваемые, при использовании функции FRAME, также называют координатной системой отсчета.

В следующей типовой программе, выполняется та же самая операция как в 11.5.2, используя относительные координаты.



Первая процедура должна раскладывать детали на поддоне А. Обучаются три угла поддона, один как начало координат, другой как точка на оси X, и другой как точка на оси Y (см. рисунок выше). Выполните программу, показанную ниже, для того чтобы разложить детали на поддоне А (обратите внимание, что после того как точки определены, остальная часть программы, та же самая, как предыдущая типовая программа).

Для того чтобы разложить детали на поддоне В, повторно обучите эти три угла, и выполните ту же самую программу. Относительные координаты будут переопределены, и детали будут разложены на поддоне В, как на поддоне А.

Пример программы

```
.PROGRAM frame.test
; начальные установки (2 ряда,3 столбца, ΔX=90, ΔY=50,etc.)
row.max=2
col.max=3
xs=90
yx=50
OPENI
;
POINT pallet=FRAME(org,x,y,org) ;задает относительные координаты поддона
; (3 точки: для начала координат, для X/Y осей)
POINT put=start ; Начало пакетирования деталей
FOR row=1 TO row.max
FOR col=1 TO col.max
JAPPRO #a,100
LMOVE #a
CLOSEI ;Поднимает деталь с подающего лотка
LDEPART 100
;
POINT put pt=pallet+put
JAPPRO put pt,200
LMOVE ;кладет деталь в место pt на поддоне
OPENI
LDEPART 200
;
POINT put=SHIFT(put BY xs,0,0) ;определяет куда положить деталь в следующем столбце
END
;
POINT put=SHIFT(startBY 0,yx*row,0) ; определяет куда положить деталь в следующей строке
END
```

11.7 УСТАНОВКА КОНФИГУРАЦИЙ РОБОТА

Большинство роботов имеет шесть суставов, и когда позиция обучается, используя угловые координаты, значения смещения каждого из шести суставов даются, так что позиция робота определена однозначно. С другой стороны, когда позиция определена, используя значения декартовых координат, бывают случаи, когда в зависимости от конфигурации руки робота, может быть больше чем одно расположение суставов в позиции, заданной значением одной декартовой координаты. В AS системе, в основном, робот сохраняет конфигурацию предыдущего действия, так что изменения в конфигурации робота не являются необходимыми. Однако в следующих случаях конфигурация робота должна быть определена командой задания конфигурации:

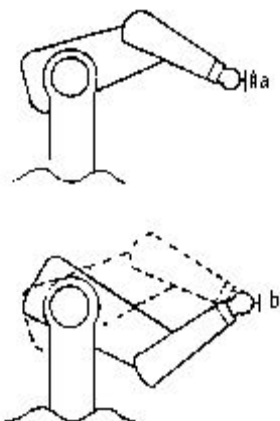
1. Когда робот перемещается из точки с непонятной конфигурацией в точку, заданную в декартовых координатах.
2. Когда 5-ый сустав (сустав изгиба) проходит через начало координат (0°) в SBS конфигурации (SBS: поворот, изгиб, поворот)

ABOVE, BELOW

Например, на рисунке ниже, если позиция #a определена конфигурацией локоть вверх, тогда результат команды JMOVE b будет локоть вверх (пунктир на рисунке), даже если позиция b была задана, как локоть вниз.

JMOVE #a

JMOVE b



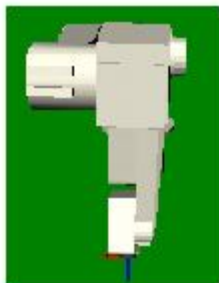
Тем же самым способом, если #a определена - угол запястья положительный UWRIST ($JT5 > 0$), конфигурацией в позиции b будет UWRIST независимо из конфигурации, когда точка b обучалась.

Чтобы решить эти проблемы, необходимо изменить конфигурацию робота, в то время как он находится в движении. Делайте это, выполняя команду конфигурации всякий раз, когда траектория заканчивается точкой, обученной в декартовых координатах, но движение выполняется в инструкциях интерполированного движения суставов (инструкции интерполированного движения суставов: JMOVE, JAPPRO, JDEPART, DRIVE и т.д.)

Шесть инструкций для конфигурации перечислены здесь, программный пример дается в примечании на следующих страницах.

LEFTY, RIGHTY

Устанавливает положение первых трех суставов (JT1, JT2, JT3) робота. LEFTY устанавливает конфигурацию робота, чтобы напоминать левую руку человека, RIGHTY, чтобы напоминать правую руку.



LEFTY



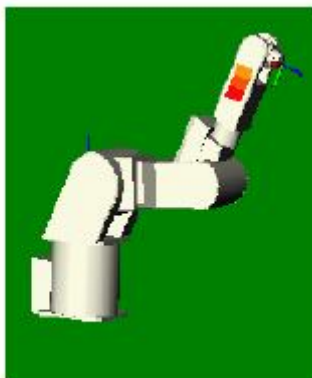
RIGHTY

ABOVE, BELOW

Устанавливает конфигурацию робота, так что третий сустав робота (JT3) находится верхней позиции (аналогия локоть вверх (ABOVE)), или в нижней позиции (аналогия локоть вниз (BELOW)).



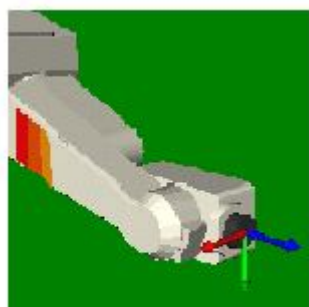
ABOVE



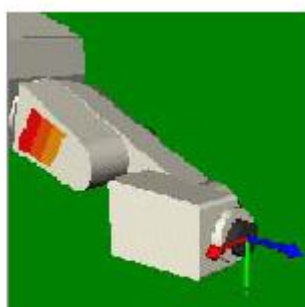
BELOW

UWRIST, DWRIST

Устанавливает конфигурацию робота, так что значение правого сустава (JT5) является положительным (UWRIST) или отрицательным (DWRIST) с поверхностью фланца инструмента в одном и том же положении.



UWRIST



DWRIST

(ПРИМЕЧАНИЕ)

1. Вообще, команды конфигурации не имеют эффекта на значение угловых координат (позиции, названные с #), и робот перемещаются в обученную позицию в обученной конфигурации. Однако, даже если позиция назначения обучается в значениях угловых координат, может произойти ошибка при перемещении робота в прямолинейных интерполированных движениях между позициями, где конфигурация в начальной точке значительно отличается от конфигурации в конечной точке.

2. Робот не реагирует немедленно на команду конфигурации. Конфигурация изменяется при выполнении следующего интерполированного движения суставов (JMOVE, JAPPRO, JDEPART, DRIVE и т.д.).

3. В обычных программах, конфигурация не должна изменяться, за исключением того, когда это изменение производится преднамеренно. Инструкции конфигурации используются в следующих случаях:

(1) Когда программа не начинается с команды движения, которая перемещает робот в позицию, заданную угловыми координатами, команда конфигурации должна быть вставлена в начале программы, чтобы определить конфигурацию робота.

(2) Когда команда JAPPRO используется, как показано ниже, команда конфигурации должна быть использована:

```
JMOVE a
JAPPRO #b,100
JMOVE #b
```

Конфигурация после команды движения “JAPPRO #b,100” может, в соответствии с конфигурацией предыдущего движения, отличаться от положения в #b. Если положения запястья (\pm угла JT5) различны при выполнении следующего шага инструкция “JMOVE #b”, может заставить JT4 и JT6 вращаться очень быстро. Способ избежать этого состоит в том, чтобы обучить позицию, которая на 100 мм выше чем #b, как #bb и использовать команду JMOVE.

```
JMOVE a
JMOVE #bb
JMOVE #b
```

:

Другой способ избежать отклонения в JT4 и JT6 состоит в том, чтобы задать конфигурацию запястья командой конфигурации.

```
JMOVE a
UWRIST
JAPPRO #b,100
JMOVE #b
```

:

Когда градус угла JT5, заданного в #b, положителен используйте команду UWRIST (команда для JT5 > 0) прямо перед командой “JAPPRO #b, 100” так, чтобы конфигурация Запястья была такой же, как конфигурация в #b.

ПРИЛОЖЕНИЕ 1 КОДЫ ОШИБОК

Код	Старый код	Сообщение об ошибке
P0100	- 350	Недопустимый ввод данных
P0101	- 351	Очень много аргументов
P0102	- 353	Вводимые данные очень большие
P0103	- 361	Недопустимый номер РС
P0104	- 365	Недопустимый номер робота
P0105	- 366	Недопустимая программа
P0106	- 367	Недопустимый приоритет
P0107	- 368	Недействительное координатное значение
P0108	- 400	Синтаксическая ошибка
P0109	- 401	Недействительный оператор (утверждение)
P0110	- 402	Двусмысленный оператор (утверждение)
P0111	- 403	Эта команда или инструкция не может использоваться здесь
P0112	- 404	Не может быть выполнена с DO командой
P0113	- 406	Нет такой программной инструкции
P0114	- 410	Недопустимое выражение
P0115	- 411	Недопустимая функция
P0116	- 412	Недопустимый аргумент функции
P0117	- 413	Недопустимое имя переменной (программы)
P0118	- 414	Недопустимый тип переменной
P0119	- 415	Недопустимый индекс массива
P0120	- 416	Отсутствуют круглые скобки
P0121	- 417	Ожидаемый быть бинарным оператором
P0122	- 418	Недопустимая постоянная
P0123	- 419	Недопустимый классификатор
P0124	- 420	Недействительная метка
P0125	- 422	Отсутствие ожидаемых символов
P0126	- 423	Недопустимое имя переключателя
P0127	- 424	Двусмысленное имя переключателя
P0128	- 425	Недопустимый формат спецификатора
P0129	- 426	Повторение оператора метки
P0130	- 430	Не может быть определен как массив
P0131	- 431	Измерение выходит за пределы 3 (больше трехмерного)
P0132	- 433	Массив переменных существует
P0133	- 434	Ни один массив переменных не существует
P0134	- 435	Ожидаемая переменная типа массив
P0135	- 440	Ожидаемая локальная переменная
P0136	- 441	Непредвиденный индекс
P0137	- 442	Несовпадение аргументов в вызове подпрограммы
P0138	- 443	Несовпадение типа аргумента в вызове подпрограммы
P0139	- 450	Недействительная структура управления
P0140	- 451	Шаг: XX, не соответствует END состоянию
P0141	- 452	Шаг: XX дополнительное END состояние
P0142	- 453	Шаг: XX не может завершить DO с END
P0143	- 454	Шаг: XX нет VALUE состояния после CASE
P0144	- 455	Шаг: XX предшествует отсутствующему IF

P0145	- 456	Шаг: XX предшествует отсутствующему CASE
P0146	- 457	Шаг: XX предшествует отсутствующему DO
P0147	- 458	Шаг: XX Не может быть обнаружен END для XX
P0148	- 459	Шаг: XX Очень много управляющих структур
P0149	- 460	Переменная (программа) уже существует
P0150	- 461	Переменная иного типа уже существует
P0151	- 464	Внутренний буфер для сложного выражения
P0152	- 465	Неопределенная переменная (программа)
P0153	- 466	Недопустимые значения синхронизации
P0154	- 470	Ожидает '='
P0155	- 471	Ожидает ')'
P0156	- 472	Ожидает ']'
P0157	- 473	Ожидает "TO"
P0158	- 474	Ожидает "BY"
P0159	- 475	Ожидает ':'
P0160	- 476	Ожидает "ON/OFF"
P0161	-477	Ожидает номер робота
P0162	-	Не может скорректировать позицию в этой инструкции
P1000	- 200	Не может выполнить программу, т.к. силовое питание привода выключено
P1001	- 201	Не может выполнить программу в ручном режиме
P1002	- 202	Не может выполнить программу, т.к. переключатель (T.Lock) в положение ON
P1003	- 213	Не может выполнить программу из-за EXT-IT
P1004	- 217	Не может выполнить программу из-за программно сброса
P1005	- 218	Не может выполнить программу из-за EXT.START ENABLE
P1006	- 219	Не может выполнить программу из-за EXT.START DISABLE
P1007	- 220	Сигнал старта был получен, нет RPS END шага.
P1008	- 221	Не может выполнить программу из-за положения HOLD переключателя на ОП
P1009	- 300	Программа уже выполняется
P1010	- 301	Управляющая программа робота уже выполняется
P1011	- 302	Не может продолжить выполнение этой программы. Используйте EXEC.
P1012	- 303	Робот находится в движении
P1013	- 304	Не может выполнить, т.к. состояние ошибки. Сбросьте ошибку
P1014	- 314	Не может выполняться, т.к. программа уже используется
P1015	- 326	Не может удалить, т.к. используется другой командой
P1016	- 327	Используется в программах
P1017	- 328	Используется в редакторе
P1018	- 329	KILL или PSKILL для того, чтобы удалить программу
P1019	- 308	Выполняется РС программа
P1020	- 320	Не может работать, т.к. ПРУ работает
P1021	- 306	Не может выполняться с DO командой
P1022	- 324	Не может выполняться с MC инструкцией
P1023	- 325	Не может выполняться в программе робота
P1024	- 405	Утверждение не может быть выполнено
P1025	- 1231	Функция не может быть выполнена, т.к. она не установлена
P1026	- 835	Не может быть KILL т.к. программа выполняется

P1027	- 211	Программа не может редактироваться, т.к. переключатель (T.Lock) в положение ON
P1028	- 330	Не может быть вставлен
P1029	- 490	Программное имя не определено
P1030	- 494	Программа заблокирована другой процедурой
P1031	- 700	Нет свободной памяти
P1032	- 801	Нет программных шагов
P1033	- 834	Программное имя уже существует
P1034	- 1905	Программа не доступна для редактирования
P1035	- 545	Запись запрещена. Установите “запись разрешена” и работайте снова
P1036	- 546	Программное изменение запрещено. Установите “АССЕРТ” и работайте снова
P1037	- 1907	Программное имя, включающее “calib_load”, недопустимо. Измените имена этих программ
P1038	- 800	Программа не выполняется
P1039	- 208	Пульт ручного управления не подключен
P1040	-	Не может выполнить эту программу с панели интерфейса
P1041	-	Автоматический контроль команд нарушен
P1042	-	NUM программа выполняется
P1043	-	Не может выполняться в автоматическом режиме
P1044	-	Не может выполняться из-за включенного силового питания привода
P2000	- 101	Выключено силовое питание привода
P2001	- 207	Установите в HOLD переключатель HOLD/RUN
P2002	- 103	Нет никакой внешней оси
P2003	- 104	Недопустимый тип позиционера
P2004	- 105	Не может изменить данные, т.к. существуют данные пользователя
P2005	- 106	Ошибка области графики
P2006	- 107	Опция в состоянии OFF
P2007	- 212	Не может выполняться, т.к. выполняется другими устройствами
P2008	- 514	Внешнее устройство не готово
P2009	- 523	Недопустимое имя файла
P2010	- 531	Диск не готов
P2011	- 533	Недействительный формат диска
P2012	- 535	Диск защищен
P2014	- 537	Диск заполнен
P2015	- 538	Не может быть записан в файл только для чтения
P2016	- 551	Не может открыть файл
P2017	- 552	Не может закрыть файл
P2018	- 559	Данные памяти в настоящее время записываются
P2019	- 563	ADC функция используется другим процессом
P2020	- 591	Недопустимый номер внешнего устройства
P2021	- 598	Не может выполняться с этого терминала
P2022	- 654	Не может использовать DOUBLE OX.
P2023	- 670	В совместном режиме
P2024	- 678	Недействительное значение координаты X
P2025	- 679	Недействительное значение координаты Y

P2026	- 680	Недействительное значение координаты Z
P2027	- 690	Не может использоваться, т.к. сигнал используется в панели интерфейса
P2028	- 1223	Плата ID манипулятора в состоянии занятости
P2029	- 1247	Данные установки сустава не корректны
P2030	- 444	Неопределенный номер функции
P2031	- 446	Удаленный шаг был шагом места назначения JUMP или CALL инструкций
P2032	- 360	Недопустимый параметр WHERE
P2033	- 390	Запись выполняется
P2034	- 391	Неопределенная память
P2035	- 392	Нет данных
P2036	- 393	Ошибка проверки памяти
P2037	- 463	Модуляция траектории в реальном масштабе времени уже началась
P2038	- 100	Ошибка вычисления матрицы
P2039	- 447	Не может выполняться от FN инструкции
P2040	- 4210	Карта (плата) не готова
P2041	- 4211	Неправильная карта загрузилась
P2042	- 4212	Карта защищена от записи
P2044	- 4214	Карта не отформатирована
P2045	- 4215	Не может отформатировать эту карту
P2046	- 4216	Ошибка инициализации карты
P2047	- 4217	Файл уже открыт
P2048	- 4218	Файл не существует на карте
P2049	- 4219	Пытается открыть очень много файлов
P2050	- 4220	Неизвестная ошибка в момент доступа к карте
P2051	- 4221	Недопустимые номера последовательности для данных I/O файла
P2052	- 4252	[LSEQ] Программа включает недействительные инструкции
P2053	- 4253	[LSEQ] Существует очень много шагов
P2054	- 4254	[LSEQ] Недействительный тип сигнальных переменных
P2055	- 4255	[LSEQ] Программа уже выполняется
P2056	- 4256	[LSEQ] Сигнальный номер есть верхний предел
P2057	- 4263	[Serial flash] Не может открыть файл
P2058	- 4264	[Serial flash] Ошибка считывания данных
P2060	- 4266	[Serial flash] Файл или каталог не существует
P2061	- 4269	Файл не существует на дискете
P2062	- 4270	[FDD/PCCARD] Сбой в записи данных при проверки функции
P2063	- 4271	[FDD/PCCARD] Недопустимая характеристика для установки функции проверки
P2064	- 4272	[FDD] Существует несколько возможностей допустимого диапазона чисел
P2065	- 4273	[Multi Discs] Недопустимый диск был загружен
P2066	- 4278	Состояние загрузки групповой записи или группового считывания делает неспособной запись
P2067	- 4279	[Serial Flash] Ошибка каталога файлов
P2068	- 4280	Программа не может выполняться, т.к. она сейчас редактируется
P2069	-	[FDD/PCCARD] Внешнее устройство используется в другом направлении
P2070	-	Данные не могут быть больше зарегистрированы

P2071	- 332	C/S переключатель логически непоследовательно установлен
P2072	-	[LSEQ] Избыток максимальных циклов выполнения
P2073	-	[LSEQ] Другие программы ожидают выполнения
P2074	- 558	Гибкий диск сломался
P2075	-	Обозначение канала JtXX ошибочно
P4500	- 3064	(FIELD-BUS) Интерфейс не разрешен
P4501	- 3053	DEVNET) Устройство XX не находится в списке возможных устройств
P4502	- 3052	DEVNET) Уже в этом режиме
P4503	- 15	Дублированный номер сигнала
P5000	- 316	Ожидание завершения сварки
P5001	- 318	Ожидание отмены или продолжения положительного входного сигнала
P5002	- 319	Выполняется последовательность точечной сварки
P5003	- 371	Тип внешней оси и тип данных горелки неудачно сочетаются
P6000	- 2101	Сдвинутое положение шага XX вышло за допустимые границы
P6001	- 2102	Шаг XX в исходной программе находится вне пределов движения
P6002	- 2106	Заданный PAINTING DATA BANK не определен
P6003	- 2144	Не может выполнить программу из-за отложенного воспроизведения
P6500	- 818	Не может создать направление рабочей линии
P6501	- 819	Недопустимое положение инструмента
P6502	- 934	В многослойной сварке. База данных не существует
P6503	- 935	В многослойной сварке. Точка расплавления выходит из пределов
P6504	- 944	Шаг: XX предшествующий L.START отсутствует
P7000	-	Не может сбросить программу, т.к. не находится в позиции home1
P7001	-	Только NOP Interp, из-за режима настройки давления
W1000	- 50	Не может двигаться вдоль прямой линии JtXX в этой конфигурации
W1001	- 57	Превышает максимальную скорость движения суставов при проверке. Установите медленную скорость
W1002	- 58	Рабочая информация была очищена
W1003	- 63	Калибровка была повреждена. Попробуйте снова после изменения конфигурации.
W1004	- 64	JTXX-M вышел за пределы движения. Проверьте рабочую область
W1005	- 65	Недопустимый центр гравитации, установлен параметр по умолчанию
W1006	- 66	Недопустимый момент нагрузки, установлен параметр по умолчанию
W1007	- 102	Использование установки изменено. Выключите – включите силовое питание
W1008	- 108	Параметр изменен. Выключите – включите силовое питание
W1009	- 317	Ошибка границы позиции JTXX-M в последнем аварийном останове
W1010	- 1022	Низкое напряжение батареи ОЗУ
W1011	- 1217	PLC (программируемый логический контроллер) аварийный сигнал (XX)
W1012	- 1249	Изменился серво параметр. Выключите – включите силовое питание
W1013	- 1511	Низкое напряжение батареи энкодера [Servo (XX)]
W1014	- 1248	Номер оси изменен. Инициализируйте снова
W1015	- 61	Возможен сбой
W1016	- 60	Вращающий момент двигателя сверх предела. JTXX
W1017	-	Низкое напряжение батареи энкодера [Внешняя ось (AX)]
W1018	-	Параметр цепи изменен. Выключите – включите силовое питание
W1019	- 62	Зарегистрированное значение вне номинальной нагрузки
W1020	- 547	Был найден ошибочный сектор

W4500	- 3004	FIELD-BUS) подчиненный порт OFFLINE
W4501	- 3007	FIELD-BUS) главный порт OFFLINE
W5000	- 609	Разблокировка позиции ожидания из-за режима измерения давления
W5001	- 605	PLC ошибка обмена информацией
W5002	- 606	Сварочный контроллер XX не подключен
W5003	- 607	Сварочный контроллер XX не отвечает
W5004	- 608	Сварочный контроллер XX отвечает ошибкой
W5005	- 621	(Точечная сварка) Нет ответа от RWC (чтение-запись-счет) XX
W5006	- 622	(Точечная сварка) RWC отвечает ошибкой XX
W5007	- 623	(Точечная сварка) Сварочный отказ XX
W5008	- 625	(Точечная сварка) Ошибка XX отсоединения кабеля
W5009	- 626	(Точечная сварка) Внутренняя утечка XX
W5010	- 627	(Точечная сварка) Основной кабель обмена тревожным сигналом XX
W5011	- 629	(Точечная сварка) Нет соединения с RWC XX
W6000	- 2109	Пришло время смазывать редуктора и подшипники двигателя
W6001	- 2110	Пришло время для замены основного кабеля робота
W6002	- 2111	Пришло время заменить охлаждающие вентиляторы в контроллере
W6003	- 2112	Пришло время заменить источник постоянного тока в контроллере
W6004	- 2113	Пришло время заменить устройство питания серводвигателя
W6005	- 2114	Пришло время заменить силовой усилитель для руки робота
W6006	- 2115	Пришло время заменить силовой усилитель для запястья робота
W6007	- 2116	Пришло время заменить силовой усилитель для транспортной каретки
E0001	- 1	Неизвестная ошибка
E0100	- 445	Существует непредусмотренный комментарий инструкции
E0101	- 802	Запрещенная метка
E0102	- 803	Метка не определена
E0103	- 804	Данные места положения не определены
E0104	- 805	Строковая переменная не определена
E0105	- 807	Программа или метка не заданы
E0106	- 808	Значение выходит из допустимого диапазона
E0107	- 809	Нет индекса массива переменных
E0108	- 810	Деление на ноль
E0109	- 811	Переполнение с плавающей запятой
E0110	- 812	Очень длинная строка
E0111	- 813	Недопустимая операция возведения в степень
E0112	- 814	Слишком много сложных выражений
E0113	- 815	Нет выражений для вычисления
E0114	- 817	Отрицательное значение квадратного корня
E0115	- 820	Недопустимое значение индекса массива переменных
E0116	- 821	Недопустимое значение аргумента
E0117	- 822	Недопустимый номер сустава
E0118	- 840	Слишком много вызовов подпрограммы
E0119	- 842	Несуществующая подпрограмма
E0900	- 427	Ошибка контрольной суммы блокирует пошаговую инструкцию
E0901	- 499	Данные шага нарушены

E0902	- 816	Данные выражения нарушены
E0903	- 1019	Ошибка контрольной суммы системных данных
E1000	- 561	Ошибка канала ADC (аналого-цифровой преобразователь)
E1001	- 562	Ошибка диапазона входа ADC
E1002	- 571	Ошибка PLC интерфейса
E1003	- 573	Встраиваемый PLC (программируемый логический контроллер) не установлен
E1004	- 578	Межшинная плата не готова
E1005	- 983	Ошибка приращения энкодера оси вращения
E1006	- 1029	Контактный панельный переключатель коротко замкнут
E1007	- 1201	Плата последовательности питания не установлена
E1008	- 1202	Второстепенная плата последовательности питания не установлена
E1009	- 1203	№ XX-M I/O плата не установлена
E1010	- 1205	Последовательность питания обнаруживает ошибку
E1011	- 1206	Встраиваемая плата последовательности питания не установлена
E1012	- 1208	RI/O плата или C-NET плата не установлены
E1013	- 1210	Межшинная плата не установлена (INTER-BUS)
E1014	- 1211	Сдвоенный порт памяти для обмена информацией не установлен
E1015	- 1212	Плата интерфейса усилителя не установлена
E1016	-1213	№ XX-M CC-LINK (канал передачи данных цветового (классификационного) кода) плата не установлена
E1017	-1214	Ошибка PLC. Код ошибки Hex.XX (шестнадцатеричный)
E1018	-1215	Ошибка состояния INTER-BUS
E1019	-1216	Плата последовательности питания для устройства обеспечения безопасности (аварийной защиты) не установлена
E1020	- 1218	Внешнее оборудование аварийно
E1021	- 1221	Ошибка платы ID манипулятора (код XX)
E1022	- 1222	Ошибка платы последовательности питания (код XX)
E1023	- 1228	Ошибка обмена информацией в вычислительной сети робота
E1024	- 1262	Ошибка последовательности отключения внешней оси (код XX)
E1025	- 1263	Ошибка последовательности подключения внешней оси (код XX)
E1026	- 1264	Основной идентификатор центрального процессора неправильно подобран
E1027	- 1336	Цепь аварийной защиты была отключена
E1028	- 1500	JtXX-M двигателя перегружен
E1029	- 1513	Данные вращения энкодера неправильные (jtXX-M)
E1030	- 1516	Данные энкодера неправильные (jtXX-M)
E1031	- 1518	Неправильный подсчет энкодерных данных (jtXX-M)
E1032	- 1521	Несоответствие ABS (абсолютные значения) и INC (увеличенные на единицу) данных энкодера (jtXX-M)
E1033	- 1524	Ошибка линии энкодера (jtXX-M)
E1034	- 1534	Ошибка инициализации энкодера (jtXX-M)
E1035	- 1553	Ошибка срабатывания энкодера (jtXX-M)
E1036	- 1554	Ошибка обмена информации энкодера (jtXX-M)
E1037	- 1555	Ошибка преобразования данных энкодера (jtXX-M)
E1038	- 1556	Ошибка ABS- канала энкодера (jtXX-M)
E1039	- 1557	Ошибка INC-импульса энкодера (jtXX-M)
E1040	- 1558	Ошибка регистра памяти датчика энкодера (jtXX-M)
E1041	- 1601	Переключатель предела (JT-XX) включен
E1042	- 1602	Сигнальная линия переключателя предела не подключена

E1043	- 1605	Штепсельный разъем пульта сбоит
E1045	- 628	(Точечная сварка) Горелка – фиксатор не соответствуют друг другу
E1046	- 659	Очень короткое расстояние между начальной и конечной точками
E1047	- 2015	Не может выполняться в режиме отслеживания конвейера
E1048	- 109	Данные смещения обнуления - недопустимое значение
E1049	- 897	Текущая позиция вне области
E1050	- 634	Сигнал отключения питания для тормоза и энкодера не является приоритетным
E1051	- 653	Недопустимый парадоксальный ОХ выход
E1052	- 656	Сигнал опознавания детали не является приоритетным
E1053	- 657	Сигнал опознавания детали уже получен
E1054	- 661	Не может выполнить инструкцию движения
E1055	- 662	Ошибка позиции начальной точки для круговой интерполяции
E1056	- 663	Главный робот уже существует
E1057	- 664	Этот робот не является главным
E1058	- 665	Подчиненный робот уже существует
E1059	- 699	Не является инструкцией для сложного движения
E1060	- 671	Не может выполняться в режиме обратной проверки
E1061	- 672	Не может выполняться в ONE программе
E1062	- 673	JT2 и JT3 служат помехой друг другу в позиции старта
E1063	- 674	JT2 и JT3 служат помехой друг другу в позиции окончания движения
E1064	- 681	Недопустимый номер паллеты
E1065	- 682	Недопустимый номер детали
E1066	- 683	Недопустимый номер образца
E1067	- 684	Недопустимый тип образца
E1068	- 685	Недопустимые данные детали
E1069	- 686	Недопустимые данные паллеты
E1070	- 693	ON/ONI сигнал уже введен
E1071	- 694	XMOVE сигнал уже введен
E1072	- 697	Данные позиции безопасности не определены
E1073	- 824	Недопустимый номер таймера
E1074	- 825	Превышает максимальный сигнальный номер
E1075	- 826	Недопустимый номер фиксатора
E1076	- 827	Недопустимое значение времени
E1077	- 828	Значение не установлено
E1078	- 829	Недопустимый сигнальный номер
E1079	- 837	Не может использовать приоритетный сигнал
E1080	- 838	Не является RPS режимом
E1081	- 839	Не может использовать отрицательное значение
E1082	- 850	Выходит за абсолютный нижний предел
E1083	- 851	Выходит за абсолютный верхний предел
E1084	- 852	Выходит за нижний предел, установленный пользователем
E1085	- 853	Выходит за верхний предел, установленный пользователем
E1086	- 855	Начальное положение сустава XX вне диапазона движения
E1087	- 856	Конечное положение сустава XX вне диапазона движения
E1088	- 857	Позиция назначения вне диапазона движения

E1089	- 858	Недопустимая конфигурация для прямолинейного движения
E1090	- 863	Данные внешней модуляции не введены
E1091	- 864	Данные внешней модуляции неправильные
E1092	- 865	Данные модуляции за пределами
E1093	- 866	Недопустимая команда движения в выполнении скорректированного движения
E1094	- 871	Недопустимый номер сустава
E1095	- 872	Не может выполнить инструкцию движения в РС программе
E1096	- 873	Недопустимый номер дополнительных данных
E1097	- 874	Нет следующей C1MOVE или C2MOVE инструкции
E1098	- 875	C1MOVE (CIR1) не инструкция
E1099	- 876	Проверьте три точки (назад вперед)
E1100	- 877	Не может выполняться в спецификации герметизации
E1101	- 879	Не может выполняться, кроме как в случае спецификации герметизации
E1102	- 896	Т.к. опция не установлена, выполниться не может
E1103	- 898	Выше позиции конвейера
E1104	- 978	Очень много SPINMOVE инструкций
E1105	- 1032	Координата находится в CUBE
E1106	- 1051	Не может выполнить с этим роботом
E1107	- 1219	Не может использовать SEPARATE CONTROL (независимое управление)
E1108	- 1227	ID схема энергоснабжения робота дублируется
E1109	- 1267	Плата интерфейса конвейера не установлена
E1110	- 1735	GROUP не является начальным этапом процесса
E1111	-2018	JTXX не может двигаться для упорядочивания движения
E1113	- 658	Сигнал опознавания детали не обнаружен
E1114	- 666	Прерывание в совместном управлении
E1115	- 669	Принудительное завершение совместного управления
E1116	- 979	Ось вращения не останавливается в каждом 360 градусах
E1117	- 1011	Сверх времени обработки
E1118	- 1012	Командная позиция jtXX внезапно изменилась
E1119	- 1014	Выход за пределы движения в командной позиции JTXX
E1120	- 1017	Текущая команда вмешивается между Jt2 и Jt3
E1121	- 1031	Другой робот уже находится в зоне интерференции (пересечения)
E1122	- 1308	Непредсказуемое выключение силового питания привода
E1123	- 1503	Ошибка скорости jtXX-M
E1124	- 1504	Ошибка граничной позиции jtXX-M
E1125	- 1505	Ошибка граничной скорости jtXX-M
E1126	- 1506	Ошибка командной скорости jtXX-M
E1127	- 1507	Ошибка командного ускорения jtXX-M
E1128	- 1600	Ошибка рассогласования между позицией назначения и текущей позицией jtXX-M
E1129	-1737	JtXX движется в момент регулирования внешних осей
E1130	- 1902	JtXX столкновение обнаружено
E1131	- 1904	JtXX внезапное ударное воздействие обнаружено
E1132	- 1906	Выключение силового питания привода. Контроль (измерение) остановлен
E1133	- 2011	Деталь достигла максимального положения на конвейера
E1134	- 2012	Неправильный рабочий ход конвейера

E1135	- 600	Силовое питание привода выключено
E1136	- 675	Стандартный терминал не подключен
E1137	- 676	Не может быть ввода/вывода с мульти функциональной панели
E1138	- 698	Дополнительный терминал не подключен
E1139	- 869	DA плата не установлена
E1140	- 2009	Нет оси конвейера
E1141	- 2010	Положение конвейера вне предела
E1142	- 2013	Нет следующей оси
E1143	- 2014	Номер оси конвейера не установлен
E1144	- 4245	Нет платы управления манипулятором
E1145	- 4249	Не может использовать заданный канал, т.к. он используется сейчас
E1146	- 4251	[LSEQ] Прерывание во время процесса
E1147	- 4257	Не может открыть файл настройки, т.к. он не был установлен во время отгрузки
E1148	- 4258	Не может прочитать файл настройки, т.к. он не был установлен во время отгрузки
E1149	- 4259	Не может открыть данные настройки, т.к. они не были установлены при состоянии поставки
E1150	- 4260	Не может прочитать данные настройки, т.к. они не были установлены при состоянии поставки
E1151	- 4261	Очень много данных для настройки при состоянии поставки
E1152	- 4262	Имя данных настройки при состоянии поставки очень длинное
E1153	- 4268	Плата последовательности питания обнаружила ошибку (Code=XX)
E1154	- 1204	Дополнительный SIO (последовательный ввод/вывод) порт не установлен
E1155	- 560	A/D (аналого-цифровой) преобразователь не установлен
E1156	-	[Плата управления манипулятором] Время обработки превышено
E1157	- 1225	Ошибка платы Arm ID I/F (code XX)
E1158	-	(SSCNET) Ошибка серводвигателя jtXX-M
E1159	-	(SSCNET) Код ошибки для серводвигателя XX
E1160	-	(SSCNET) Ошибка двигателя и контроля установленной ошибки jtXX-M
E1161	-	Эта машина не может поддерживать функцию автоматической калибровки инструмента
E1162	-	Буфер излишнего технологического процесса обнаружен в вычислительном канале XX-M буфера гравитации
E1163	-	Робот остановился. Проверка рабочей области остановлена
E1164	-	[LSEQ] Ошибка программного выполнения при включенном силовом питании (код XX)
E1165	-	Ошибка загрузки параметра внешней оси (Jt-A)
E1166	-	Номер сустава не установился в канале (Jt-A)
E1167	-	Ошибка загрузки параметра внешней оси (Jt-B)
E1168	-	Номер сустава не установился в канале (Jt-B)
E1169	-	Ошибка в параметре двигателя изменяет последовательность (код XX)
E4000	-	Ошибка данных связи
E4001	-	Ошибка чтения данных
E4002	-	Ошибка записи данных
E4003	-	Непредсказуемая ошибка доступа к файлу
E4004	-	Повторяющаяся ошибка связи
E4005	-	Процесс связи был остановлен
E4006	-	Не получено никаких данных после запроса
E4007	-	Очень длинные полученные данные (MAX=255 символов)
E4008	-	Неправильные данные (ЕОТ конец передачи) получены при связи

E4009	- 586	Выход ошибки во время связи
E4010	- 596	Не может подключить терминал
E4011	- 597	Не может подключить порт связи
E4012	- 599	Ожидание ввода данных для PROPT. Подключите устройство ввода
E4013	- 2200	TELNET) (сетевой теледоступ) SEND ошибка (отправки) Код=XX
E4014	- 2201	TELNET) (сетевой теледоступ) RECV ошибка (получения). Код=XX
E4015	- 2202	TELNET) (сетевой теледоступ) IAC ошибка получения. Код=XX
E4016	- 2203	TELNET) Закрытое повреждение.. Код=XX
E4017	- 2204	TELNET) Повреждение главного двунаправленного закрытого канала Код=XX
E4018	- 2205	TELNET) системная ошибка . Код=XX
E4019	- 2206	TCP/IP) Повреждение открытого двунаправленного канала. Код=XX. Dst.IP=XX.XX.XX.XX
E4020	- 2207	TCP/IP) Повреждение закрытого двунаправленного канала. Код=XX. Dst.IP=XX.XX.XX.XX
E4021	- 2208	TCP/IP) Ошибка обмена информацией Код=XX. Dst.IP=XX.XX.XX.XX
E4022	- 2209	TCP/IP) Очень длинное сообщение
E4023	- 2210	TCP/IP) Не может достигнуть главного компьютера
E4024	- 2211	TCP/IP) Выход времени связи. Dst.IP=XX.XX.XX.XX
E4025	- 2212	TCP/IP) Подключение прервано
E4026	- 2213	TCP/IP) Нет буферного пространства
E4027	- 2214	TCP/IP) Испорченный разъем
E4028	- 2216	FTP) (программа передачи файлов) Ошибка приема данных (код=XX)
E4029	- 2217	FTP) Ошибка отправки данных (код=XX)
E4030	- 2218	FTP) Неузнанная команда (код=XX)
E4031	- 2219	FTP) Неудачный выход из системы с FTP сервером
E4032	- 2220	FTP) Обнаружена незарегистрированная операционная система
E4033	- 2221	FTP) Неудачное соединение с сервером (код=XX)
E4034	- 2222	FTP) Неудача в получении информации об операционной системе главного компьютера (код=XX)
E4035	- 2224	FTP) TCP/IP (протокол управления передачей) не инициализирован
E4036	- 2225	FTP) FTP обслуживание сейчас занято
E4037	- 2226	FTP) Неудачное AUTO-SAVING (автосохранение)
E4050	- 4206	Нет ответа от платы драйвера FDD/PC_CARD
E4051	-	Нет связи с платой драйвера FDD/PC_CARD
E4052	-	[FDD/PC_CARD] отказ в установке проверочной функции. Повторите снова
E4053	- 1075	Ошибка канала
E4054	-	TCP/IP) Не может быть выполнена, т.к. Интернет плата не установлена
E4055	- 2239	TCP) Не может быть создан двунаправленный канал
E4056	- 2231	TCP) Этот порт не находится в ожидании
E4057	- 2232	TCP) Недопустимое распознавание двунаправленного канала
E4058	-	Загрузка программного обеспечения в плату драйвера FDD/PC_CARD завершена неправильно
E4500	- 3003	ANYBUS) IN-AREA требование тайм аута XX.
E4501	- 3006	ANYBUS) OUT/FB.CTRL отключение тайм аута XX.
E4510	- 3050	DN) (отличительное имя) Состояние ведущего устройства XX
E4511	- 3051	DN) Состояние узла XX
E4512	- 3008	ABM-DN) Ошибка почтового ящика
E4520	- 3066	ABMA-PDP) Состояние STOP.XX

E4521	- 3067	ABMA_PDP) Состояние OFFLINE.XX
E4522	- 3056	ABMA_PDP) Ошибка связи I/O данных
E4523	- 3057	ABMA_PDP) Блокировка по времени отправки I/O данных. XX
E4524	- 3058	ABMA_PDP) Блокировка по времени приема I/O данных. XX
E4525	- 3059	ABMA_PDP) Блокировка по времени отправки сообщения. XX
E4526	- 3060	ABMA_PDP) Блокировка по времени приема сообщения. XX
E4527	- 3061	ABMA_PDP) Проверка данных конфигурации. XX
E4528	-3062	PROFIBUS) (высокоскоростная шина цифрового технологического оборудования) Обнаружен ответ подчиненной Diag-errgr. XX
E4529	- 3063	PROFIBUS) Обнаружен ответ статистического счетчика ошибок. XX
E5000	- 603	Подключенный сигнал разрешения не был включен
E5001	- 648	RWC вид не является видом управления процессом
E5002	- 649	IGS плата не является видом управления процессом
E5003	- 611	Недопустимое расширение (сужение) сигнала выхода
E5004	- 615	Сигнал завершения сварки уже введен
E5005	- 620	(Точечная сварка) Список данных сварочных установок неправильный
E5006	- 635	CLAMP SPEC не установлен как PULSE
E5007	- 641	Сварочная горелка с серво управлением отсоединена или подключена другая горелка
E5008	- 643	Критерий износа чипа (STAGE1) был не выполнен
E5009	- 644	Сигнал опознавания детали не установлен
E5010	- 646	Механические параметры сварочной горелки с серво управлением не установлены
E5011	- 647	Номер зажима сварочной горелки с серво управлением сдублирован
E5012	- 689	Не может заменить горелку, т.к. автономные данные неправильные
E5013	- 691	Не может изменить множество горелок в том же самом шаге
E5014	- 692	Горелка подсоединена к другому суставу
E5015	- 695	Состояние горелки не совпадает с состоянием фиксатора
E5016	- 696	Данные SRVPRESS ошибочны
E5017	- 1912	Данные базы износа не зарегистрированы
E5018	- 610	Сигнал завершения сварки не был обнаружен
E5019	- 612	Сигнал сбоя сварки обнаружен
E5020	- 613	Ошибка контроля положения втягивания
E5021	- 614	Ошибка контроля положения расширения
E5022	- 616	Позиция втягивания данной горелки отличается от места назначения
E5023	- 617	Не может измерить износ из-за неправильного износа
E5024	- 618	Закрытие сигнала столкновения не было обнаружено
E5025	- 619	Открытие сигнала столкновения не было обнаружено
E5026	- 624	(Точечная сварка) Ошибка RWC (чтение – запись – счет). XX
E5027	- 630	Робот остановился во время сварки
E5028	- 631	Не может достигнуть желаемого давления
E5029	- 632	Наконечник горелки приварился
E5030	- 633	Износ медной пластины сверх предела. Шаг=XX
E5031	- 638	Сигнал завершения сварки не отключился
E5032	- 642	Калибровка не закончилась нормально
E5033	- 645	Не может быть сварки из-за неправильной толщины
E5034	-	Износ сварочного наконечника сверх предела (MOVING SIDE)
E5035	-	Износ сварочного наконечника сверх предела (FIXED SIDE)

E5500	- 1750	Плата технического зрения не установлена
E5501	- 2751	(Техническое зрение) Камера не подключена
E5502	-2710	(Техническое зрение) Недопустимый параметр
E5503	- 2711	(Техническое зрение) Недопустимый символ
E5504	- 2712	(Техническое зрение) Недопустимое имя
E5505	- 2713	(Техническое зрение) Недопустимая память для хранения изображений
E5506	- 2715	(Техническое зрение) Недопустимые данные гистограммы
E5507	- 2716	(Техническое зрение) Недопустимый режим
E5508	- 2717	(Техническое зрение) Недопустимая плотность (цвет)
E5509	- 2718	(Техническое зрение) Недопустимое назначение входа камеры
E5510	- 2719	(Техническое зрение) Недопустимый номер канала камеры
E5511	- 2720	(Техническое зрение) Недопустимый номер окна
E5512	- 2722	(Техническое зрение) Недопустимые координатные данные
E5513	- 2723	(Техническое зрение) Недопустимый номер
E5514	- 2726	(Техническое зрение) Недопустимое изображение кода (двоичный/ мульти)
E5515	- 2728	(Техническое зрение) Недопустимая пороговая величина
E5516	- 2729	(Техническое зрение) PROTO (/TEMPLATE) не зарегистрированы или уже существуют
E5517	- 2745	(Техническое зрение) Калибровочные данные не зарегистрированы
E5518	- 2746	(Техническое зрение) Графический курсор не инициализирован
E5519	- 2747	(Техническое зрение) Очень много образцов фото объекта
E5520	- 2748	(Техническое зрение) Очень много обнаружений цели
E5521	- 2754	(Техническое зрение) Команда технического зрения не включена
E5522	- 2768	(Техническое зрение) Системные данные сейчас неправильны
E5523	- 2770	(Техническое зрение) Ошибка при обработке изображений
E5524	- 2771	(Техническое зрение) Порт звукового фона назначил другую функцию
E5525	- 2749	(Техническое зрение) Недостаток данных области памяти
E5526	- 2752	(Техническое зрение) Недопустимый режим синхронизации звука и изображения
E5527	- 2753	(Техническое зрение) Теперь обрабатывается система технического зрения
E5528	- 2766	(Техническое зрение) Ошибка процесса копирования изображения
E5529	- 2767	(Техническое зрение) Блокировка по времени или переполнение буфера
E5530	- 2769	(Техническое зрение) Неудачная попытка записи на Флэш память
E5531	- 2772	(Техническое зрение) Фото данные неправильные, поэтому проведите инициализацию
E5532	- 2774	(Техническое зрение) Обнаружение детали не произошло
E5533	- 2797	(Техническое зрение) Ошибка инициализации. Код=XX
E5534	- 2799	(Техническое зрение) Ошибка системы технического зрения
E5535	- 2744	(Техническое зрение) Сейчас автономный режим
E5536	- 2732	(Техническое зрение) Неподходящий параметр камеры/проектора
E5537	- 2721	(Техническое зрение) Недопустимое задание переключателя камеры
E5538	- 2727	(Техническое зрение) Эта плоскость назначена в другую камеру
E5539	- 2755	(Техническое зрение) Край не найден
E5540	- 2761	(Техническое зрение) Неподходящие HIS (цветовой фон) данные
E5541	- 2762	(Техническое зрение) Н данные диапазона ширины больше чем 128
E5542	- 2731	(Техническое зрение) Никакое изображение расстояния не введено в устройство
E5543	- 2758	(Техническое зрение) Несоответствующие данные точек кромки для вычисления

E5544	- 2760	(Техническое зрение) Несоответствующая цветная перекодировка типа таблицы преобразования в конфигурации системы
E5545	- 2725	(Техническое зрение) Недопустимый размер площади
E5546	- 2756	(Техническое зрение) Световой штрих не существует
E5547	- 2724	(Техническое зрение) Недопустимое количество корреляционных векторов
E5548	- 2759	(Техническое зрение) Несоответствующие данные вектора
E5549	- 2775	(Техническое зрение) X-FIT окружающая среда не является оборудованием
E5550	- 2776	(Техническое зрение) Мышь не инициализирована
E5551	- 2777	(Техническое зрение) Плата переключателя камеры не установлена
E6000	- 2100	Пульт ручного управления для проверки извлечения архивированных файлов не подключен
E6001	- 2140	Следующий шаг XD(2)START должен быть LMOVE или HMOVE
E6002	- 2141	Сигнальное состояние уже введено
E6003	- 2143	Сигнал обнаружения двери не является приоритетным
E6004	- 2145	Данные позиции не были обнаружены
E6005	- 2173	Недопустимая установка устройства ограждения
E6006	- 2142	Сигнал не обнаружен
E6007	- 2103	Запястье не может выпрямиться больше (особая точка 1)
E6008	- 2104	Запястье не может быть изогнуто больше (особая точка 2)
E6009	- 2171	Не хватает воздушного потока
E6010	- 2105	Выход из за пределы площади движения XYZ (MOVING AREA XYZ LIMIT)
E6011	- 2172	Низкое внутреннее давление
E6012	- 2146	Относительное расстояние между горелками очень маленькое
E6013	- 2147	Не может установить номер программы в программном списке очередности
E6014	- 2148	Не может установить номер программы во временном списке очередности
E6500	- 999	Нет платы сварочного интерфейса
E6501	- 998	Нет платы сварочного интерфейса №2
E6502	- 900	Сбой дуги
E6503	- 901	Проволока прилипла
E6504	- 904	Дуга не зажглась в момент старта
E6505	- 948	Дефект изоляции сварочной дуги
E6506	- 1610	Столкновение горелки
E6507	- 984	Недопустимые данные интерполяции
E6508	- 949	Нет E/N (земля/нейтраль) отношения D/A (цифро-аналоговой) платы
E6509	- 910	Деталь не обнаружена
E6510	- 911	Направление опознавания не обнаружено
E6511	- 912	Неподходящие точки опознавания
E6512	- 913	Материнская или дочерняя деталь не существуют
E6513	- 914	Очень много точек опознавания
E6514	- 915	Недоступное месторасположение детали
E6515	- 916	Недоступное расположение точек опознавания
E6516	- 917	Повреждение контроля проволоки
E6517	- 920	Недоступный номер сварочного состояния
E6518	- 921	Данные сварки не установлены
E6519	- 922	Сварочные данные выходят из допустимого диапазона
E6520	- 923	Вне лазерного датчика, значение слежения

E6521	- 924	Вне лазерного датчика мощность слежения
E6522	- 925	Лазерный датчик не может обнаружить стык
E6523	- 926	Калибровочные данные между горелкой и камерой не готовы
E6524	- 927	Ошибка вычисленных данных, используемых лазерным датчиком
E6525	- 929	Не может обнаружить стык из-за слежения лазерным датчиком
E6526	- 936	Контроллер лазерного датчика не отвечает
E6527	- 937	Ошибка связи лазерного датчика. Код XX.
E6528	- 938	Начало края не найдено лазерным датчиком
E6529	- 939	Окончание края не найдено лазерным датчиком
E6530	- 941	Не может использовать круговую интерполяцию с функцией лазерного датчика
E6531	- 942	Не может включить лазер, потому что выключено силовое питание привода
E6532	- 943	Нет обмена информацией между платой и лазерным датчиком
E6533	- 951	Нет платы RTPM
E6534	- 960	Обученные точки RTPM вне диапазона
E6535	- 961	Ошибка датчика дуги RTPM
E6536	- 964	Ошибка отклонения тока RTPM
E6537	- 962	RTPM значение слежения вне диапазона
E6538	- 963	Вне RTPM мощности слежения
E6539	- 969	Вне AVC значения слежения
E6540	- 971	Вне AVC мощности слежения
E6541	- 967	Нет платы AVC
E6542	- 972	Ошибка отклонения напряжения AVC
E6543	- 968	Очень много обученных точек для AVC
E6544	- 990	Значение слежения гипер дуги вне диапазона
E6545	- 991	Вне гипер дуги мощность слежения
E6546	- 993	Окончание сварки узким швом не найдено
E6547	- 994	Окончание края не найдено
E6548	- 995	Погрешность защитной оболочки горелки гипер дуги
E6549	- 996	Ошибка калибровки горелки гипер дуги
E6550	- 997	Ошибка индекса фазы Z гипер дуги
E6551	- 985	Нет платы гипер дуги
E6552	- 986	Ошибка платы гипер дуги. Код XX.
E6553	- 987	Ошибка датчика тока гипер дуги
E6554	- 988	Ошибка датчика напряжения гипер дуги
E6555	- 989	Ошибка отклонения тока гипер дуги
E6556	- 992	Ошибка усилителя гипер дуги. Код XX.
E6557	- 945	Нет платы управления подачей проволоки
E6558	- 946	Ошибка управления подачей проволоки. Код XX.
E6559	- 947	Погрешность отклонения скорости подачи проволоки
E6560	- 965	Не может калиброваться снова при сварке
E6561	- 966	Не может сваривать, т.к. перекалибровка выполняется снова
E6562	- 902	Залипание электрода
E7000	-	Сварочная горелка с серво управлением отсоединяется
E7001	-	Данные положения содержат неправильные данные
E7002	-	Место расположения слишком далеко от заданной координаты точки

D0001	-1100	Ошибка CPU (центральный процессор) (PC=XX)
D0002	-1101	Ошибка главной шины центрального процессора (CPU BUS) (PC=XX)
D0003	-1102	Ошибка VME шины (PC=XX)
D0004	-4238	[Плата управления манипулятором] Ошибка центрального процессора (PC=XX)
D0005	-4239	[Плата управления манипулятором] Ошибка шины центрального процессора (PC=XX)
D0900	-1003	Данные обучения нарушены
D0901	-1025	Ошибка контрольной суммы AS флэш памяти
D0902	-1026	Ошибка контрольной суммы флэш памяти серво двигателя
D0903	-1028	Ошибка памяти платы IP (XX)
D0904	-1805	Память заблокирована из-за AC_FAIL (нарушения подачи переменного тока)
D1000	-4201	Ошибка чтения программного обеспечения управления серводвигателя
D1001	-4202	Ошибка загрузки программного обеспечения управления серводвигателем
D1002	-4204	Ошибка инициализации ПО серводвигателя
D1003	-4205	Ошибка инициализации ПО управления серводвигателем
D1004	-1300	[Плата управления манипулятором] Ошибка сторожевого таймера ПО управления серводвигателем
D1005	-1306	Командная ошибка платы серводвигателя (XX)
D1006	-1407	Системная ошибка серводвигателя
D1007	-1567	Сверх времени восстановления
D1008	-1568	P-N низкое напряжение [XX]
D1009	-1569	P-N высокое напряжение [XX]
D1010	-1570	Регенеративный резистор перегрелся [XX]
D1011	-1903	Несоответствие между AS ПО и ПО серводвигателя для робота данного типа
D1012	-1909	Тип серводвигателя неподходящий. Проверьте установки
D1013	-4236	P-N конденсатор не разгружен
D1014	-4267	Ошибка системы серводвигателя (Код=XX)
D1015	-4274	Файл данных серводвигателя не существует
D1016	-4275	Файл данных серводвигателя не включает, имеющиеся в действительности данные серводвигателя
D1017	-4276	Ошибка загрузки данных серводвигателя
D1018	-	Несоответствующая версия программного обеспечения серводвигателя
D1500	-1517	Ошибка чтения энкодера JtXX-M
D1501	-1564	Поврежденное подключение преобразователя или ошибка связи энкодера
D1502	-1401	Перегрузка по току усилителя JtXX-M
D1503	-1420	Тип устройства измерительного преобразователя тока (XX) не подходит
D1504	-1424	Неправильная обратная связь по току для JtXX-M (авария усилителя или отсоединение силового разъема)
D1505	-1501	Разъем двигателя отсоединился или перегрелся (XX)
D1506	-1559	Ошибка силового модуля JtXX-M
D1507	-1800	АС (переменный ток) первичного питания отключен
D1508	-1801	24VDC источник питания является слишком заниженным
D1509	-1802	Первичный источник питания является слишком завышенным
D1510	-1803	Первичный источник питания является слишком заниженным
D1511	-1804	+12VDC, -12VDC являются аварийными
D1512	-4208	Ошибка линии тормоза для JtXX-M
D1513	-4209	Мощность тормоза аварийна (XX)
D1514	-4222	I/O 24V плавкий предохранитель не замкнут (отключен)

D1515	-4223	Несоответствие линии (кабеля) цепи аварийной защиты номеру установки
D1516	-4224	Несоответствие HOLD-BACKUP-TIME установке цепи аварийной защиты
D1517	-4225	Плавкий предохранитель аварийной линии цепи безопасности разомкнут
D1518	-4226	Не соответствует условию аварии цепи аварийной защиты
D1519	-4227	Не соответствует LS (менее значащим) условию цепи аварийной защиты
D1520	-4228	Не соответствует TEACH/REPEAT состоянию цепи аварийной защиты
D1521	-4229	Не соответствует состоянию ограждения цепи аварийной защиты
D1522	-4230	Не соответствует состоянию отпирающих устройств цепи аварийной защиты
D1523	-4231	Не соответствует состоянию внешних отпирающих устройств цепи аварийной защиты
D1524	-4232	Некорректное действие реле блокировки
D1525	-4233	Некорректное действие MC (K1)
D1526	-4234	Некорректное действие MC (K2)
D1527	-4235	Некорректное действие MC (K3)
D1528	-4237	Температура внутри контроллера вне диапазона
D1529	-4244	Сигнальный разъем отсоединился или ошибка питания энкодера
D1530	-	Неправильный токовый предел JtXX-M
D2000	-928	Плата обмена информацией для лазерного датчика не отвечает
D2001	-1209	Ошибка инициализации платы RI/O или C-NET
D2002	-1220	Нет ответа от платы ID манипулятора
D2003	-1229	Нет данных в плате ID манипулятора
D2004	-1230	Не соответствует данным в плате ID манипулятора
D2005	-1266	Несоответствие версии CC-LINK (цветовой код-канал связи) программного обеспечения
D2006	-2174	Нет ответа от платы обмена информацией для пульта ручного управления для проверки извлечения архивированных файлов
D2007	-1207	Нет ответа от встроенной платы последовательности
D2008	-1224	Совокупность XX MC P-N напряжения прикреплена
D2009	-2170	Ошибка датчика внутреннего давления
D2010	-4200	Ошибка синхронизации между платой пользовательского интерфейса и платой управления манипулятором
D2011	-4203	Ошибка пересылки параметра между платой пользовательского интерфейса и платой управления манипулятором
D2012	-677	Ошибка SOFT ABSORBER. Включите, выключите силовое питание
D2013	-687	Ошибка CHANGE GAIN. Включите, выключите силовое питание
D2014	-1226	Ошибка инициализации вычислительной сети робота
D2016	-4207	Нет ответа от платы управления манипулятором
D2017	-4240	[USER I/F BOARD] не отвечает
D2018	-4241	[ARM CTRL BOARD] не отвечает
D2019	-4242	[ARM CTRL BOARD] программное обеспечение серводвигателя не отвечает
D2020	-4243	[ARM CTRL BOARD] программное обеспечение управления серводвигателя не отвечает
D2021	-4246	Файл данных манипулятора не найден
D2022	-4247	Данные манипулятора не найдены
D2023	-4248	Неудачная попытка загрузить данные манипулятора
D2024	-4250	[ARM CTRL BOARD] Неудачная установка типа робота
D2025	-	Несоответствие между программным обеспечением и платой управления манипулятором по коду робота
D2026	-	Несоответствие между программным обеспечением и платой интерфейса датчика тока по коду
D2027	-	Несоответствие между программным обеспечением и блоком питания по коду
D2028	-	(SSCNET) Ошибка инициализации (код XX)

D2029	-	Несоответствие между программным обеспечением и платой управления манипулятором по коду двигателя (Jt-A)
D2030	-	Несоответствие между программным обеспечением и платой интерфейса датчика тока по коду (Jt-A)
D2031	-	Несоответствие между программным обеспечением и дополнительным блоком питания по коду (Jt-A)
D2032	-	Несоответствие между программным обеспечением и платой управления манипулятором по коду двигателя (Jt-B)
D2033	-	Несоответствие между программным обеспечением и платой интерфейса датчика тока по коду (Jt-B)
D2034	-	Несоответствие между программным обеспечением и дополнительным блоком питания по коду (Jt-B)
D2035	-	Ошибка выполнения программы
D2036	-	(SSCNET) Произошла системная ошибка платы интерфейса
D4500	-3000	FIELD-BUS плата интерфейса не обнаружена
D4501	-3055	ABMA-PDP) Ошибка I/F модуля (XX)
D4502	-3001	FIELD-BUS-INIT) Ошибка ответа. XX
D4503	-3002	FIELD-BUS-INIT) время блокировка ответа. XX
D4504	-3005	ANYBUS) OUT/FB.CTRL время блокировки запроса. XX

ПРИЛОЖЕНИЕ 2 ЛИСТИНГ AS ЯЗЫКА (АЛФАВИТНЫЙ ПОРЯДОК)

Сокращения для каждого AS языка могут изменяться без предварительного примечания.

Буквы после назначения означают следующее:

M: мониторные команды, E: команды редактора, P: программные инструкции,

S: переключатели, F: функции, O: операторы, K: другие ключевые слова.

Наименование	Сокращение	Назначение		Формат (Параметр)	Стр.
ABORT	AB	Останавливает выполнение	M	ABORT	90
ABOVE	AB	Изменяет локтевой сустав в положение вверх	P	ABOVE	164
ABS	ABS	Выдает абсолютное значение	F	ABS (real value)	265
ABS.SPEED	ABS.SPEED	Разрешает использовать абсолютную скорость	S	...ABS.SPEED...	227
ACCEL	ACCE	Устанавливает ускорение	P	ACCEL speed ALWAYS	154
ACCURACY	ACCU	Устанавливает диапазон точности	P	ACCURACY distance ALWAYS	153
AFTER.WAIT.TMR	AF	Устанавливает как таймеры запускаются в шагах блочных программ	S	...AFTER.WAIT.TMR...	228
ALIGN	AL	Выравнивает ось Z инструмента с ближайшей базовой координатной осью	P	ALIGN	146
AND	AND	Логическое AND	O	... AND ...	232
ASC	ASC	Выдает ASCII значение	F	ASC (string, character number)	244
ATAN2	ATAN2	Выдает значение арктангенса	F	ATAN2 (real value1, real value2)	265
AUTOSTART.PC	AUTOSTART.	Автоматический запуск PC программ	S	...AUTOSTART.PC...	222
AUTOSTART2.PC	AUTOSTART2.	Автоматический запуск PC программ	S	...AUTOSTART2.PC...	222
AUTOSTART3.PC	AUTOSTART3.	Автоматический запуск PC программ	S	...AUTOSTART3.PC...	222
AUTOSTART4.PC	AUTOSTART4.	Автоматический запуск PC программ	S	...AUTOSTART4.PC...	222
AUTOSTART5.PC	AUTOSTART5.	Автоматический запуск PC программ	S	...AUTOSTART5.PC...	222
AVE_TRANS	AVE_TRANS	Выдает среднее значение	F	AVE_TRANS (transformation values1, transformation values2)	260
BAND	BAND	Бинарное AND	O	... BAND ...	234
BASE	BA	Задаст значение преобразования базовой системы координат	M	BASE transformation values	104
BASE	BA	Задаст значение преобразования базовой системы координат	P	BASE transformation values	208
BASE	BASE	Выдает значение преобразования базовой системы координат	F	BASE	260
BATCHK	BAT	Разрешает/запрещает проверку низкого напряжения батареи	M	BATCHK	120

BELOW	BE	Изменяет локтевой сустав в положение вниз	P	BELOW	164
BITS	BI	Устанавливает выходные сигналы	M	BITS start number, number of signals = value	129
BITS	BI	Устанавливает выходные сигналы	P	BITS start number, number of signals=decimal values	187
BITS	BITS	Выдает битовую комбинацию сигнала	F	BITS (starting signal number, number of signals)	241
BOR	BOR	Бинарное OR	O	... BOR ...	234
BRAKE	BRA	Немедленно останавливает движение робота	P	BRAKE	155
BREAK	BRE	Вызывает прерывание в CP движении	P	BREAK	155
BSPEED	BSP	Устанавливает блочную скорость	P	BSPEED speed	155
BXOR	BX	Бинарное XOR	O	... BXOR ...	234
BY	BY		K	SHIFT (trans BY X shift, Y shift, Z shift)	259
C	C	Изменяет программу для редактирования	E	C program name, step number	64
C1MOVE	C1	Движение круговой интерполяции	P	C1MOVE pose variable name, clamp number	149
C2MOVE	C2	Движение круговой интерполяции	P	C2MOVE pose variable name, clamp number	149
CALL	CA	Вызывает подпрограмму	P	CALL program name	168
CALLAUX	CALLAUX	Отображает экран дополнительных функций	P	CALLAUX auxiliary function number	174
CARD_COPY	CARD_COPY	Копирует программы на PC карту	M	CARD_COPY new program name = source program name + ...	77
CARD_FDEL	CARD_FDEL	Удаляет данные на PC карте	M	CARD_FDEL file name...	75
CARD_FDIR	CARD_FDIR	Отображает имена программ/переменных на PC карту	M	CARD_FDIR	73
CARD_FORMAT	CARD_FORMAT	Форматирует PC карту	M	CARD_FORMAT	81
CARD_LOAD	CARD_LOAD	Загружает содержимое PC карты в память робота	M	CARD_LOAD/Q filename	84
CARD_RENAME	CARD_REN	Изменяет имя программы на PC карте	M	CARD_RENAME new program name = existing program name	76
CARD_SAVE	CARD_SA	Записывает программы/переменные на PC карту	M	CARD_SAVE/ SEL filename =program name...	81
CARD_SAVE/ELOG	CARD_SA/ ELOG	Записывает журнал ошибок на PC карту	M	CARD_SAVE/ELOG file name	82
CARD_SAVE/P,L,R,S,A	CARD_SA/P,L,R,S,A	Записывает данные на PC карту	M	CARD_SAVE (/P) (/L)/(R)/(S)/(A) /SEL file name=program name...	82
CARD_SAVE/ROB	CARD_SA/ROB	Записывает данные робота на PC карту	M	CARD_SAVE/ROB file name	82
CARD_SAVE/SYS	CARD_SA/SYS	Записывает системные данные на PC карту	M	CARD_SAVE/SYS file name	82
CARD_VERIFY	CARD_VERIFY	Устанавливает ON/OFF	M	CARD_VERIFY mode	75

		функции проверки			
CASE	CASE	CASE структура	P	CASE number OF ... VALUE ... ANY ...END	181
CCENTER	CCENTER	Выдает центр дуги окружности	F	CCENTER (pose1, pose2, pose3)	262
CHECK.HOLD	CH	Разрешает/запрещает ввод команд от клавиатуры, когда HOLD/RUN в положении HOLD	S	... CHECK.HOLD....	218
\$CHR	\$CHR	Выдает ASCII символы	F	\$CHR (real value)	267
CHSUM	CH	Разрешает/запрещает возврат в исходное состояние ошибки неправильной проверки суммы	M	CHSUM	123
CLAMP	CLAMP	Управляет сигналами открытия /закрытия фиксаторов	P	CLAMP clamp number 1, ..., clamp number 8	196
CLOSE	CLOSE	Закрывает фиксатор схвата	P	CLOSE clamp number	159
CLOSEI	CLOSEI	Закрывает фиксатор схвата	P	CLOSEI clamp number	159
CLOSES	CLOSES	Устанавливает ON/OFF сигнал закрытия фиксатора	P	CLOSES clamp number	161
COM	COM	Бинарное дополнение	O	... COM ...	234
CONTINUE	CON	Возобновляет выполнение	M	CONTINUE NEXT	90
COPY	COP	Копирует программы в память робота	M	COPY new program name = source program name + ...	77
COS	COS	Выдает значение косинуса	F	COS (real value)	265
CP	CP	Функция непрерывности движения по траектории (CP)	S	...CP.....	218
CS	CS	Переключатель CYCLE START состояние ON/OFF	S	Switch (CS)	223
CSHIFT	CSHIFT	Возвращает сдвинутую позицию	F	CSHIFT (pose1, pose2, pose3, object pose BY shift amount)	262
CYCLE.STOP	CY	Останавливает цикл с вводом внешнего HOLD	S	... CYCLE.STOP....	218
D	D	Удаляет программные шаги	E	D step count	66
\$DATE	\$DATE	Выдает системное время	F	\$DATE (date form)	272
DECEL	DECE	Устанавливает замедление	P	DECEL speed ALWAYS	154
\$DECODE	\$DECODE	Извлекает символы	F	\$DECODE (string variable, separator character, mode)	269
DECOMPOSE	DECO	Выделяет компоненты координатной переменной	P	DECOMPOSE array variable name [element number]= pose variable name	207
DEFSIG	DEF	Отображает и изменяет программно организованные приоритетные сигналы	M	DEFSIG OUTPUT/INPUT	112
DELAY	DEL	Останавливает робот на заданное время	P	DELAY time	142
DELETE	DEL	Удаляет данные из памяти робота	M	DELETE (/P) (/L) (/R) (/S) data, ...	74
DELETE	DEL	Удаляет данные из памяти робота	P	DELETE (/P) (/L) (/R) (/S) data, ...	215
DEST	DEST	Выдает место назначения в декартовых координатах	F	DEST	254

#DEST	#DEST	Выдает место назначения в угловых координатах	F	#DEST	254
DEXT	DEXT	Выдает выбранный элемент данной позиции	F	DEXT (pose variable name, element number)	243
DISPIO_01	DIS	Изменяет режим отображения IO команд	S DISPIO_01....	225
DISTANCE	DISTANCE	Выдает расстояние	F	DISTANCE (transformation value, transformation value)	242
DLYSIG	DL	Выводит сигнал после задержки	M	DLYSIGNAL signal number time	129
DLYSIG	DL	Выводит сигнал после задержки	P	DLYSIGNAL signal number time	186
DO	DO	Выполняет одну программную инструкцию	M	DO program instruction	91
DO	DO	DO структура	P	DO ... UNTIL logical expression	178
DRAW	DRA	Передвигает робот на заданное расстояние	P	DRAW X translation, Y translation, Z translation, X rotation, Y rotation, Z rotation, speed	145
DRIVE	DRI	Передвигает один сустав	P	DRIVE joint number, displacement, speed	145
DWRIST	DW	Изменяет конфигурацию запястья	P	DWRIST	165
DX	DX	Выдает координату X	F	DX (transformation value)	243
DY	DY	Выдает координату Y	F	DY (transformation value)	243
DZ	DZ	Выдает координату Z	F	DZ (transformation value)	243
E	E	Выход из режима редактирования	E	E	68
EDIT	ED	режим редактирования	M	EDIT program number, step number	64
ELSE	EL	IF структура	P	IF ... ELSE ... END	176
ENCCHK_EMG	ENCCHK E	Устанавливает диапазон отклонения позиции робота в момент аварийного останова	M	ENCCHK_EMG	120
ENCCHK_PON	ENCCHK P	Устанавливает допустимое значение энкодера для положения робота в момент аварийного останова	M	ENCCHK_PON	121
\$ENCODE	\$ENCODE	Выдает строку, созданную при помощи данных печати	F	\$ENCODE (print data, print data)	270
END	END	FOR структура	P	FOR ... END, CASE ... END	179
ENV_DATA	ENV	Устанавливает данные об окружающей среде аппаратных средств	M	ENV_DATA	122
ENV2 DATA	ENV2	Устанавливает данные об окружающей среде программных средств	M	ENV2 DATA	123
ERESET	ERE	Сброс состояния ошибки	M	ERESET	117
ERRLOG	ERRLOG	Отображение журнала ошибок	M	ERRLOG	108
ERROR	ERROR	Состояние программной ошибки	S	Switch (ERROR)	224
ERROR	ERROR	Выдает код ошибки	F	ERROR	248

\$ERROR	\$ERROR	Выдает сообщение об ошибке	F	\$ERROR (error code)	272
\$ERRORS	\$ERRORS	Выдает сообщение об ошибке	F	\$ERRORS (error number)	272
ERRSTART.PC	ERRS	Когда произошла ошибка, выполняет PC программу	S	...ERRSTART.PC...	222
EXECUTE	EX	Выполняет программу робота	M	EXECUTE program name, execution cycles, step number	88
EXTCALL	EX	Вызывает программу, выбранную при помощи сигналов	P	EXTCALL	188
F	F	Поиск строки	E	F character string	66
FD_COPY	FD_COPY	Копирует программы на дискету	M	FD_COPY new program name = source program name + ...	77
FD_FDEL	FD_FDEL	Удаляет данные с дискеты	M	FD_FDEL file name, ...	75
FD_FDIR	FD_FDIR	Отображает имена программ/переменных на дискете	M	FD_FDIR	73
FD_FORMAT	FD_FORMAT	Форматирует дискету	M	FD_FORMAT format type	81
FD_LOAD	FD_LOAD	Загружает содержимое дискеты в память робота	M	FD_LOAD/Q filename	84
FD_RENAME	FD_RENAME	Изменяет программное имя	M	FD_RENAME new program name = existing program name	76
FD_SAVE	FD_SA	Записывает программы/переменные на дискету	M	FD_SAVE/SEL filename -program name...	81
FD_SAVE/ELOG	FD_SA/ELOG	Записывает журнал ошибок на дискету	M	FD_SAVE/ELOG file name	82
FD_SAVE/P,L,R,S,A	FD_SA/P...	Записывает данные на дискету	M	FD_SAVE (/P)/(L)/(R)/(S)/(A) /SEL file name=program name...	82
FD_SAVE/ROB	FD_SA/ROB	Записывает данные робота на дискету	M	FD_SAVE/ROB file name	82
FD_SAVE/SYS	FD_SA/SYS	Записывает системные данные на дискету	M	FD_SAVE/SYS file name	82
FD_VERIFY	FD_VERIFY	Устанавливает ON/OFF функции проверки	M	FD_VERIFY mode	75
FLOWRATE	FLOWRATE	Изменяет режим управления расходом	S	... FLOWRATE ...	226
FOR	FOR	FOR структура	P	FOR loop=start TO end STEP step program instruction END	179
FRAME	FRAME	Выдает значение преобразований для системы координат рамки	F	FRAME (x1, x2, y, origin)	254
FREE	FR	Отображает размер свободной памяти	M	FREE	102
GOTO	G	Переход на метку	P	GOTO label IF condition	167
GUNOFF	GUNOFF	Устанавливает OFF для сигналов сварочных клещей	P	GUNOFF gun number, distance	161
GUNOFFTIMER	GUNOFFTIMER	Управляет синхронизацией выходного сигнала сварочных клещей в состоянии OFF	P	GUNOFFTIMER gun number , time	161
GUNON	GUNON	Устанавливает состояние ON для сигналов сварочных клещей	P	GUNON gun number, distance	161

GUNONTIMER	GUNONTIMER	Управляет синхронизацией выходного сигнала сварочных клещей в состоянии ON	P	GUNONTIMER gun number, time	161
HALT	HA	Останавливает выполнение	P	HALT	171
HELP	HEL	Отображает листинг AS команд и инструкций	M	HELP	118
HELP/DO	HEL/DO	Отображает листинг функций	M	HELP/DO	118
HELP/F	HEL/F	Отображает листинг мониторинговых команд	M	HELP/F	118
HELP/M	HEL/M	Отображает листинг команд, используемых с MC инструкцией	M	HELP/M	118
HELP/MC	HEL/MC	Отображает листинг инструкций, используемых с DO командой	M	HELP/MC	118
HELP/P	HEL/P	Отображает листинг программных инструкций	M	HELP/P	118
HELP/PPC	HEL/PPC	Отображает листинг инструкций, используемых в PC программах	M	HELP/PPC	118
HELP/SW	HEL/SW	Отображает листинг системных переключателей	M	HELP/SW	118
HERE	HE	Запоминает текущую позицию	M	HERE pose variable name	94
HERE	HE	Запоминает текущую позицию	P	HERE pose variable name	205
HERE	HERE	Выдает координатное значение текущей позиции	F	HERE	257
#HERE	#HERE	Выдает угловое значение координат текущей позиции	F	#HERE	257
HMOVE	HM	Перемещается с гибридным движением	P	HMOVE pose variable name, clamp number	147
HOLD	HO	Останавливает выполнение	M	HOLD	90
HOLD.STEP	HOLD.STEP	Разрешает отображение шага выполнения, когда программа остановлена	S HOLD.STEP....	226
HOME	HO	Перемещается в позицию безопасности (домашнюю позицию)	P	HOME position number	144
#HOME	#HOME	Выдает значение домашней позиции	F	#HOME (home position number)	261
HSETCLAMP	HSETCLAMP	Присваивает сигнальные номера для работы фиксаторов	M	HSETCLAMP	111
I	I	Вставляет новые шаги	E	I	65
ID	ID	Отображает информацию о версиях программных обеспечений	M	ID	118
IF	IF	Переходит к метке, если состояние установлено	P	IF condition GOTO label	167
IF	IF	IF структура	P	IF ...THEN... ELSE ... END	176
IFWPRINT	IFWPRINT	Отображает строку в строковом окне, установленном при помощи дополнительной функции	M	IFWPRINT window, row, column, background color, label color = "character string", ...	138

IFWPRINT	IFWPRINT	Отображает строку в строковом окне, установленном при помощи дополнительной функции	P	IFWPRINT window, row, column, background color, label color = "character string", ...	
IGNORE	IG	Отменяет ON или ONI инструкцию	P	IGNORE signal number	191
INPUT	I	Программно созданные приоритетные входные сигналы	K	DEFSIG INPUT	112
INRANGE	INRANGE	Выдает результат проверки диапазона движения	F	INRANGE (pose variable, joint displacement values)	250
INSTR	INSTR	Выдает стартовую точку выбранной строки	F	INSTR (starting point, string 1, string 2)	246
INT	INT	Выдает целое число таким же образом как числовое выражение	F	INT (numeric expression)	248
IO	IO	Отображает состояние сигналов	M	IO/E signal number	102
IPEAKCLR	IPEAKCLR	Отображает пиковые значения тока для каждого сустава	M	IPEAKCLR	125
IPEAKLOG	IPEAKLOG	Отображает журнал значений пиковых токов	M	IPEAKLOG	124
JAPPRO	JA	Приближается к позиции по интерполированному угловому движению	P	JAPPRO pose variable name, distance	143
JDEPART	JD	Отходит от текущей позиции по интерполированному угловому движению	P	JDEPART distance	144
JMOVE	JM	Начинает интерполированное угловое движение	P	JMOVE pose variable name, clamp number	142
KILL	KI	Инициализирует программный стек	M	KILL	91
L	L	Выбирает предыдущий шаг.	E	L	65
LAPPRO	LA	Приближается к позиции по прямолинейному интерполированному движению	P	LAPPRO pose variable name, distance	143
LDEPART	LD	Отходит от текущей позиции по прямолинейному интерполированному движению	P	LDEPART distance	144
\$LEFT	\$LEFT	Выдает крайне левые символы	F	\$LEFT (string, number of characters)	267
LEFTY	LE	Изменяет конфигурацию робота в положение левой руки	P	LEFTY	164
LEN	LEN	Выдает число символов	F	LEN (string)	245
LIST	LI	Отображает данные или программные распечатки	M	LIST (/P)/(L)/(R)/(S) prog/data...	73
LLIMIT	LL	Устанавливает нижний предел движения робота	M	LLIMIT joint displacement values	103
LLIMIT	LL	Устанавливает нижний предел движения робота	P	LLIMIT joint displacement values	209
LMOVE	LM	Начинает прямолинейное интерполированное движение	P	LMOVE pose variable name, clamp number	142

LOAD	LOAD	Загружает содержимое персонального компьютера в память робота	M	LOAD/Q filename	84
LOCK	LO	Изменяет приоритет	P	LOCK priority	171
LSTRACE	LSTRACE	Отображает данные регистрирования	M	LSTRACE stepper number: logging number	78
M	M	Корректирует символы	E	M/existing characters/new characters	67
MAXVAL	MAXVAL	Выдает наибольшее значение	F	MAXVAL (real value1, real value 2, ...)	247
MC	MC	Выполняет мониторинговые команды из PC программ	P	MC monitor commands	212
MESSAGES	ME	Разрешает или запрещает вывод на терминал	S	...MESSAGES...	219
\$MID	\$MID	Выдает символы	F	\$MID (string, real value, number of characters)	268
MINVAL	MINVAL	Выдает наименьшее значение	F	MINVAL (real value1, real value 2, ...)	247
MM/MIN	MM/M	Миллиметры в минуту	K	... MM/M	
MM/S	MM/S	Миллиметры в секунду	K	... MM/S	
MOD	MOD	Остаток	O	... MOD ...	230
MSPEED	MSPEED	Выдает текущее значение мониторинговой скорости	F	MSPEED	250
MSPEED2	MSPEED2	Выдает текущее значение мониторинговой скорости	F	MSPEED2	250
MSTEP	MS	Выполняет одно движение робота	M	MSTEP program name, execution cycles, step number	89
MVWAIT	MVWAIT	Ожидает до тех пор, пока данное время или расстояние не будет достигнуто	P	MVWAIT value	170
NCHOFF	NCHOFF	Устанавливает OFF для режекторного фильтра	P	NCHOFF	211
NCHON	NCHON	Устанавливает ON для режекторного фильтра	P	NCHON	211
NEXT	N	Переходит к следующему шагу	K	CONTINUE NEXT	90
NOT	NOT	Логическое NOT	O	... NOT ...	232
NULL	NULL	Устанавливает ноль для значений преобразования базовой системы координат	F	NULL	257
O	O	Располагает курсор в текущей линии	E	O	68
OFF	OF	Устанавливает OFF для системных переключателей	M	switch name ... OFF	110
OFF	OF	Устанавливает OFF для системных переключателей	P	switch name ... OFF	210
OFF	OFF	Выдает значение FALSE (ложно)	F	... OFF ...	245
ON	ON	Устанавливает ON для системных переключателей	M	switch name... ON	110
ON	ON	Устанавливает ON для системных переключателей	P	switch name... ON	210
ON	ON	Устанавливает состояние прерывания	P	ON mode signal number CALL program name,	189

				priority	
ON	ON	Устанавливает состояние прерывания	P	ON mode signal number GOTO label, priority	189
ON	ON	Выдает значение TRUE (верно)	F	... ON ...	245
ONE	ONE	Вызывает определенную программу, если произошла ошибка	P	ONE program name	173
ONI	ONI	Устанавливает состояние прерывания	P	ONI mode signal number CALL program name, priority	189
ONI	ONI	Устанавливает состояние прерывания	P	ONI mode signal number GOTO label, priority	189
OPEINFO	OPEINFO	Отображает операционную информацию	M	OPEINFO robot number: joint number	125
OPEINFOCLR	OPEINFOCLR	Сбрасывает операционную информацию	M	OPEINFOCLR	126
OPEN	OPEN	Открывает фиксаторы	P	OPEN clamp number	159
OPENI	OPENI	Открывает фиксаторы	P	OPENI clamp number	159
OPENS	OPENS	Устанавливает ON/OFF сигнала открытия фиксатора	P	OPENS clamp number	161
OPLOG	OP	Отображает хронологию операций	M	OPLOG	109
OR	OR	Логическое OR	O	... OR ...	232
OUTPUT	O	Программно созданные приоритетные выходные сигналы	K	DEFSIG OUTPUT	112
OX.PREOUT	OX	Устанавливает генерацию синхронизации сигнала выхода	S OX.PREOUT....	219
P	P	Отображает программные шаги	E	P step count	65
PAUSE	PA	Временно останавливает выполнение	P	PAUSE	171
PCABORT	PCA	Останавливает выполнение PC программы	M	PCABORT PC program number:	276
PCABORT	PCA	Останавливает выполнение PC программы	P	PCABORT PC program number:	276
PCCONTINUE	PCC	Возобновляет выполнение PC программы	M	PCCONTINUE PC program number NEXT	278
PCEND	PCEN	Останавливает выполнение PC программы	M	PCEND PC program number: task number	277
PCEND	PCEN	Останавливает выполнение PC программы	P	PCEND PC program number: task number	277
PCEXECUTE	PCEX	Выполняет PC программу	M	PCEXECUTE PC program number, program name, execution cycle, step	275
PCEXECUTE	PCEX	Выполняет PC программу	P	PCEXECUTE PC program number, program name, execution cycle, step	275
PCKILL	PCK	Инициализирует стек PC программ	M	PCKILL PC program number:	277
PCSCAN	PCSC	Устанавливает время цикла для выполнения PC программы	P	PCSCAN time	279

PCSTATUS	PCSTA	Отображает состояние PC программы	M	PCSTATUS PC program number:	275
PCSTEP	PCSTE	Выполняет один шаг PC программы	M	PCSTEP PC program number; program name, execution cycles, step number	278
PI	PI	Выдает постоянную π	F	PI	265
PLCAIN	PLCAIN	Выдает значение входных данных в целом числе	F	PLCAIN (data number)	249
PLCAOUT	PLCAOUT	Устанавливает реальное числовое значение в данные числа	M	PLCAOUT data number = real value	124
PLCAOUT	PLCAOUT	Устанавливает реальное числовое значение в данные числа	P	PLCAOUT data number =real value	212
POINT	PO	Задаёт переменную позиции	M	POINT pose variable name= pose values, joint displacement values	94
POINT	PO	Задаёт переменную позиции	P	POINT pose variable name1= pose variable name2, joint displacement values	205
POINT/7	PO/7	Присваивает значение 7 сустава	M	POINT/7 transformation variable name = transformation values	96
POINT/7	PO/7	Присваивает значение 7 сустава	P	POINT/7 transformation variable name = transformation variable name2	206
POINT/A	PO/A	Присваивает A элемент значения	M	POINT/A transformation variable name = transformation values	96
POINT/A	PO/A	Присваивает A элемент значения	P	POINT/A transformation variable name 1= transformation variable name2	206
POINT/O	PO/O	Присваивает O элемент значения	M	POINT/O transformation variable name = transformation values	96
POINT/O	PO/O	Присваивает O элемент значения	P	POINT/O transformation variable name 1= transformation variable name2	206
POINT/OAT	PO/OAT	Присваивает A, O, T элементы значения	M	POINT/OAT transformation variable name = transformation values	96
POINT/OAT	PO/OAT	Присваивает A, O, T элементы значения	P	POINT/OAT transformation variable name 1 = transformation variable name2	206
POINT/T	PO/T	Присваивает T элемент значения	M	POINT/T transformation variable name = transformation values	96

POINT/T	PO/T	Присваивает T элемент значения	P	POINT/T transformation variable name1 = transformation variable name2	206
POINT/X	PO/X	Присваивает X элемент значения	M	POINT/X transformation variable name = transformation values	96
POINT/X	PO/X	Присваивает X элемент значения	P	POINT/X transformation variable name1 = transformation variable name2	206
POINT/Y	PO/Y	Присваивает Y элемент значения	M	POINT/Y transformation variable name = transformation values	96
POINT/Y	PO/Y	Присваивает Y элемент значения	P	POINT/Y transformation variable name1 = transformation variable name2	206
POINT/Z	PO/Z	Присваивает Z элемент значения	M	POINT/Z transformation variable name = transformation values	96
POINT/Z	PO/Z	Присваивает Z элемент значения	P	POINT/Z transformation variable name1 = transformation variable name2	206
POWER	POWER	Состояние ON/OFF переключателя MOTOR POWER	S	switch (POWER)	223
#PPOINT	#PPOINT	Дает значение угловых координат суставов	F	#PPOINT (jt1,jt2,jt3,jt4,jt5,jt6)	259
PREFETCH.SIGINS	PR	Разрешает или запрещает более раннюю обработку I/O сигналов	S PREFETCH.SIGINS....	220
PRIME	PRIM	Подготавливает программу для выполнения	M	PRIME program, execution cycles, step number	87
PRINT	PRIN	Отображает данные на терминале	M	PRINT device number: print data, ...	136
PRINT	PRIN	Отображает данные на терминале	P	PRINT device number: print data, ...	199
PRIORITY	PRIORITY	Выдает приоритетный номер	F	PRIORITY	250
PROMPT	PROMPT	Отображает сообщение на терминале и ожидает ввод подсказки	P	PROMPT device number: character string, variables	201
PULSE	PU	Устанавливает сигнал в состояние ON для данного периода времени	M	PULSE signal number, time	128
PULSE	PU	Устанавливает сигнал в состояние ON для данного периода времени	P	PULSE signal number, time	185
QTOOL	Q	Значение преобразования инструментальной системы координат для блочного программирования	S QTOOL....	220
R	R	Заменяет символы	E	R character string	67
RANDOM	RANDOM	Выдает случайное число от 0 до 1	F	RANDOM	265

REC_ACCEPT	REC	Разрешает/запрещает функцию RECORD/PROGRAM CHANGE	M	REC_ACCEPT	121
RELAX	RELAX	Устанавливает состояние OFF сигнала фиксатора (закрытие и открытие)	P	RELAX clamp number	160
RELAXI	RELAXI	Устанавливает состояние OFF сигнала фиксатора (закрытие и открытие)	P	RELAXI clamp number	160
RELAXS	RELAXS	Устанавливает состояние ON/OFF сигнала фиксатора (закрытие и открытие)	P	RELAXS clamp number	161
RENAME	REN	Изменяет программное имя	M	RENAME new program name = existing program name	76
REP_ONCE	REP	Устанавливает выполняется ли цикл повторения один раз или непрерывно	S	...REP ONCE...	221
REPEAT	REPEAT	Состояние ON/OFF переключателя TEACH/REPEAT	S	switch (REPEAT)	224
RESET	RES	Устанавливает в состояние OFF все внешние выходные сигналы	M	RESET	128
RESET	RES	Устанавливает в состояние OFF все внешние выходные сигналы	P	RESET	185
RETRACE	RETRACE	Освобождает память зарезервированную для SETTRACE	M	RETRACE	78
RETURN	RET	Возврат из вызываемой программы	P	RETURN	168
RETURNE	RETURNE	Возврат в шаг после ошибки	P	RETURNE	174
RGSO	RGSO	Состояние ON/OFF переключателя сервоуправления	S	switch (RGSO)	224
\$RIGHT	\$RIGHT	Выдает крайне правые символы	F	\$RIGHT (string, number of characters)	268
RIGHTY	RI	Изменяет конфигурацию робота в положение правой руки	P	RIGHTY	164
RPS	RP	Вызывает подпрограмму, выбранную при помощи сигналов	S RPS....	221
RUN	RUN	Состояние переключателя HOLD/RUN	S	switch (RUN)	225
RUNMASK	RU	Наложение маски на сигналы	P	RUNMASK start number, number of signals	186
RX	RX	Вращение вокруг оси X	F	RX (angle)	258
RY	RY	Вращение вокруг оси Y	F	RY (angle)	258
RZ	RZ	Вращение вокруг оси Z	F	RZ (angle)	258
S	S	Выбирает программный шаг	E	S step number	64
SAVE	SA	Записывает программы/переменные в персональный компьютер	M	SAVE/ SEL filename =program name...	81
SAVE/ELOG	SA/ELOG	Записывает журнал ошибок в персональный компьютер	M	SAVE/ELOG file name	82

SAVE/P,L,R,S,A	SA/P...	Записывает данные в персональный компьютер	M	SAVE (/P)/(L)/(R)/(S)/(A)/SEL file name=program name...	82
SAVE/ROB	SA/ROB	Записывает данные робота в персональный компьютер	M	SAVE/ROB file name	82
SAVE/SYS	SA/SYS	Записывает системные данные в персональный компьютер	M	SAVE/SYS file name	82
SCALL	SCA	Переходит к подпрограмме	P	SCALL string expression, variable	172
SCNT	SCN	Выдает сигнал счетчика, когда значение счетчика достигнуто	M	SCNT counter signal number = count up signal, count down signal, counter clear signal, counter value	130
SCNT	SCNT	Выдает сигнал счетчика, когда значение счетчика достигнуто	P	SCNT counter signal number = count up signal, count down signal, counter clear signal, counter value	192
SCNTRESET	SCNTR	Сбрасывает значение внутреннего счетчика	M	SCNTRESET counter signal number	131
SCNTRESET	SCNTR	Сбрасывает значение внутреннего счетчика	P	SCNTRESET counter signal number	193
SCREEN	SC	Управляет отображением терминала	S SCREEN....	221
SETHOME	SETH	Задает домашнюю позицию 1	M	SETHOME accuracy, HERE	108
SET2HOME	SET2	Задает домашнюю позицию 2	M	SET2HOME accuracy, HERE	108
SETPICK	SETPICK	Устанавливает время для начала управления закрытием схвата	M	SETPICK time1,..., time8	134
SETPICK	SETPICK	Устанавливает время для начала управления закрытием схвата	P	SETPICK time1,..., time8	196
SETPLACE	SETPLACE	Устанавливает время для начала управления открытием схвата1	M	SETPLACE time1,..., time8	134
SETPLACE	SETPLACE	Устанавливает время для начала управления открытием схвата1	P	SETPLACE time1,..., time8	196
SETTRACE	SETTRACE	Резервирует необходимую память для резервирования данных	M	SETTRACE step count	78
SFLK	SFLK	Устанавливает ON/OFF сигнал в данном цикле времени	M	SFLK signal number = time	131
SFLK	SFLK	Устанавливает ON/OFF сигнал в данном цикле времени	P	SFLK signal number = time	193
SFLP	SFLP	Устанавливает ON/OFFсигнал, используя установленный /сброшенный сигнал	M	SFLP output signal = set signal expression, reset signal expression	132
SFLP	SFLP	Устанавливает ON/OFFсигнал, используя установленный /сброшенный сигнал	P	SFLP output signal = set signal expression, reset signal expression	193
SHIFT	SHIFT	Устанавливает сдвиг декартовых координат X, Y, Z	F	SHIFT (trans BY X shift, Y shift, Z shift)	259
SIG	SIG	Выдает логическое AND состояния сигналов	F	SIG (signal number, ...)	240
SIGNAL	SIG	Устанавливает сигналы в	M	SIGNAL signal number....	128

		состояние ON/OFF			
SIGNAL	SIG	Устанавливает сигналы в состояние ON/OFF	P	SIGNAL signal number....	185
SIN	SIN	Выдает значение синуса	F	SIN (real values)	265
SLOW_REPEAT	SL	Устанавливает скорость повторения в режиме замедленного повторения	M	SLOW_REPEAT	121
SLOW_START	SLOW_START	Разрешает или запрещает функцию замедленного старта	S	...SLOW_START...	228
SOUT	SO	Выводит сигнал, когда состояние установлено	M	SOUT signal number = signal expression	132
SOUT	SO	Выводит сигнал, когда состояние установлено	P	SOUT signal number = signal expression	194
\$SPACE	\$SPACE	Выводит пробелы	F	\$SPACE (number of blanks)	267
SPEED	SP	Устанавливает мониторинговую скорость	M	SPEED monitor speed	87
SPEED	SP	Устанавливает программную скорость	P	SPEED speed, rotational speed, ALWAYS	152
SQRT	SQRT	Выдает значение квадратного корня	F	SQRT (real values)	265
STABLE	STA	Останавливает движение робота на заданное время	P	STABLE time	143
STATUS	STA	Отображает состояние системы	M	STATUS	100
STEP	STE	Выполняет один шаг программы	M	STEP program name, execution cycles, step number	89
STIM	STI	Устанавливает состояние ON сигнала таймера	M	STIM timer signal = input signal number, time	133
STIM	STI	Устанавливает состояние ON сигнала таймера	P	STIM timer signal = input signal number, time	195
STOP	STO	Завершает выполнение цикла	P	STOP	172
STP_ONCE	ST	Устанавливает выполнение одного шага один раз или непрерывно	S	...STP_ONCE...	222
STPNEXT	STPNEXT	Выполняет следующий шаг	M	STPNEXT	91
SWAIT	SW	Ожидает требуемое состояние сигнала	P	SWAIT signal number,...	187
SWITCH	SW	Устанавливает системные переключатели	M	SWITCH switch name...=ON (=OFF)	109
SWITCH	SW	Устанавливает системные переключатели	P	SWITCH name ... ON (OFF)	210
SWITCH	SWITCH	Отображает состояние системного переключателя	S	SWITCH (switch name)	223
SWITCH	SWITCH	Выдает состояние системного переключателя	F	SWITCH (switch name)	248
SYSDATA	SYSDATA	Выдает параметры	F	SYSDATA (keyword, opt1)	251
SYSINIT	SY	Инициализирует систему	M	SYSINIT	117
T	T	Разрешает обучение при помощи пульта ручного управления в режиме редактирования	E	T variable name	70

TASK	TASK	Выдает состояние выполнения программы	F	TASK (task number)	248
TDRAW	TDRA	Перемещает робот на заданное количество по осям инструментальной системы координат	P	TDRAW X translation, Y translation, Z translation, X rotation, Y rotation, Z rotation, speed	145
TEACH	TE	Обучает позицию	M	TEACH pose variable name	70
TEACH_LOCK	TEACH_LOCK	Состояние ON/OFF переключателя TEACHLOCK	S	switch (TEACH LOCK)	224
THEN	THEN	IF структура	K	IF logical expression THEN ... ELSE ... END	176
TIME	TI	Устанавливает и отображает дату и время	M	TIME yy-mm-dd hh:mm:ss	103
\$TIME	\$TIME	Выдает системное время	F	\$TIME	273
TIMER	TI	Устанавливает таймер	P	TIMER number = time	209
TIMER	TIMER	Выдает значение таймера	F	TIMER (timer number)	241
TO	TO	FOR структура	K	FOR ... TO ... END	
TOOL	TOOL	Задаст значение преобразования инструментальной системы координат	M	TOOL transformation values	106
TOOL	TOOL	Задаст значение преобразования инструментальной системы координат	P	TOOL transformation values	208
TOOL	TOOL	Выдает значение преобразования инструментальной системы координат	F	TOOL	261
TPLIGHT	TPLIGHT	Устанавливает задний фон на пульте ручного управления	M	TPLIGHT	124
TPLIGHT	TPLIGHT	Устанавливает задний фон на пульте ручного управления	P	TPLIGHT	213
TRACE	TRACE	Регистрирует и отслеживает программы	M	TRACE stepper number: ON/OFF	77
TRACE	TRACE	Регистрирует и отслеживает программы	P	TRACE stepper number: ON/OFF	215
TRADD	TRADD	Returns the sum of traverse axis and transformation values	F	TRADD (transformation values)	261
TRANS	TRANS	Выдает координатное значение	F	TRANS (X, Y, Z, O, A, T)	258
TRIGGER	TRIGGER	Состояние ON/OFF переключателя TRIGGER us	S	switch (TRIGGER)	223
TRSUB	TRSUB	Выдает разность между осью линейного перемещения и координатным значением	F	TRSUB (transformation values)	261
TWAIT	TW	Ожидает определенный период времени	P	TWAIT time	169
TYPE	TY	Отображает данные на терминале	M	TYPE device number: print data, ...	136
TYPE	TY	Отображает данные на терминале	P	TYPE device number: print data, ...	199
ULIMIT	UL	Устанавливает нижний предел для движения робота	M	ULIMIT joint displacement values	103
ULIMIT	UL	Устанавливает нижний предел для движения робота	P	ULIMIT joint displacement values	209

UNTIL	U	DO структура	P	DO ... UNTIL logical expression	178
UTIMER	UTIMER	Устанавливает таймер пользователя	P	UTIMER @timer variable = timer value	210
UTIMER	UTIMER	Выдает значение переменной таймера	F	UTIMER (@timer)	250
UWRIST	UW	Изменяет конфигурацию запястья	P	UWRIST	165
VAL	VAL	Выдает реальное значение	F	VAL (string, code)	246
WAIT	WA	Ожидает заданное состояние	P	WAIT condition	169
WEIGHT	WE	Устанавливает данные массы груза	M	WEIGHT load mass, x, y, z	119
WEIGHT	WE	Устанавливает данные массы груза	P	WEIGHT load mass, x, y, z	211
WHERE	W	Отображает текущее положение робота	M	WHERE display mode	101
WHILE	WH	DO структура	P	WHILE ... DO ... END	177
WS.ZERO	WS.ZERO	Изменяет технологический процесс сварки	S	... WS.ZERO...	227
WS_COMPOFF	WS_COMPOFF	Изменяет синхронизацию выхода WS сигнала	S	... WS_COMPOFF...	226
XD	XD	Вырезает шаги и помещает их в буфер	E	XD step count	68
XFER	XF	Копирует и переносит шаги	M	XFER destination program name, step number 1= source program name, step number 2, step count	76
XMOVE	XM	Двигается до сигнального изменения	P	XMOVE mode pose name TILL signal number	147
XOR	XOR	Исключение логического OR	O	XOR	232
XP	XP	Вставка содержимого буфера вставки	E	XP	69
XQ	XQ	Вставка содержимого буфера вставки в обратном порядке	E	XQ	69
XS	XS	Отображение содержимого буфера вставки	E	XS	70
XY	XY	Копируются шаги и вставляются в буфер	E	XY step count	69
ZSIGSPEC	ZSIG	Устанавливает число смонтированных сигналов	M	ZSIGSPEC	111
ZZERO	ZZE	Устанавливает обнуление данных	M	ZZERO	115
FALSE	FALSE	Выдает значение FALSE	F	... FALSE ...	245
TRUE	TRUE	Выдает значение TRUE	F	... TRUE ...	245
—	—	Вычитание	O	... — ...	230
*	*	Умножение	O	... * ...	230
/	/	Деление	O	... / ...	230
^	^	Возведение в степень	O	... ^ ...	230
+	+	Сложение	O	... + ...	230
+	+	Сцепление строк	O	... + ...	237
<	<	Меньше чем	O	... < ...	231
<=	<=	Меньше чем или равно	O	... <= ...	231
<>	<>	Не равно	O	... <> ...	231

=<	=<	Меньше чем или равно	O =< ...	231
==	==	равно	O = ...	231
=>	=>	Больше чем или равно	O => ...	231
>	>	Больше чем	O > ...	231
>=	>=	Больше чем или равно	O >= ...	231

ПРИЛОЖЕНИЕ 3 ASCII КОДЫ

ASCII character	Octal	Decimal	Hexa-decimal	Description
NULL	000	00	00	Null
SOH	001	01	01	Start of heading
STX	002	02	02	Start of text
ETX	003	03	03	End of text
EOT	004	04	04	End of transmission
ENQ	005	05	05	Enquiry
ACK	006	06	06	Acknowledge
BEL	007	07	07	Bell
BS	010	08	08	Backspace
HT	011	09	09	Horizontal tabulation
LF	012	10	0A	Line feed
VT	013	11	0B	Vertical tabulation
FF	014	12	0C	Form feed
CR	015	13	0D	Carriage return
SO	016	14	0E	Shift out
SI	017	15	0F	Shift in
DLE	020	16	10	Data link escape
DC1	021	17	11	Device control 1
DC2	022	18	12	Device control 2
DC3	023	19	13	Device control 3
DC4	024	20	14	Device control 4
NAK	025	21	15	Negative acknowledge
SYN	026	22	16	Synchronous idle
ETB	027	23	17	End of transmission block
CAN	030	24	18	Cancel
EM	031	25	19	End of medium
SUB	032	26	1A	Substitute
ESC	033	27	1B	Escape
FS	034	28	1C	File separator
GS	035	29	1D	Group separator
RS	036	30	1E	Record separator
US	037	31	1F	Unit separator
SP	040	32	20	Space

ASCII character	Octal	Decimal	Hexa- decimal	ASCII character	Octal	Decimal	Hexa- decimal
!	040	32	20	0	060	48	30
"	041	33	21	1	061	49	31
#	042	34	22	2	062	50	32
\$	043	35	23	3	063	51	33
%	044	36	24	4	064	52	34
&	045	37	25	5	065	53	35
'	046	38	26	6	066	54	36
(047	39	27	7	067	55	37
)	050	40	28	8	070	56	38
*	051	41	29	9	071	57	39
+	052	42	2A	:	072	58	3A
,	053	43	2B	;	073	59	3B
-	054	44	2C	<	074	60	3C
.	055	45	2D	=	075	61	3D
/	056	46	2E	>	076	62	3E
	057	47	2F	?	077	63	3F

ASCII character	Octal	Decimal	Hexa- decimal	ASCII character	Octal	Decimal	Hexa- decimal
@	100	64	40		140	96	60
A	101	65	41	a	141	97	61
B	102	66	42	b	142	98	62
C	103	67	43	c	143	99	63
D	104	68	44	d	144	100	64
E	105	69	45	e	145	101	65
F	106	70	46	f	146	102	66
G	107	71	47	g	147	103	67
H	110	72	48	h	150	104	68
I	111	73	49	i	151	105	69
J	112	74	4A	j	152	106	6A
K	113	75	4B	k	153	107	6B
L	114	76	4C	l	154	108	6C
M	115	77	4D	m	155	109	6D
N	116	78	4E	n	156	110	6E
O	117	79	4F	o	157	111	6F
P	120	80	50	p	160	112	70
Q	121	81	51	q	161	113	71
R	122	82	52	r	162	114	72
S	123	83	53	s	163	115	73
T	124	84	54	t	164	116	74
U	125	85	55	u	165	117	75
V	126	86	56	v	166	118	76
W	127	87	57	w	167	119	77
X	130	88	58	x	170	120	78
Y	131	89	59	y	171	121	79
Z	132	90	5A	z	172	122	7A
[133	91	5B		173	123	7B
¥	134	92	5C		174	124	7C
]	135	93	5D		175	125	7D
	136	94	5E		176	126	7E
-	137	95	5F	DEL	177	127	7F

ПРИЛОЖЕНИЕ 4 ОГРАНИЧЕНИЕ СИГНАЛЬНЫХ НОМЕРОВ

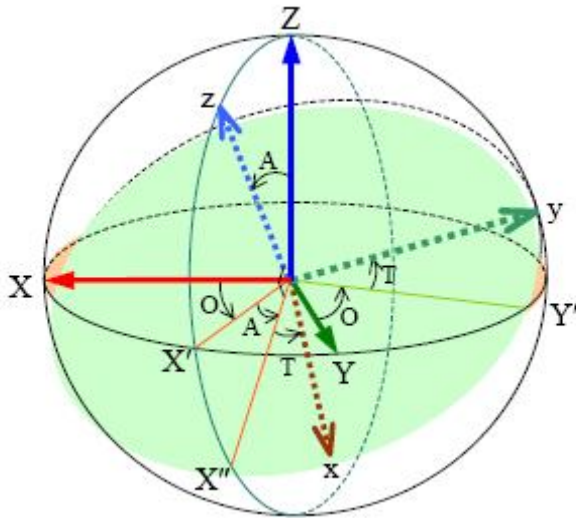
No	M, P, F*		Output Signals	Input Signals	Internal Signals
1	BITS	M P	1 to maxsig**	-----	2001 to 2256
2	BITS	F	1 to maxsig**	1001 to maxsig**	2001 to 2256
3	DEFSIG	M	1 to maxsig**	1001 to maxsig**	-----
4	DLYSIG	M P	1 to 64***	-----	2001 to 2256
5	ON, ONI	P	-----	1001 to 1064***	2001 to 2256
6	PULSE	M P	1 to 64***	-----	2001 to 2256
7	RUNMASK	P	1 to 64***	-----	2001 to 2256
8	SIGNAL	M P	1 to maxsig**	-----	2001 to 2256
9	SIG	F	1 to maxsig**	1001 to maxsig**	2001 to 2256
10	SWAIT	P	1 to maxsig**	1001 to maxsig**	2001 to 2256
11	XMOVE	P	-----	1001 to 1064***	2001 to 2256
12					
13					

ПРИМЕЧАНИЕ * M= мониторинговая команда, P= программная инструкция, F= функция

ПРИМЕЧАНИЕ ** максимальное внешних сигналов
32 (стандарт), 64 (опция), 96 (опция), 128 (опция)

ПРИМЕЧАНИЕ*** 64(1064): максимальное количество, которое может быть задано
32(1032), если установлен стандартный I/O модуль

ПРИЛОЖЕНИЕ 5 УГЛЫ ЭЙЛЕРА



Положение координатной системы $\Sigma(x, y, z)$ по отношению к базовой системе координат $\Sigma(X, Y, Z)$ в общем выражается, используя углы Эйлера O, A, T . Как показано на рисунке выше, три угла могут быть определены следующим образом. На рисунке, показанном выше две координатные системы $\Sigma(x, y, z)$ и $\Sigma(X, Y, Z)$ имеют одно и тоже начало координат.

O : угол между плоскостью Zz и плоскостью XZ

A : угол между осью z и осью Z

T : угол между осью x и осью X''

X'' ось находится в плоскости Zz и угол между этой осью и осью z равняется 90° .

Об этих трех углах можно говорить, что они представляют углы вращения, необходимые для базовой системы координат $\Sigma(X, Y, Z)$, для того чтобы совпасть с системой координат $\Sigma(x, y, z)$. Порядок вращения не может быть изменен, иначе результата будут отличаться.

1. O вращение координатной системы $\Sigma(X, Y, Z)$ вокруг оси Z (т.е. перемещение $\Sigma(X, Y, Z)$ в $\Sigma(X', Y', Z)$.)
2. A вращение координатной системы $\Sigma(X', Y', Z)$ вокруг оси Y' . (перемещение $\Sigma(X', Y', Z)$ в $\Sigma(X'', Y', z)$.)
3. T вращение координатной системы $\Sigma(X'', Y', z)$ вокруг оси z . (перемещение $\Sigma(X'', Y', z)$ в $\Sigma(x, y, z)$.)

Поэтому, это можно рассматривать в понятиях значений полярных координат. Когда точка P , которая находится на оси z на расстоянии d от начала координат, записывается как (d, A, O) , тогда O и A в этих значениях координат равны O и A , описанных выше. Направление оси z выражается этими двумя величинами.