ADEM GPP

Руководство пользователя

ADEM
CAD/CAM/CAPP

Содержание

ADEM GPP. Генератор постпроцессоров

Данный раздел содержит инструкции по использованию модуля генерации постпроцессоров для станков с УЧПУ - **ADEM GPP**.

Для лучшего понимания основных принципов работы системы и во избежание разночтений терминов и понятий перед началом работы советуем ознакомиться с разделом «Основные положения».

Если это Ваше первое знакомство с модулем **ADEM GPP**, Вам будет полезно ознакомиться с разделом «Этапы создания постпроцессора», включающим в себя «Формирование паспорта станка», «Формирование файла макрокоманд», «Формирование макета кадра», а также «Формирование файла алгоритмов». Как правило пользователь, занимающийся написанием постпроцессоров, прочитывает эти разделы только один раз.

В модуле **ADEM GPP** предусмотрен запуск процесса адаптации в режиме отладчика, что значительно облегчает процесс написания и редактировании постпроцессора. Необходимая информация об этом изложена в разделе «Отладка постпроцессора».

Описание основных команд, функций и системных переменных, используемых при написании постпроцессора, содержится в разделах «Основные команды и функции» и «Системные переменные».

Система **ADEM** дает возможность технологу при проектировании маршрута обработки создавать собственные технологические команды и даже циклы обработки, что позволяет значительно облегчить процесс создания управляющей программы. В разделе «Пользовательские команды и циклы обработки» описываются способы создания этих объектов.

Список основных транслируемых команд **CLData**, а также структуру некоторых команд Вы найдете в разделе «Приложения».

Кроме того, в разделе «Примеры» Вы обнаружите различные варианты трансляции некоторых команд **CLData**, которые, надеемся, помогут Вам в нелёгком деле - проектировании постпроцессоров.

Основные положения

В этом разделе приводится описание основных этапов работы системы, объясняются общие термины и описываются основные задачи, решаемые адаптером.

Этапы работы системы

Проектирующая часть модуля **ADEM CAM** (процессор) готовит последовательность команд обработки в универсальном виде (CLData). Программа, переводящая эту последовательность команд из формата CLData в формат конкретной стойки ЧПУ, называется процессором адаптации или адаптером. Схема работы адаптера представлена ниже.

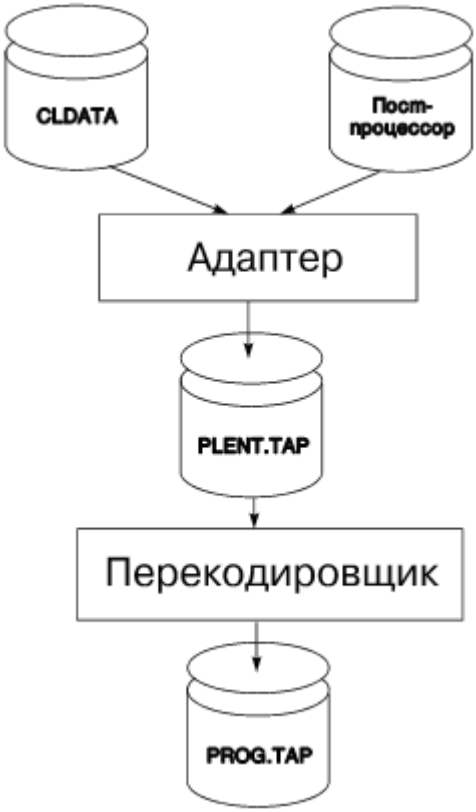


Схема работы адаптера

В своей работе адаптер использует постпроцессор на станок и формирует на рабочем диске файл с именем **PLENT.TAP**, содержащий текст управляющей программы в формате ASCII.

Стойки ЧПУ работают с различными системами кодирования символов, отличающимися от формата ASCII, например: ISO, БЦК и др. Чтобы подготовить управляющую программу для загрузки в стойку ЧПУ, необходимо перекодировать каждый символ файла **PLENT.TAP** из формата ASCII в формат этой стойки. Это делает перекодировщик, который запускается автоматически после отработки адаптера. Сформированный перекодировщиком файл **PROG.TAP** содержит кодированную управляющую программу в формате стойки ЧПУ.

Файлы **CLData**, **PLENT.TAP** и **PROG.TAP** являются временными файлами, при выходе из системы они уничтожаются. Управляющая программа должна быть записана на диск командой **"Сохранить УП как"**, которая содержится в меню **"Файл"**. По этой команде создаются два файла (основное имя задается пользователем):

- **< имя >.TAP** — УП в текстовом виде (ASCII-формат);
- **< имя >.TNC** — УП для передачи на станок в формате, определенном в постпроцессоре.

Общие термины и понятия

CLData в системе **ADEM** называется промежуточная (процессор – адаптер) информация о траектории инструмента и технологических параметрах обработки. **CLData** состоит из последовательности команд.

Каждая команда **CLData** обозначает определенное действие, имеет свой код и может иметь параметры.

Пример команд

Команда	Код	Параметры	Действие
включить ускоренное перемещение	25	без параметров	включение ускоренного перемещения
линейное перемещение	181	координаты X, Y, Z	пемещение инструмента в точку X, Y, Z
включить рабочую подачу	23	величина подачи	включение заданной подачи

Чтобы получить управляющую программу, необходимо представить последовательность действий, содержащихся в файле **CLData**, в виде кадров управляющей программы на конкретный станок.

Два термина, известные всем программистам для станков с ЧПУ:

- **Слово УП** (слово) – составная часть кадра УП, содержащая данные о параметре процесса обработки или другие управляющие данные.
- **Адрес ЧПУ** (адрес) – часть слова УП, определяющая назначение следующих за ним данных этого слова.

Как было отмечено выше, команда **CLData** может иметь параметры. Значения параметров присваиваются соответствующим **системным переменным** при реализации ее алгоритма. Например, при отработке алгоритма команды **Линейное перемещение** (код 181) значения координат точки текущего положения инструмента **X**, **Y** и **Z** присваиваются системным переменным с именами **XT**, **YT** и **ZT** (Описание системных переменных смотрите в разделе Системные переменные).

Кроме системных переменных существуют также и пользовательские переменные. Их имена должны начинаться со знака подчеркивания "**_**", но значения пользовательских переменных определяет только разработчик постпроцессора.

Часть алгоритма может выглядеть следующим образом:

```
_X=XT;
_Y=YT;
```

В приведенном выше примере **XT** и **YT** – системные переменные, а **_X** и **_Y** – пользовательские.

Примечание

Имя пользовательской переменной может состоять максимум из 8 символов, включая знак "**_**"!

Также в инструкции часто употребляются следующие термины:

Окно кадра

Окно кадра описывает слово управляющей программы и состоит из двух частей:

- **Символьная часть** соответствует адресу (может содержать последовательность символов).
- **Формат вывода** определяет вид выводимой числовой информации (например, максимальное количество выводимых символов, количество позиций после десятичной точки и т. д.).

Пример:

```
G[...]
```

В приведённом выше примере **G** — символьная часть окна, а **[...]** — условное обозначение формата вывода.

Макет кадра

Окна кадра в той последовательности, в которой они должны располагаться в кадре, содержатся в файле макета кадра. Таким образом, **макет кадра** – это структура кадра УП, то есть взаимное расположение всех возможных слов кадра и описание каждого из них.

Этот файл является частью анкеты на станок и имеет имя, например для анкеты с номером **222** - **KADR0222.ANK**. Без этого файла управляющая программа формироваться не будет, адаптер выдаст сообщение "**Нет макета кадра**".

Системы координат

В программе не используются зоны обработки

Система координат детали (СКД) устанавливается рабочей плоскостью. Адрес системы координат записывается во фразе ПРОГРАМ. В случае использования подпрограммы адрес её СК также записывается во фразе ПРОГРАМ.

В CLDATA СКД выводится как трансформ с кодом 402 (трансформ СКД в абсолютной системе координат для симулятора), после фразы ПРОГРАМ.

Если вектор оси Z системы координат конструктивного элемента (СК КЭ) не совпадает с таковым у СКД или происходит переход от токарной обработки или на неё, то в CLDATA формируется запись 10123 (СК КЭ, пересчитанная в СКД).

СК осей поворота совпадает с СКД.

В программе используются зоны обработки

Во фразу ПРОГРАМ записывается адрес СК рабочей плоскости. Он устанавливает СК осей поворота станка относительно модели детали.

Все констркутивные элементы привязаны к своей зоне. При смене зоны обработки в CLDATA выводится:

фраза ОТВОД — если инструмент не в безопасной позиции;

фраза ЗОНА (код 902) — содержит адрес зоны;

трансформ зоны (код 402) — в абсолютной СК модели для симулятора, определяет СКД;

трансформ СК осей поворота (СК операции) в СК зоны (код 405);

фраза СТОЛ;

фраза ПОВОРОТ — в том случае, если заданы углы поворота (старый вариант задания зон);

фраза СК;

фраза НТО — считается, что если не задана, то инструмент в точке зоны с координатами 0, 0, 0;

фраза БЕЗПОЗ.

Если вектор оси Z системы координат конструктивного элемента (СК КЭ) не совпадает с таковым у СК зоны или происходит переход от токарной обработки или на неё, то в CLDATA формируется запись 10123 (адаптер производит поворот все последующие перемещения будудт пересчитаны в текущую СК).

Системы кооодинат во фразах CLDATA

Фраза	Система координат
НАЧАЛО (6)	СК КЭ
ПЕРЕХВАТ (29)	СКД
ЦИКЛ (36)	в своей СК
ПОВОРОТ (40)	углы поворота в СК станка

МУЛЬТИ (41)	линейные координаты в текущей СК, угловые — в СК станка
ИДИ (181)	текущая СК
ИДИ ТОЧВКТ(182)	текущая СК
ИДИ ОКР(183)	текущая СК
ИДИ ДОБАВОЧНОЕ ТОЧ(185)	текущая СК
ИДИ ДОБАВОЧНОЕ ОКР(187)	текущая СК
ИДИ ТОЧВКТ КОР(189)	текущая СК
ИДИ КРИВАЯ(190)	текущая СК
ИДИ ДОБАВОЧНОЕ КРИВАЯ (191)	текущая СК
ТРАНСФОРМ ЗОНЫ(402)	абсолютная СК модели, переопределяет СКД
ТРАНСФОРМ ОСЕЙ ПОВОРОТА(405)	СКД (трансформ СК операции)
НТО(406)	СКД. Если задано в ПП, то в СК ПП, сначала идет трансформ ПП, а потом НТО.
БЕЗПОЗ(451)	СКД
ПХХ(452)	если ПХХ относится к отработке ТО, то в текущей СК (номер фразы равен номеру фразы ТО), иначе в СКД
USER команда (459)	текущая СК
ПЛОСК (901)	текущая СК
СТРОКА (227)	СКД

Задачи, решаемые адаптером

Адаптер решает следующие задачи:

- Определение имени станка, на который нужно получить управляющую программу.
- Определение по имени станка номера постпроцессора, который будет использоваться при формировании управляющей программы.
- Преобразование команды CLData в слова и кадры управляющей программы.
- Компоновка информации в виде единой управляющей программы.

Определение имени станка

Имя станка определяется в объекте технологического процесса **"Программная операция"** в модуле **ADEM CAM** (смотрите руководство по **ADEM CAM**). Это имя является параметром команды **CLData "Станок"** (её код 3, смотри Список основных транслируемых команд CLData).

Примечание

После изменения имени станка необходимо повторно сгенерировать **CLData**, то есть выполнить команду **"Процессор"**.

Определение по имени станка номера постпроцессора

Определение номера постпроцессора производится по имени станка, который ищется в каталоге станков. Каталог станков – текстовый файл с именем **STANKI.SKR**, расположенный по умолчанию в директории **...ADEM\NCM\POSTPR** вместе с постпроцессорами для станков.

Каталог содержит записи следующего формата:

```
| < имя станка > | < комментарий > | < номер постпроцессора > |
```

Имя станка, определенное в программной операции, должно точно соответствовать имени, записанному в каталоге станков в графе **< имя станка >**. Если адаптер не обнаружит в каталоге заданного имени, то сначала выдаст сообщение **"Станок не включен в каталог"**, а потом завершит работу, управляющая программа сформирована не будет. При добавлении постпроцессора на станок, который уже есть в каталоге, нужно изменить его название (например, необходимо добавить каталог постпроцессор на станок **АГПН-630**, а постпроцессор на такой станок уже существует – в этом случае заносим в каталог название станка **АГПН-630 v.1**).

В графе **< комментарий >** как правило записывается наименование УЧПУ, а графа **< номер постпроцессора >** содержит номер, по которому идентифицируются файлы постпроцессора.

Преобразование команд CLData

Каждой команде **CLData** поставлен в соответствие алгоритм представления ее действия в формате управляющей программы. Преобразование команды **CLData** в часть управляющей программы осуществляется в два этапа:

- Поиск по коду команды **CLData** соответствующего алгоритма преобразования.
- Реализация найденного алгоритма.
- Вывод информации в окно макета кадра.

Поиск алгоритма по коду команды CLData

Алгоритмы отображения действия команд **CLData** содержатся в файле алгоритмов. Этот файл является частью постпроцессора на станок и имеет имя, например, для постпроцессора с номером **222** - **FTPP0222.ANK**. Без этого файла управляющая программа формироваться не будет, адаптер выдаст сообщение **"Нет файла алгоритмического заполнителя"**.

Если на какую–либо команду **CLData** алгоритм не будет найден, это никак не отразится на управляющей программе.

Реализация алгоритма

Алгоритм представляет собой последовательность строк следующего формата:

```
[IF] < условие выполнения > [ELSE] < команда алгоритма >;
```

IF указывает, что команда должна быть выполнена только при соблюдении условия, следующего за **IF**. Только строки с **IF** могут иметь альтернативные строки. **ELSE** указывает, что данная строка является альтернативной для строки с **IF**, расположенной выше.

Часть алгоритма может выглядеть следующим образом:

```
3->_E;
X->XT;
Y->YT;
IF E!=1 ПНКАДР;
КАДР;
```

Вывод информации в окно макета кадра

Алгоритм формирует УП, выводя информацию в окна макета кадра. Эту информацию он получает из **CLData** посредством системных переменных.

Команда алгоритма, выводящая информацию в окно макета кадра, имеет следующий формат:

```
< номер окна в макете кадра > –> < выводимая информация >;
```

Как видно из контекста команды, для вывода информации в окно необходимо указать его порядковый номер в макете кадра.

Инструмент находится в точке с координатами X=10, Y=20, Z=30.

Макет кадра:

```
N[...] G[...] G[...] X[...] Y[...] Z[...] M[...]
```

Отработанные команды алгоритма:

```
4->XT;
5->YT;
6->ZT;
2->1;
```

Сформированная часть кадра:

```
G1 X10. Y20. Z30.
```

Примечание

Если заранее (обычно в алгоритме №1) проинициализировать номерами окон в макете кадра пользовательские переменные , то это значительно упростит написание постпроцессора. Так как не надо будет помнить все номера выводимых окон, а при выводе информации в окно кадра УП использовать ассоциативно связанные имена пользовательских переменных.

Например, если в алгоритме №1 написать строку **_XT=4;**, то, в приведенном выше примере, можно заменить строку **4–>XT;** на **_XT–>XT;**.

Компоновка информации

Управляющая программа формируется выводом информации в окна макета кадра. При этом выводимая информация поступает сначала в буфер формируемого кадра. Затем содержимое этого буфера передается в УП и образует там отдельный кадр. Передача эта происходит, например, при отработке команды алгоритма **КАДР**. Подробно механизм формирования кадров УП описан в разделе Отладка постпроцессора.

При формировании управляющей программы выполняются следующие правила:

Автоматическая нумерация кадров

Автоматическая нумерация кадров осуществляется через окно типа **НОМЕР КАДРА** (смотрите раздел Формирование окон различных типов). Формат вывода этого окна содержит интервал нумерации.

Автоматическое формирование конца кадра

Автоматически формируется окно типа **КОНЕЦ КАДРА** (смотрите раздел Формирование окон различных типов).

Сохранение информации

Кадр управляющей программы условно делится на части, каждая из которых определяет включение/выключение какой–либо функции или ее параметры. Эти функции могут быть альтернативными, когда функция отменяет действие предыдущей (например, функции G0, G1, G2, G3 в стойке FANUC), а могут быть совместно работающими, то есть могут размещаться в одном кадре (например, G, F, M в стойке FANUC). Для контроля размещения этих функций в кадрах введены два понятия:

Тип окна – это тип функции станка, реализуемый через данное окно (вспомогательная функция, подготовительная функция и т. д.).

Номер группы окна – это номер группы альтернативных функций соответствующего типа.

Рассмотрим ситуацию. В какое-либо окно заносится информация (например, в окно **G[...]** заносится "1"). При этом возможны два случая:

- Формируемый кадр не содержит альтернативной функции, тогда он дополняется заносимой в окно информацией (**G1**)
- Формируемый кадр уже содержит альтернативную функцию (например, **G2**). В этом случае информация, содержащаяся в формируемом кадре, выведется в УП отдельным кадром, начнется формирование нового кадра.

Поддержка модальности

Функции в кадрах управляющей программы бывают двух типов: модальные и локальные. Модальные действуют до их отмены альтернативной функцией, локальные действуют в пределах одного кадра. Для учета этой особенности составления управляющих программ, введено понятие **модальность окна**.

Если указано, что окно действует модально, адаптер запоминает последнее выведенное в это окно значение. Когда по алгоритму вновь приходит команда вывести информацию в это окно, адаптер сравнивает последнее выведенное и выводимое значения. Если они одинаковы, информация не выводится.

Гашение пустых кадров

Адаптер осуществляет контроль на наличие информации при выводе кадра. Если по алгоритму отрабатывается команда **КАДР** (вывод накопленной информации в кадр УП), а в формируемом кадре нет информации, команда **КАДР** игнорируется.

Этапы создания постпроцессора

Постпроцессор состоит из четырех частей:

Паспорт станка — общие данные по станку и правилам программирования.

Макрокоманды — информация об обработке адаптером команд CLData, для реализации которых необходимо выполнить несколько дополнительных команд CLData.

Макет кадра — структура кадра управляющей программы: взаимное расположение всех возможных окон кадра и описание каждого из них.

Алгоритмы — алгоритмы представления команд CLData в виде кадров и слов управляющей программы.

Что такое макет кадра и файл алгоритмов, мы рассмотрели в разделе Задачи, решаемые адаптером. Дополнительную информацию о них Вы сможете найти в разделах Формирование макета кадра и Формирование файла алгоритмов.

Паспорт станка

Паспорт станка — это набор вопросов и возможные варианты ответов о станке и правилах программирования для него.

Примеры вопросов:

- тип оборудования;
- возможность программного управления охлаждением;
- наличие кругового интерполятора;
- точность аппроксимации.

Макрокоманды

Иногда возникает необходимость при трансляции какой–либо из команд CLData отработать несколько дополнительных команд. Наиболее часто эта ситуация возникает с командами CLData **«Загрузить инструмент»** (код 35) и **«Конец управляющей программы»** (код 4).

Например, система закончила обработку текущего объекта в точке с координатами **X=35.5**, **Y=70**, **Z=–30** (последней была команда **«Линейная интерполяция»**) и сгенерировала команду на загрузку нового инструмента для обработки следующего технологического объекта, то есть файл CLData содержит команды:

```
3[181] Идти в точку/
-74.794878 42.749822 -10.000000;
4 [301] Переход/ Фрезеровать;
4 [35] Инструмент/Фреза R 5.000000 Позиция 2;
```

Возникла ситуация, когда отработавший инструмент еще находится в зоне обработки, а команда CLData требует его заменить. В этом случае, чтобы сменить инструмент, нужно отвести его в позицию смены инструмента на ускоренной подаче, выключить при отводе корректор на длину, выключить шпиндель, выключить охлаждение и только тогда провести смену инструмента. Поэтому команде **«Загрузить инструмент»**(код 35) необходимо поставить в соответствие например следующую последовательность команд:

1. «Включить ускоренное перемещение» (код 25)
2. «Отвести инструмент» (код 28)
3. «Выключить корректор по оси Z» (код 709)
4. «Выключить охлаждение» (код 700)
5. «Выключить шпиндель» (код 701)
6. «Загрузить инструмент» (код 35)

Таким образом, команда CLData **«Загрузить инструмент»** будет заменена макрокомандой, а шесть перечисленных команд будут подкомандами этой макрокоманды.

Макрокоманды находятся в файле макрокоманд, который является необязательной частью анкеты и имеет имя, например, для постпроцессора с номером **222** - **MCOM0222.ANK**.

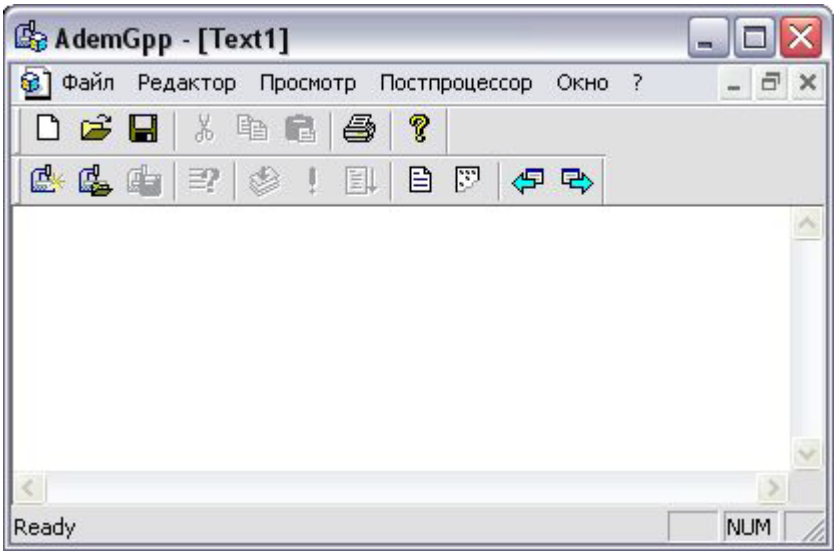
Действия пользователя при написании постпроцессора

Запуск модуля ADEM GPP

Запустить модуль **ADEM GPP** можно двумя способами:

- Если запущена программа **ADEM**, в меню **«Модуль»** выберите **«Adem GPP»**.
- Нажмите кнопку **ПУСК** на **"Панели задач"** операционной системы **Windows**. В дополнительном меню выберите **Программы->ADEM->Adem GPP** и нажмите левую кнопку мыши.

В результате на экране должно появиться главное окно программы.

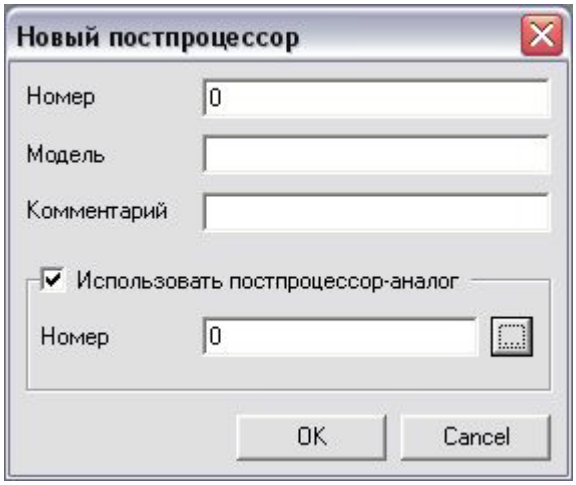


Диалоговое окно «ADEM GPP»

Генератор постпроцессоров **Adem GPP** позволяет выполнить следующие действия:

Создать новый постпроцессор

На панели инструментов **«Постпроцессор»** нажмите кнопку **«Создать постпроцессор»**  . Откроется диалог **«Новый постпроцессор»**.



Диалоговое окно «Новый постпроцессор»

Поле **«Номер»** должно содержать номер постпроцессора.

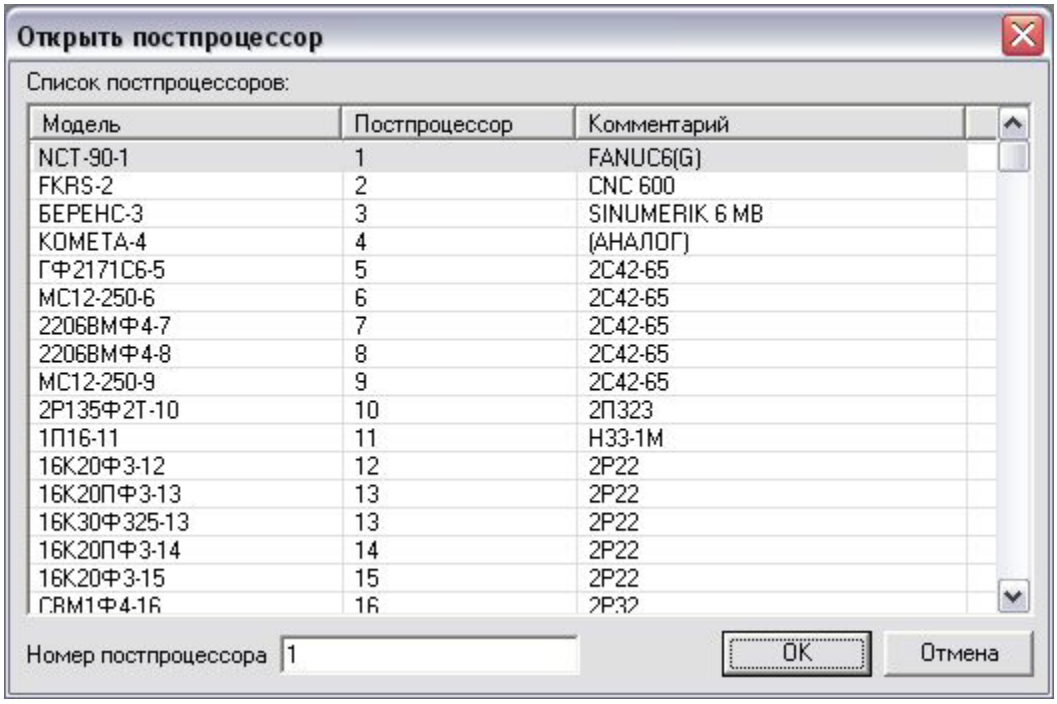
Поле **«Модель»** должно содержать наименование оборудования.

Поле **«Номер»** должно содержать наименование УЧПУ, которое установлено на данном оборудовании.

Для создания постпроцессора можно использовать любой постпроцессор, разработанный ранее. Для этого установите флажок **«Использовать постпроцессор-аналог»** и в поле **«Номер»** введите номер постпроцессора-аналога. Также можно выбрать постпроцессор из списка, если нажать кнопку возле поля **«Номер»**. После заполнения диалога нажмите клавишу **ОК**.

Открыть существующий постпроцессор

Нажмите кнопку **«Открыть постпроцессор»** . Появится диалог **«Открыть постпроцессор»**.



Диалоговое окно «Открыть постпроцессор»

Из списка выберите нужный постпроцессор и нажмите **ОК**. Также можно просто ввести номер в поле **«Номер постпроцессора»** и нажать **ОК**.

Сохранить постпроцессор

Для того, чтобы сохранить все изменения в постпроцессоре, нужно:

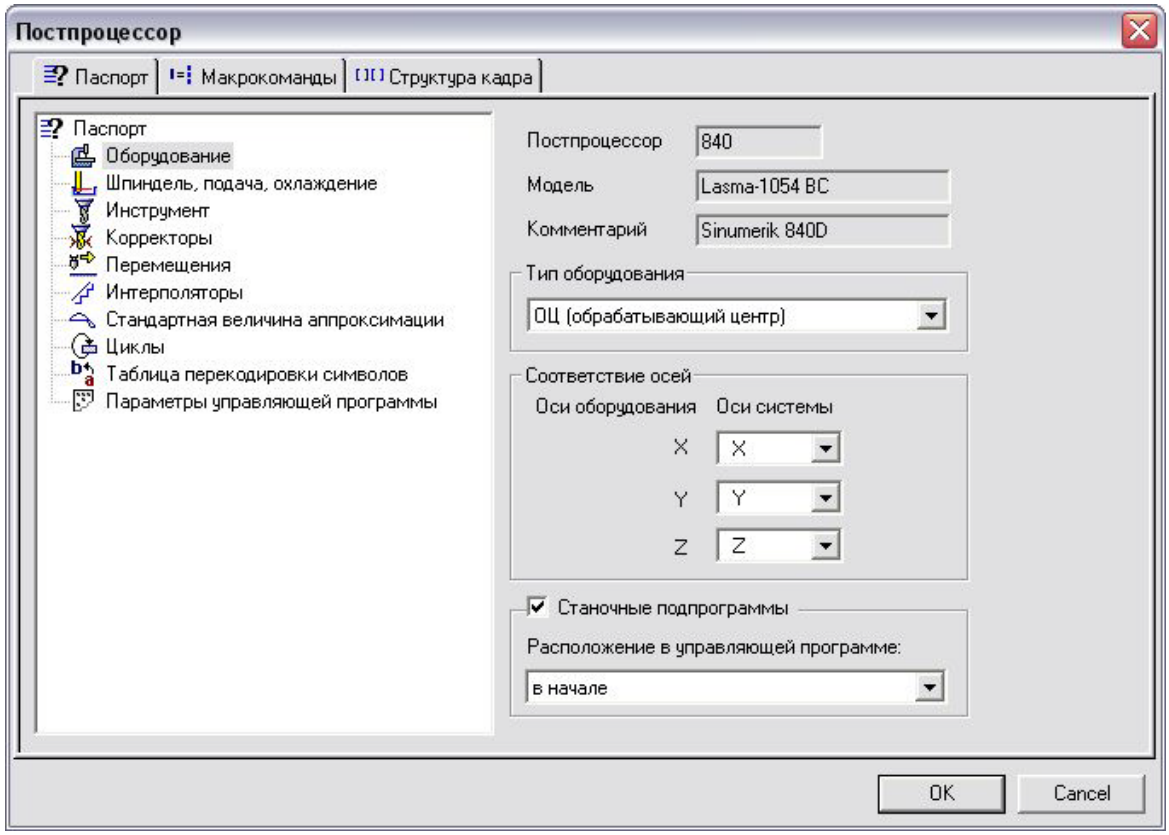
- либо нажать на кнопку **«Трансляция»** .
- либо нажать на кнопку **«Сохранить постпроцессор»** .

Примечание

Если Вы нажимаете кнопку **«Трансляция»** , то система не только сохранит все изменения, но и проведет синтаксический контроль алгоритмов.

Заполнение паспорта станка, создание макрокоманд и формирование макета кадров

Для того, чтобы заполнить паспорт станка, создать макрокоманду или внести изменения в макет кадра нажмите кнопку **«Параметры»**  . Появится диалог «Постпроцессор».



Диалоговое окно «Постпроцессор»

Более подробно процесс заполнения этого диалога описан в разделах Формирование паспорта станка, Формирование файла макрокоманд и Формирование макета кадра

Для перемещения между открытыми окнами программы (окно просмотра **"CLData"**, окно просмотра результатов и т.д.) можно использовать кнопки  - просмотр предыдущего окна программы и  - просмотр следующего окна программы.

Иногда, в процессе разработки нового постпроцессора, бывает необходимо написать несколько вариантов постпроцессора. Например, для того, что бы сравнить результат, получаемый разными способами, или иметь несколько вариантов формирования управляющей программы (ISO-кодировка, кодировка в формате внутреннего языка УЧПУ и т.д.).

Для этого нет необходимости создавать несколько разных постпроцессоров. Так как, отличаться эти постпроцессоры будут только файлом алгоритмов (паспорт станка, макет кадров и файл макрокоманд будут одинаковыми в этом случае будут одинаковыми), достаточно просто иметь несколько вариантов алгоритмического описания. И при необходимости использовать тот или другой файл алгоритмов.

Формирование нового файла алгоритмов

Для того, чтобы создать новый файл алгоритмов на панели инструментов **«Стандартная»** нажмите кнопку **«Создать»**  . Откроется окно с пустым текстовым файлом.

Открытие ранее созданного файла алгоритмов

Для того, чтобы открыть ранее созданный файл алгоритмов нажмите кнопку **«Открыть»**  . Откроется окно с текстовым файлом.

Сохранить изменения в файле алгоритмов

Для того, чтобы сохранить изменения в текущем файле алгоритмов нажмите кнопку **"Сохранить"**  . Текущий текстовый файлом будет сохранен с именем **"< имя файла алгоритмов >.pst"**.

Работа с буфером обмена

Нажмите кнопку **"Вырезать"**  чтобы вырезать текст и поместить в буфер.

Нажмите кнопку **"Копировать"**  чтобы поместить текст в буфер.

Нажмите кнопку **"Вставить"**  чтобы поместить текст из буфер в окно алгоритма.

Информация о системе

Нажмите кнопку **"Информация"**  чтобы отобразить информацию о программе, номер версии и **copyright**.

Распечатка файлов постпроцессора

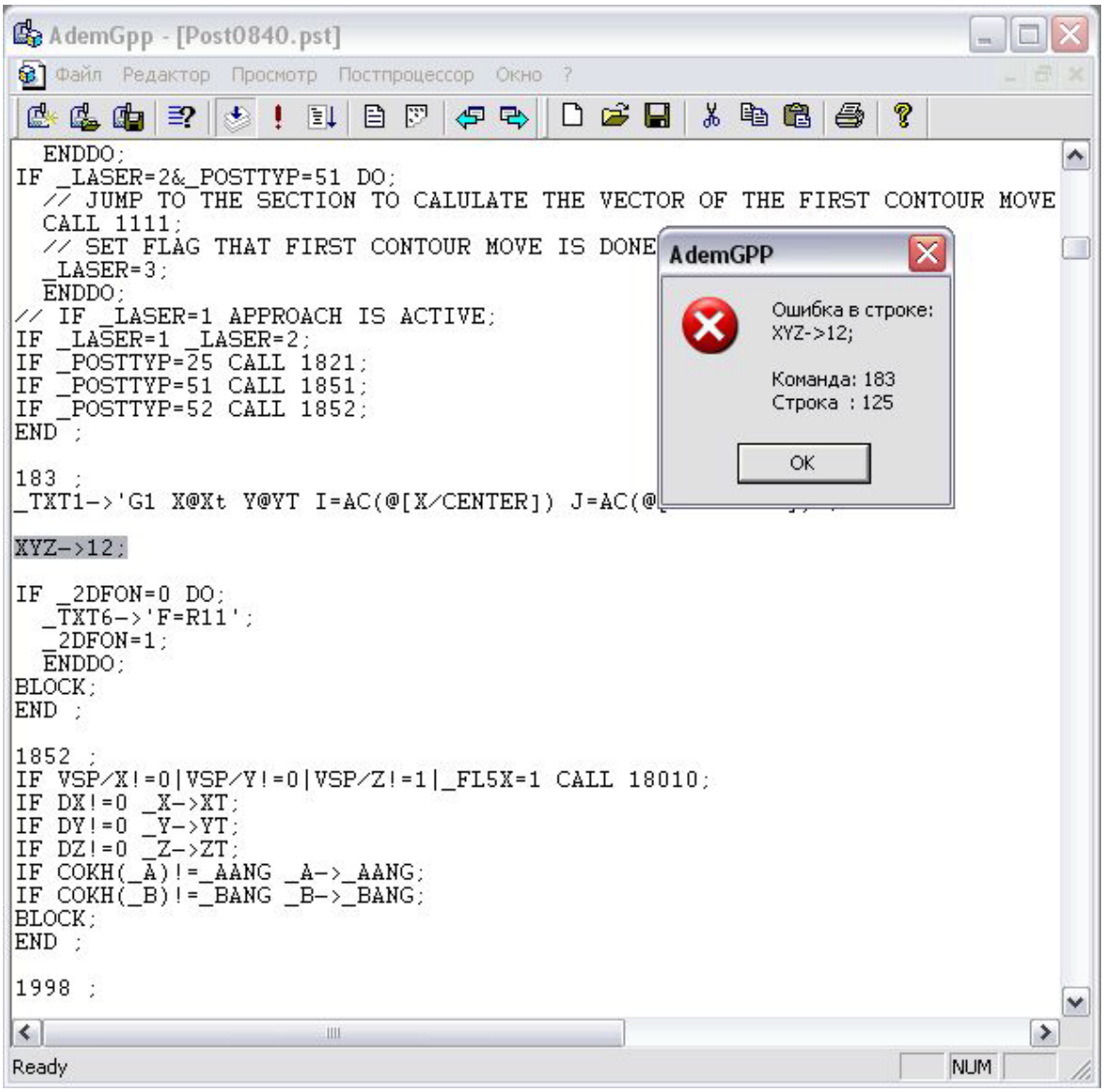
Для того ,чтобы вывести на печать текст файла алгоритмов необходимо выполнить следующие действия:

- Для настройки вашего принтера выберите из меню **«Файл»** пункт **«Настройка печати»**.
- Для просмотра печатаемого текста выберите из меню **«Файл»** пункт **«Просмотр печати»**.
- Распечатайте текст файла алгоритмов, выбрав из меню **«Файл»** пункт **«Печать»**, либо на панели иснтрументов нажмите кнопку **«Печать»**  .

Трансляция файла алгоритмов

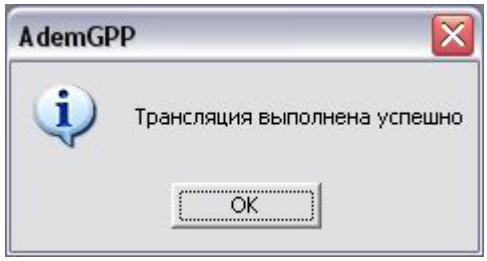
При помощи трансляции текстовый файл алгоритма переводится в код, понятный адаптеру, а также происходит синтаксический контроль текста алгоритмов и выполняется сохранение постпроцессора. Для выполнения трансляции нажмите кнопку  .

Если при трансляции система обнаружит какую-либо синтаксическую ошибку (например, неправильно написанное имя системной переменной), то процесс трансляции будет прерван, на экране появится сообщение об ошибке и строка содержащая ошибку будет выделена.



В ходе трансляции файла обнаружена ошибка

Если ошибок обнаружено не будет, появится сообщение об успешном завершении трансляции.



Трансляция завершена без ошибок

Просмотр результатов работы постпроцессора

Adem GPP формировать и просматривать текст управляющей программы, а также текст **CLData**.

Для того, чтобы посмотреть в **Adem GPP** исходный файл **CLData**, нажмите кнопку  . При этом автоматически откроется окно просмотра **CLData**.

Для того, чтобы посмотреть в **Adem GPP** результат работы редактируемого постпроцессора (сформированную УП), нажмите кнопку  . При этом автоматически откроется окно просмотра УП.

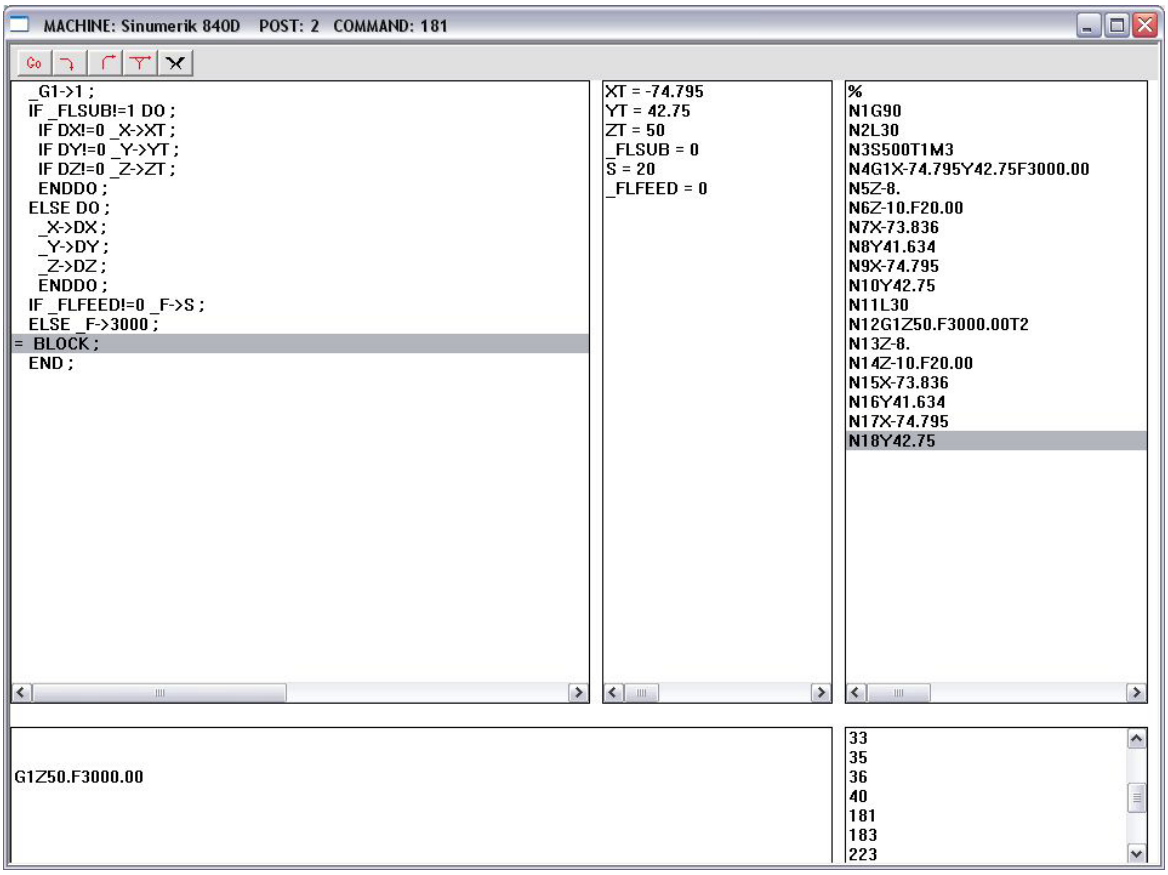
Примечание

Если постпроцессор не был предварительно оттранслирован, перед формированием УП система автоматически запустит процесс трансляции файла алгоритмов и выполнит сохранение постпроцессора.

Отладка постпроцессора

Для облегчения процесса проектирования постпроцессоров в системе реализована возможность построчной трансляции команд **CLData**.

Последовательно отрабатываются команды **CLData**, при этом Вы имеете возможность следить за значениями всех переменных и видеть как формируется каждый кадр УП. Для перехода в режим отладки постпроцессора необходимо нажать на кнопку **«Отладка»**  . На экране появится рабочее окно программы-отладчика.



Диалоговое окно программы-отладчика

Примечание

Если постпроцессор не был предварительно оттранслирован, перед запуском режима отладки система автоматически запустит процесс трансляции файла алгоритмов и выполнит сохранение постпроцессора.

В заголовке окна находится следующая информация:

- Имя станка (**MASHINE: Sinumerik 840D**)
- Код отрабатываемой команды **CLData** (**COMMAND: 181**)
- Номер отлаживаемого постпроцессора (**POST: 2**)

Окно отладчика разделено на 5 рабочих областей.

Левая верхняя область

В данной области отображается алгоритм, который отрабатывается в данный момент. Обрабатываемая в текущий момент строка подсвечивается (**= BLOCK;**).

Примечание

Знак **«=»** , стоящий в начале отрабатываемой строки алгоритма, означает точку останова. Точки останова назначаются следующим образом:

1. Подведите курсор к строке, в которой требуется установить точку останова.
2. Двойным нажатием левой клавиши мыши установите точку останова.

Средняя верхняя область
В область выводятся значения системных и пользовательских переменных, используемых в алгоритмах.

Для того, чтобы добавить в поле имя переменной, значение которой требуется контролировать:

1. Нажмите клавишу **"Insert"** на клавиатуре.
2. Введите имя переменной или арифметическое выражение.
3. Нажмите клавишу **"Enter"**.

Правая верхняя область
В данной области отображается формируемая управляющая программа.

Левая нижняя область
В данной области отображается содержимое буфера обмена (фактически, это формируемый кадр).

Правая нижняя область
В данную область выводится список алгоритмов

Для того, чтобы увидеть текст алгоритма (например, для того, чтобы поставить точку останова):

1. Выбрать в списке нужный номер алгоритма.
2. Подвести к нему курсор и дважды нажать левую кнопку мыши.

Режимы работы отладчика
Отладчик может работать в двух режимах:

- **Автоматическая отработка алгоритмов** до конца формирования УП или до первой встретившейся точки останова. Она выполняется при нажатии кнопки  или при нажатии клавиши **F8** на клавиатуре.
- **Пошаговая отработка алгоритмов.** Она выполняется при нажатии кнопки  . Все точки останова при этом игнорируются.

Также существуют два дополнительных режима отработки алгоритмов:

- **Отработка алгоритма без его трансляции** — включается при нажатии кнопки  . Этот режим позволяет отследить выполнение каждого алгоритма без его построчного выполнения.
- **Отработка алгоритмов без отработки вызовов по команде CALL** — включается при нажатии кнопки  . Этот режим позволяет при отработке алгоритма не заходить в другие алгоритмы, которые вызываются по команде **CALL**.

Для прекращения работы программы-отладчика используется кнопка  .

Примечание

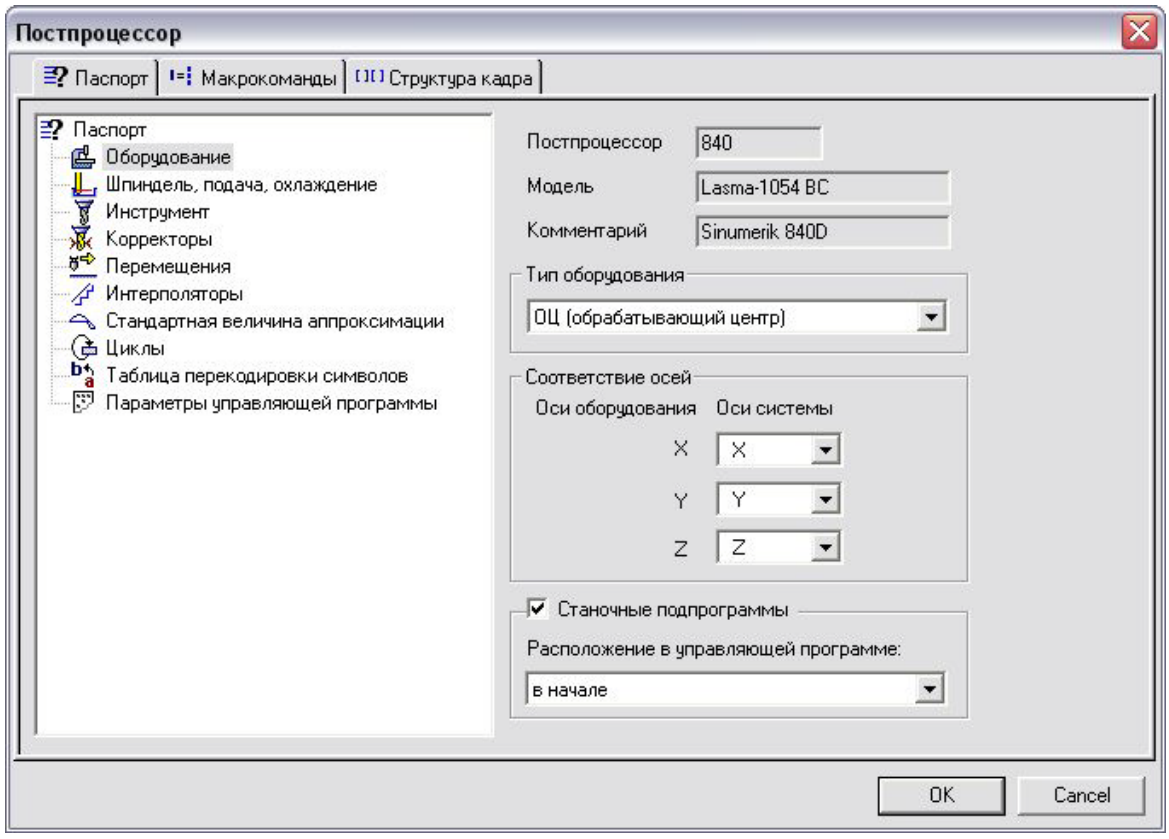
Если завершение работы было осуществлено по кнопке  , то все введенные для контроля имена переменных и арифметические выражения сохраняются и будут восстановлены при следующем запуске программы-отладчика. Это правило будет действовать до прекращения сеанса работы **Adem GPP**.

Формирование паспорта станка

Паспорт станка является частью постпроцессора и имеет имя, например, для постпроцессора с номером **222** - **FANK0222.ANK**. Без этого файла управляющая программа формироваться не будет, адаптер выдаст сообщение **"Нет файла паспорта станка"**.

Действия пользователя при формировании паспорта станка

Запустите модуль подготовки и отладки постпроцессоров. Для этого нажмите кнопку **«Параметры»**  . Откроется окно с активизированной закладкой **«Паспорт»**.



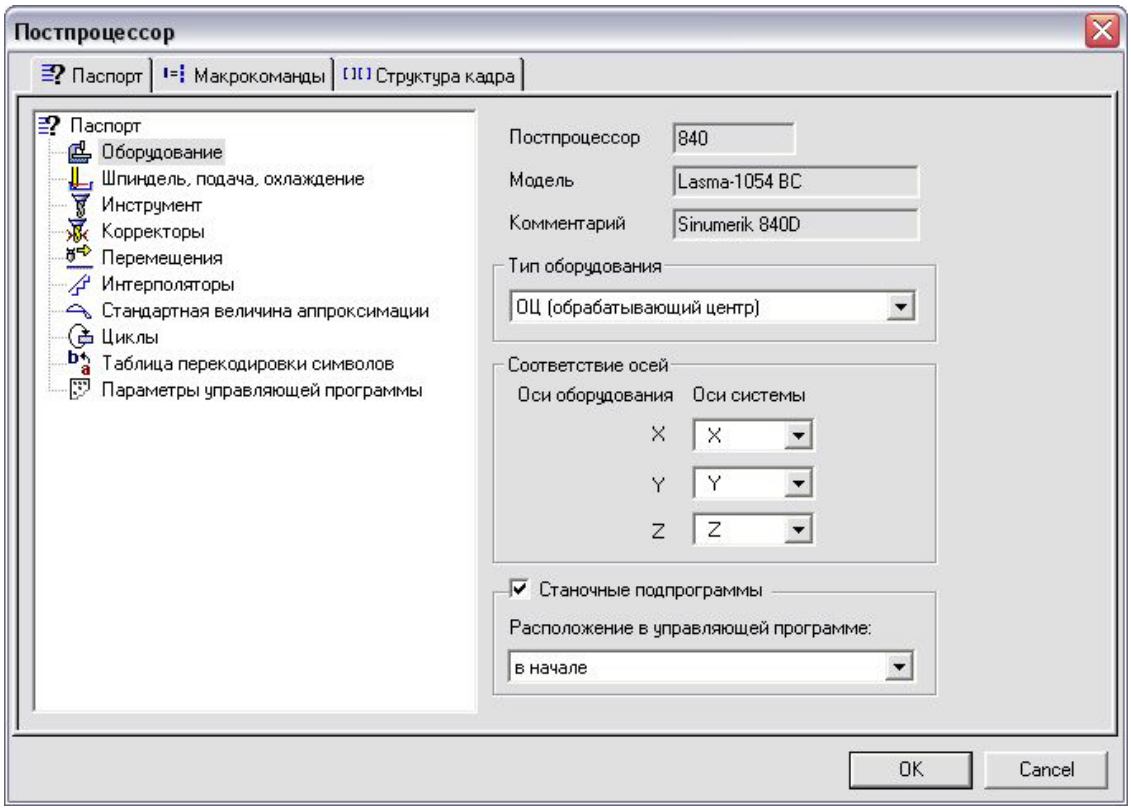
Диалоговое окно «Постпроцессор»

Всю необходимую информацию для заполнения паспорта Вы можете найти в документации к Вашему станку: в паспорте станка или инструкции по программированию.

При корректировке паспорта система выдает на экран дерево разделов паспорта. Пользователь выбирает какой-либо раздел и вводит информацию по нему.

Содержимое паспорта станка

Паспорт станка устанавливает правила, по которым адаптером будет формироваться управляющая программа для данного типа оборудования с ЧПУ.



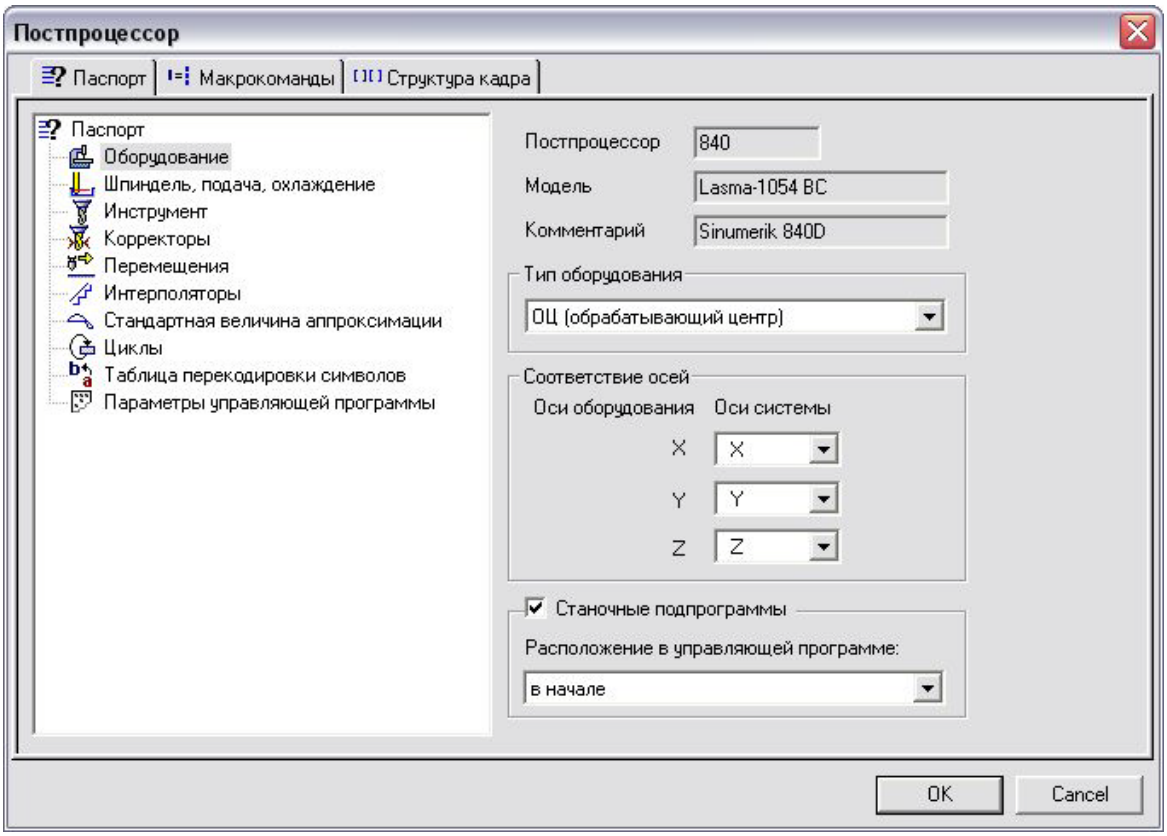
Вкладка «Паспорт» диалогового окна «Постпроцессор»

Паспорт станка включает следующие разделы:

- Оборудование
- Шпиндель, подачи, охлаждение
- Инструмент
- Корректоры
- Перемещения
- Интерполяторы
- Стандартная величина аппроксимации
- Циклы
- Таблица перекодировки символов
- Параметры управляющей программы

Оборудование

Раздел паспорта станка **«Оборудование»** содержит ощие сведения об оборудовании.



Раздел «Оборудование» паспорта станка

Постпроцессор

В поле **«Постпроцессор»** указывается номер постпроцессора. Его можно указать либо в момент создания постпроцессора, либо в любой момент можно отредактировать файл **"stanki.skr"**, указав номер в поле **"NUMBER OF POST".**

Модель

В поле **«Модель»** указывается наименование станка. Его можно указать либо в момент создания постпроцессора, либо в любой момент можно отредактировать файл **"stanki.skr"**, указав номер в поле **"MACHINE"**.

Комментарий

В поле **«Комментарий»** указывается наименование УЧПУ. Его можно указать либо в момент создания постпроцессора, либо в любой момент можно отредактировать файл **"stanki.skr"**, указав номер в поле **"REMARKS"**.

Тип оборудования

Adem GPP предлагает выбрать из нескольких типов оборудования:

- Токарное
- ОЦ (обрабатывающий центр)
- Фрезерное
- Сверлильное
- Пресс
- Фрезерное (только 2.5 координаты)
- EDM
- EDM (2 контура)
- EDM (точка и вектор)

В зависимости от выбранного типа оборудования, система **ADEM CAM** будет формировать различные типы перемещений. Например, для 4-х координатной электроэрозионной обработки особенно важно в каком виде в файле CLData содержатся данные о перемещении инструмента из точки в точку. Система **ADEM CAM** может формировать следующие типы перемещений:

- **координаты основной головки + координаты дополнительной головки** - это тип оборудования **EDM** (2 контура)
- **координаты основной головки + угол наклона проволоки** - это тип оборудования **EDM**
- **координаты основной головки + вектор угла наклона проволоки** - это тип оборудования **EDM** (точка и вектор)

Соответствие осей систем координат детали и станка

Группа параметров определяющих соответствие между координатными осями оборудования и системы **ADEM CAM**. Рассмотрим пример:

Установленное в паспорте соответствие осей

Оси оборудования	Оси системы
X	Y
Y	-Z
Z	X

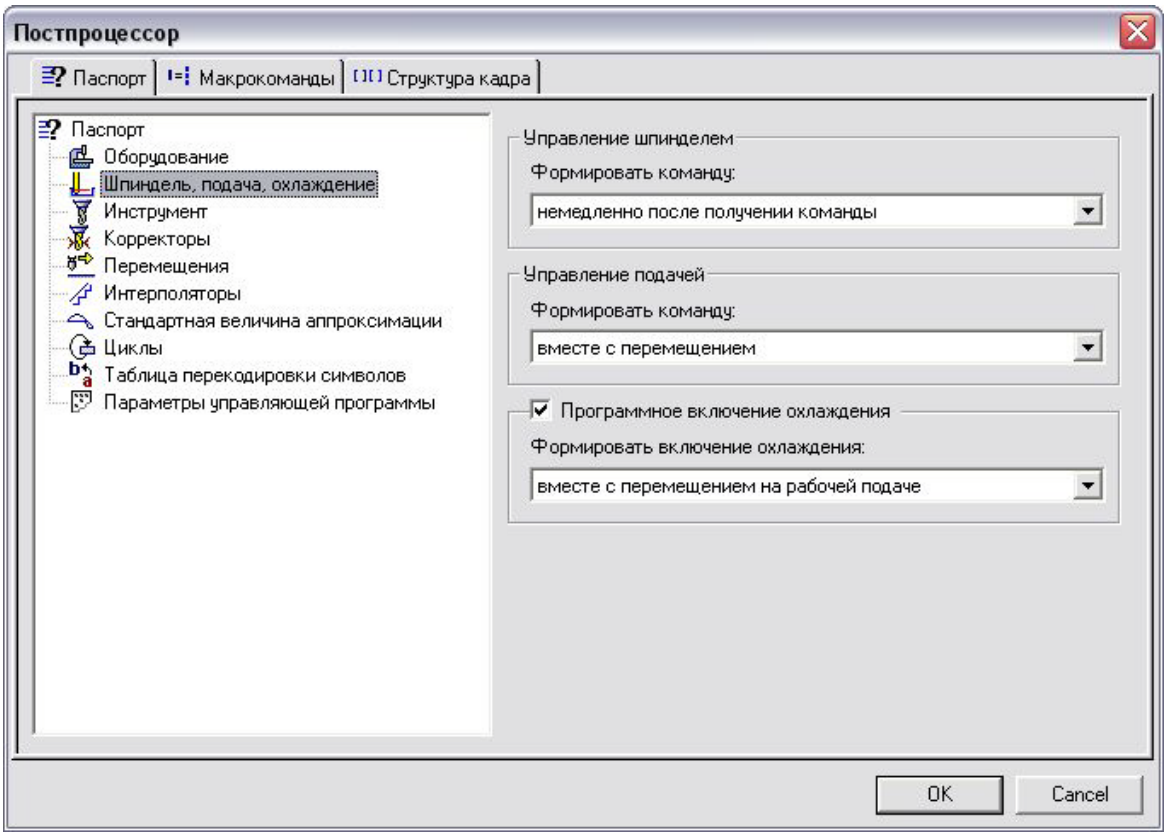
Пусть в CLData существует перемещение в точку с координатами X=100, Y=200, Z=300. Тогда, на основании установленного выше соответствия осей, адаптер сформирует перемещение в точку с координатами X=200, Y=-300, Z=100.

Станочные подпрограммы

Раскрывающийся список позволяет задать положение текста подпрограммы в управляющей программе.

Шпиндель, подача, охлаждение

В данном разделе устанавливаются правила управления шпинделем, подачей и охлаждением.



Раздел «Шпиндель, подача, охлаждение» паспорта станка

Управление шпинделем

Данный параметр определяет правила включения шпинделя. Из раскрывающегося списка можно выбрать следующие варианты:

- **Вместе с перемещением** — после получения команды CLData на включение шпинделя адаптер «придерживает» её до появления команды линейной или круговой интерполяции.
- **Немедленно после получения команды** — адаптер формирует команду на включение шпинделя сразу после получения команды из CLData.

Управление подачей

Данный параметр определяет правила включения рабочей подачи. Из раскрывающегося списка можно выбрать следующие варианты:

- **Вместе с перемещением** — после получения команды CLData на включение подачи адаптер «придерживает» её до появления команды линейной или круговой интерполяции.

- **Немедленно после получения команды** — адаптер формирует команду на включение подачи сразу после получения команды из **CLData**.

Управление охлаждением

Данный параметр определяет правила включения СОЖ. Из раскрывающегося списка можно выбрать следующие варианты:

- **Вместе с перемещением** — после получения команды CLData на включение СОЖ адаптер «придерживает» ее до появления команды линейной или круговой интерполяции.

- **Вместе с перемещением на рабочей подаче** — адаптер после получения команды CLData на включение СОЖ "придерживает" ее до появления команды линейной или круговой интерполяции на рабочей подаче.

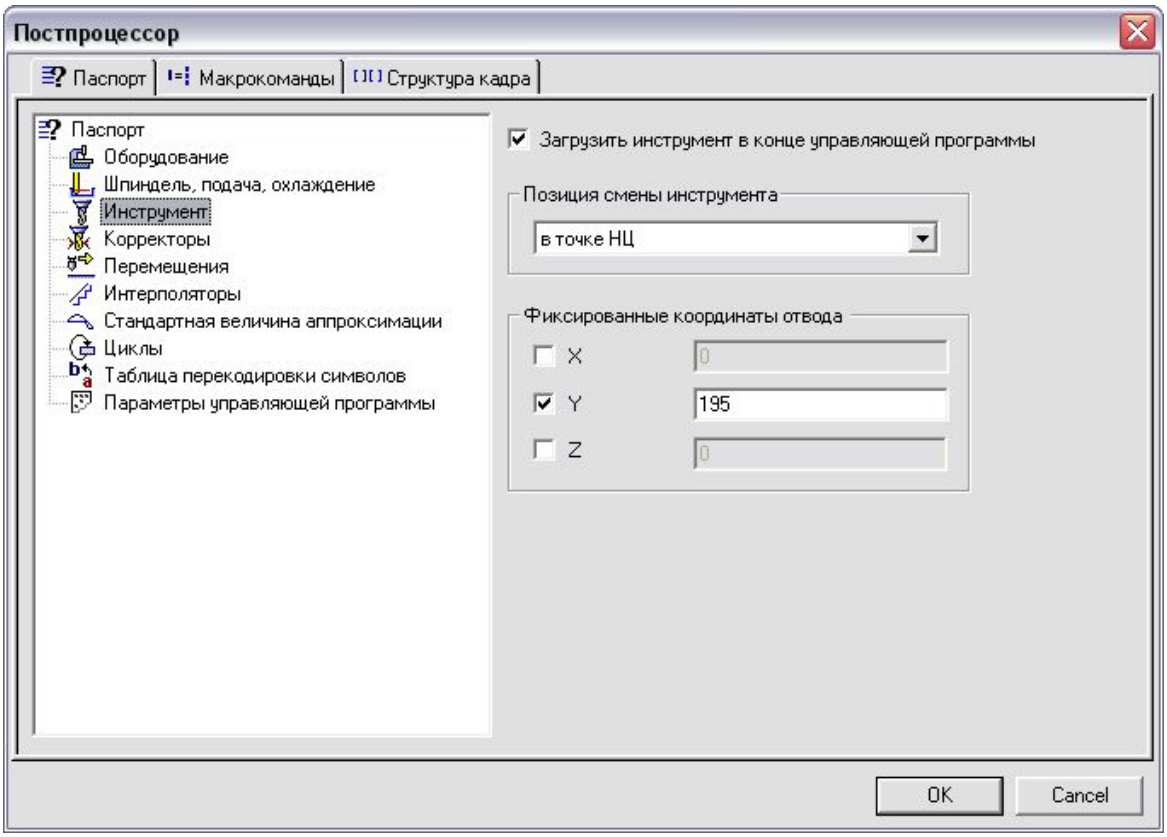
- **Немедленно после получения команды** — адаптер формирует команду на включение СОЖ сразу после получения команды из CLData.

Примечание

Если какой-либо из приведённых выше параметров не включен (отсутствует флажок), адаптер будет игнорировать команду CLData на включение СОЖ.

Инструмент

В данном разделе устанавливаются правила работы с режущим инструментом.



Раздел «Инструмент» паспорта станка

Загрузить инструмент в конце управляющей программы

Данный параметр устанавливает условия загрузки инструмента при формировании конца УП.

Если флажок установлен, то при загрузке последнего инструмента следующим инструментом будет установлен первый инструмент данной УП.

Если флажок не установлен, то при загрузке последнего инструмента номером следующего инструмента будет ноль.

Позиция смены инструмента

Данный параметр устанавливает правила отвода инструмента в позицию смены. В раскрывающемся спискепредставлены следующие варианты:

- **В любой точке** — при смене инструмента адаптер не формирует отвод инструмента при смене, даже если в макрокоманде смены инструмента стоит команда «Отвод».
- **В плоскости НЦ по X** — при смене инструмента адаптер формирует отвод инструмента в координату X начала цикла.

- **В плоскости НЦ по Y** — при смене инструмента адаптер формирует отвод инструмента в координату Y начала цикла.
- **В плоскости НЦ по Z** — при смене инструмента адаптер формирует отвод инструмента в координату Z начала цикла.
- **В плоскости НЦ по XY** — при смене инструмента адаптер формирует отвод инструмента в координату XY начала цикла.
- **В плоскости НЦ по XZ** — при смене инструмента адаптер формирует отвод инструмента в координату XZ начала цикла.
- **В плоскости НЦ по ZY** — при смене инструмента адаптер формирует отвод инструмента в координату ZY начала цикла.
- **В точке НЦ** — при смене инструмента адаптер формирует отвод инструмента в точку начала цикла или безопасную позицию, определенную в маршруте обработки.

Фиксированные координаты отвода

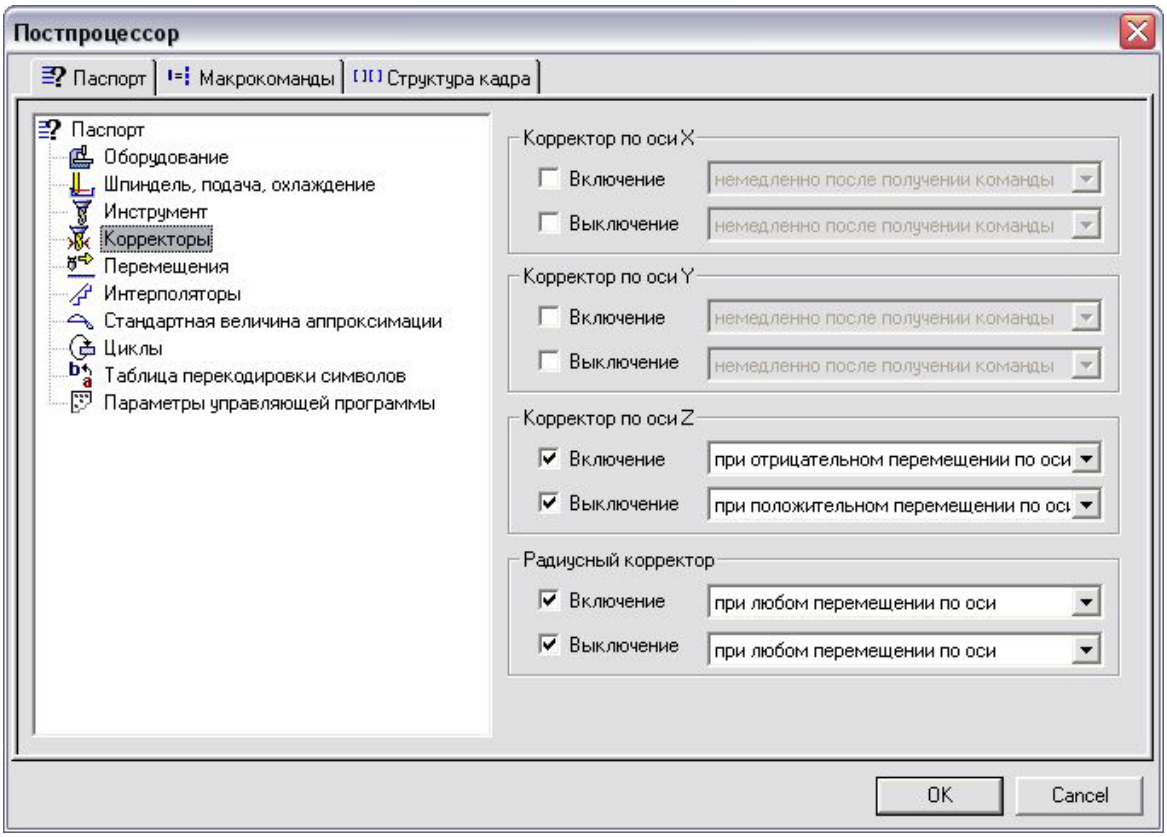
Группа параметров, которая позволяет установить фиксированные координаты отвода по осям X, Y, Z. Для ввода значения необходимо установить флажок возле соответствующей координаты.

Примечание

Если фиксированные координаты отвода не определены, инструмент отводится в точку начала цикла или безопасную позицию, определенную в маршруте обработки.

Корекрторы

В данном разделе паспорта станка устанавливаются правила работы с корректорами.



Раздел «Корректоры» паспорта станка

Включение линейных корректоров по осям X, Y и Z

Группа параметров, определяющих правила включения и выключения линейных корректоров по соответствующим осям. Предпочтительную стратегию включения можно выбрать из раскрывающегося списка для соотвтетствующей оси:

- **Немедленно после получения команды** — адаптер формирует команду на включение/выключение соответствующего корректора сразу после получения команды из CLData.
- **При положительном перемещением по оси** — после получения команды CLData включения/выключения соответствующего корректора адаптер «придерживает» её до появления команды линейной интерполяции в положительном направлении заданной оси.
- **При отрицательном перемещением по оси** — адаптер после получения команды CLData включения/выключения соответствующего корректора, "придерживает" ее до появления команды линейной интерполяции в отрицательном направлении заданной оси.

- **При любом перемещением по оси** — адаптер после получения команды CLData включения/выключения соответствующего корректора, «придерживает» её до появления команды линейной интерполяции в любом направлении заданной оси.

Включение радиусного корректора

Данный параметр определяет правила включения и выключения радиусного корректора. Различают два варианта включения/выключения:

- **Немедленно после получения команды** — адаптер формирует команду на включение/выключение радиусного корректора сразу после получения команды из CLData.

- **При любом перемещением по оси** — адаптер после получения команды **CLData** включения/выключения радиусного корректора, «придерживает» её до появления команды линейной или круговой интерполяции в любом направлении по осям X и Y.

Примечание

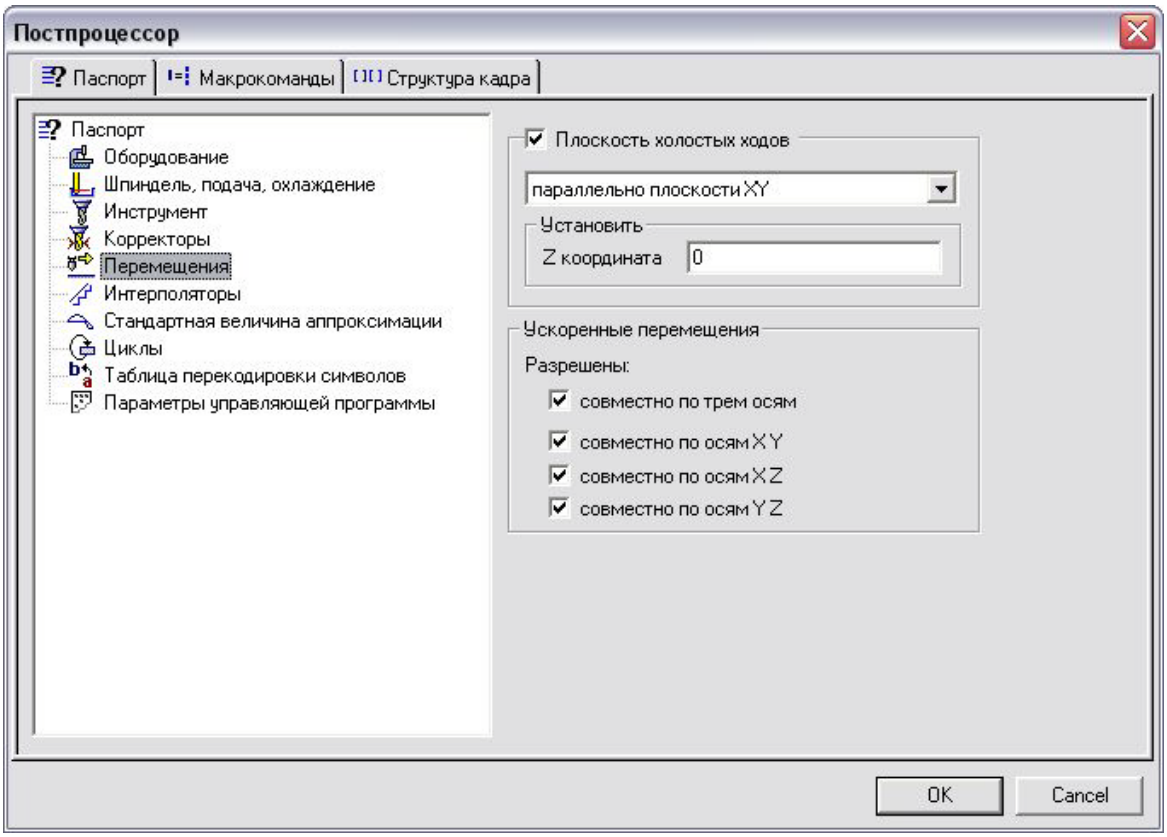
Если включение/выключение какого-либо корректора не определено (соответствующий флажок не поставлен в диалоговом окне), адаптер будет игнорировать команду CLData на включение соответствующего корректора.

Примечание

Команда включения/выключения корректора всегда приходит перед перемещением, на котором этот корректор необходимо включить/выключить.

Перемещения

В данном разделе устанавливаются правила пермещения инструмента.



Раздел «Перемещения» паспорта станка

Плоскость холостых ходов

Группа параметров, определяющих правила отвода инструмента в плоскость холостых ходов (плоскость безопасности). Плоскость, в которую будет отведён инструмент, выбирается из раскрывающегося списка:

- **Плоскость XY** – адаптер формирует команду отвод инструмента в плоскость XY на указанную ниже координату Z.
- **Плоскость XZ** – адаптер формирует команду отвод инструмента в плоскость ZY на указанную ниже координату X.
- **Плоскость ZY** – адаптер формирует команду отвод инструмента в плоскость ZY на указанную ниже координату Y.

Примечание

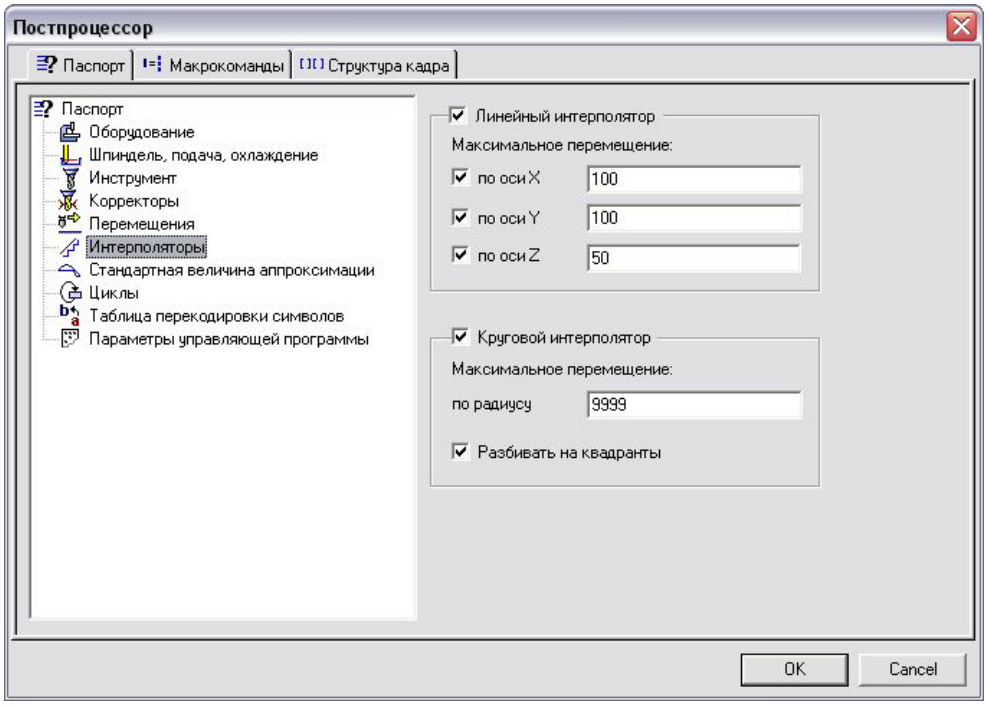
Если параметр не включен (отсутствует флажок), адаптер сформирует отвод инструмента согласно правилам, определённым в маршруте обработки.

Ускоренные перемещения

Группа параметров, определяющих правила формирования перемещений на холостом ходу совместно по нескольким осям.

- **Совместно по трем осям** - адаптер формирует ускоренные перемещения сразу по трем линейным осям.
- **Совместно по осям XY** - адаптер формирует ускоренные перемещения только по осям XY. При появлении перемещения по трём осям, система разделит его на две части: перемещение по осям XY и перемещение по Z.
- **Совместно по осям XZ** - адаптер формирует ускоренные перемещения только по осям XZ. При появлении перемещения по трём осям, система разделит его на две части: перемещение по осям XZ и перемещение по Y.
- **Совместно по осям ZY** - адаптер формирует ускоренные перемещения только по осям ZY. При появлении перемещения по трём осям, система разделит его на две части: перемещение по осям ZY и перемещение по X.

В данном разделе паспорта станка устанавливаются правила работы с интерполяторами.



Раздел «Интерполяторы» паспорта станка

Линейный интерполятор

Группа параметров, определяющих величину максимальных перемещений по линейным осям.

Если при перемещении из точки в точку разница координат будет превышать указанную максимальную величину, адаптер разобьет это перемещение на несколько равных по длине перемещений, приращения координат в которых не будут превышать указанное значение.

Если максимальное перемещение по оси равно нулю, оно считается не заданным и контролироваться системой не будет. То есть, если на станке нет ограничения по величине перемещений вдоль линеных осей (не путать с максимальными и минимальными величинами координат), эту группу параметров можно не включать.

Круговой интерполятор

Группа параметров, определяющих некоторые правила формирования круговых интерполяций.

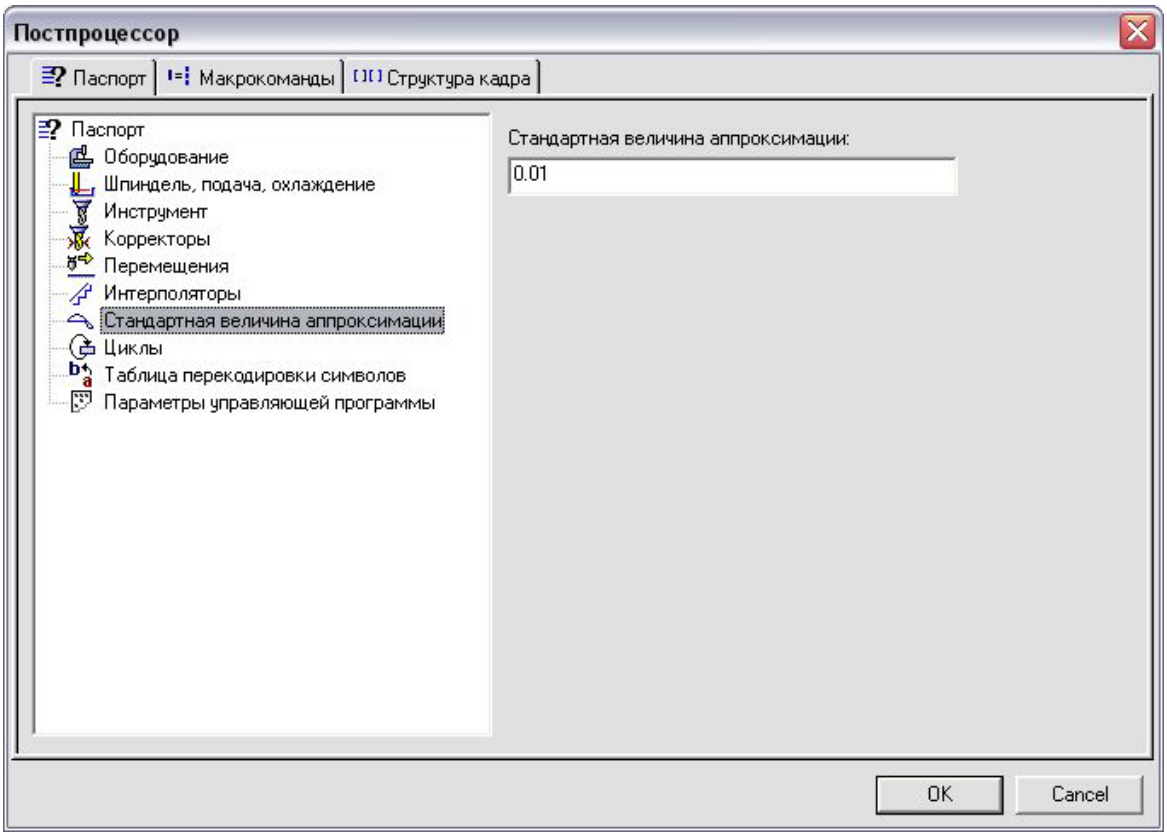
Если станок не имеет кругового интерполятора, все круговые перемещения аппроксимируются отрезками. Для включения кругового интерполятора необходимо установить соответствующий флажок.

Разбивать на квадранты — если оборудование позволяет выполнить круговую интерполяцию только в пределах одного квадранта, а круговое перемещение в CLData проходит через несколько квадрантов, адаптер разобьет его на несколько перемещений, каждое из которых будет лежать в пределах одного квадранта. Для включения разбивки на квадранты необходимо установить соответствующий флажок.

Максимальный радиус интерполяции — если радиус кругового перемещения в CLData будет превышать заданное значение, адаптер аппроксимирует дугу отрезками со стандартной точностью, определенной в разделе «Стандартная величина аппроксимации».

Стандартная величина аппроксимации

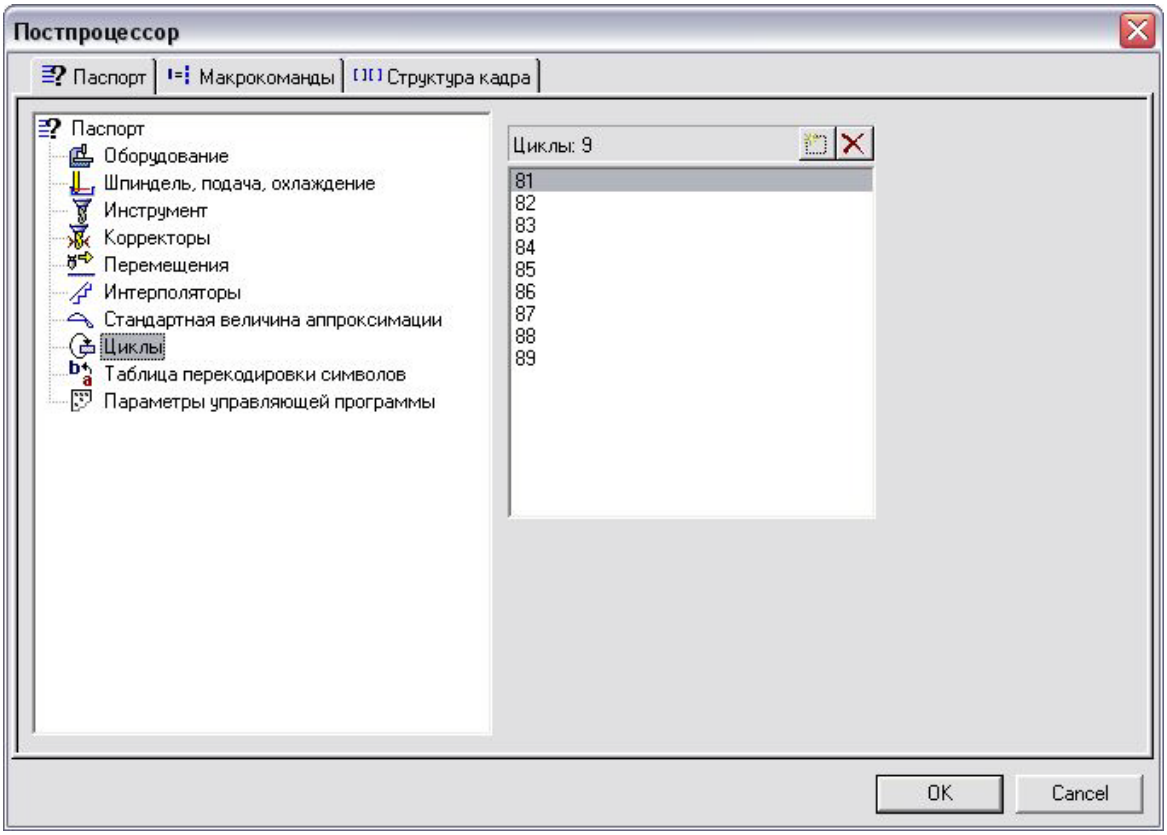
В данном разделе паспорта станка устанавливается стандартная величина аппроксимации.



Раздел «Стандартная величина аппроксимации» паспорта станка

Циклы

В данном разделе паспорта станка устанавливается перечень номеров стандартных сверлильно-расточных циклов.



Раздел «Циклы» паспорта станка

Циклы

В поле содержится перечень номеров стандартных сверлильно-расточных циклов, реализованных на данном станке с ЧПУ.

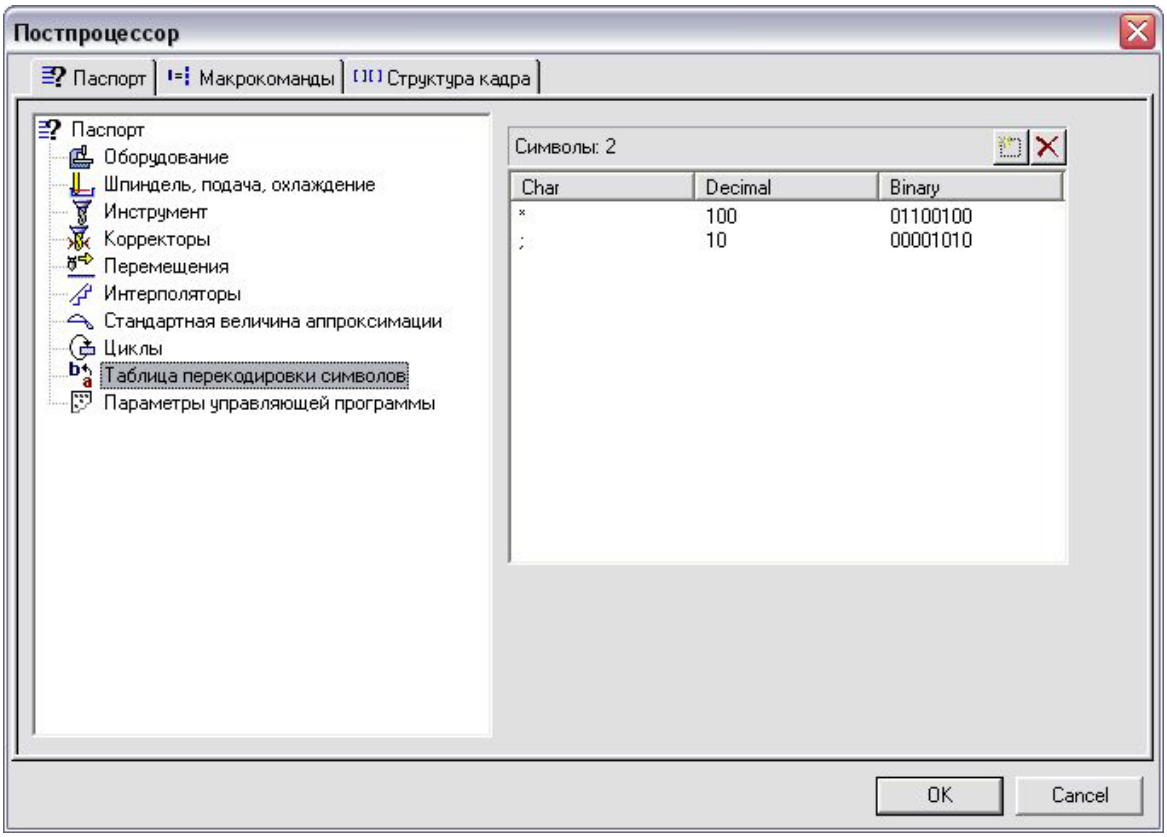
Номера циклов указываются в соответствии с разделом «Стандартные сверлильно-расточные циклы», описанном в документации на модуль ADEM CAM.

Если в CLData встретится цикл (команда **«Цикл»**, код 36), номера которого не будет в данном списке, адаптер вместо отработки команды сформирует последовательность команд для обработки данного цикла на основе макропроцедуры с именем **"mp< номер цикла >.txt"**. Все макропроцедуры располагаются в каталоге по адресу **...ADEM/ncm/MPR**.

Примечание

В системе ADEM существует возможность создавать пользовательские циклы. Указывать номера этих циклов в паспорте не нужно. Отработка пользовательских циклов происходит в соответствующем алгоритме (36 алгоритм) по их номеру.

В данном разделе паспорта станка содержится перечень символов управляющей программы, которые необходимо перекодировать при формировании файла "PROG.TAP". Данная таблица используется при всех типах кодировок, за исключением ASCII.



Раздел «Таблица перекодировки символов» паспорта станка

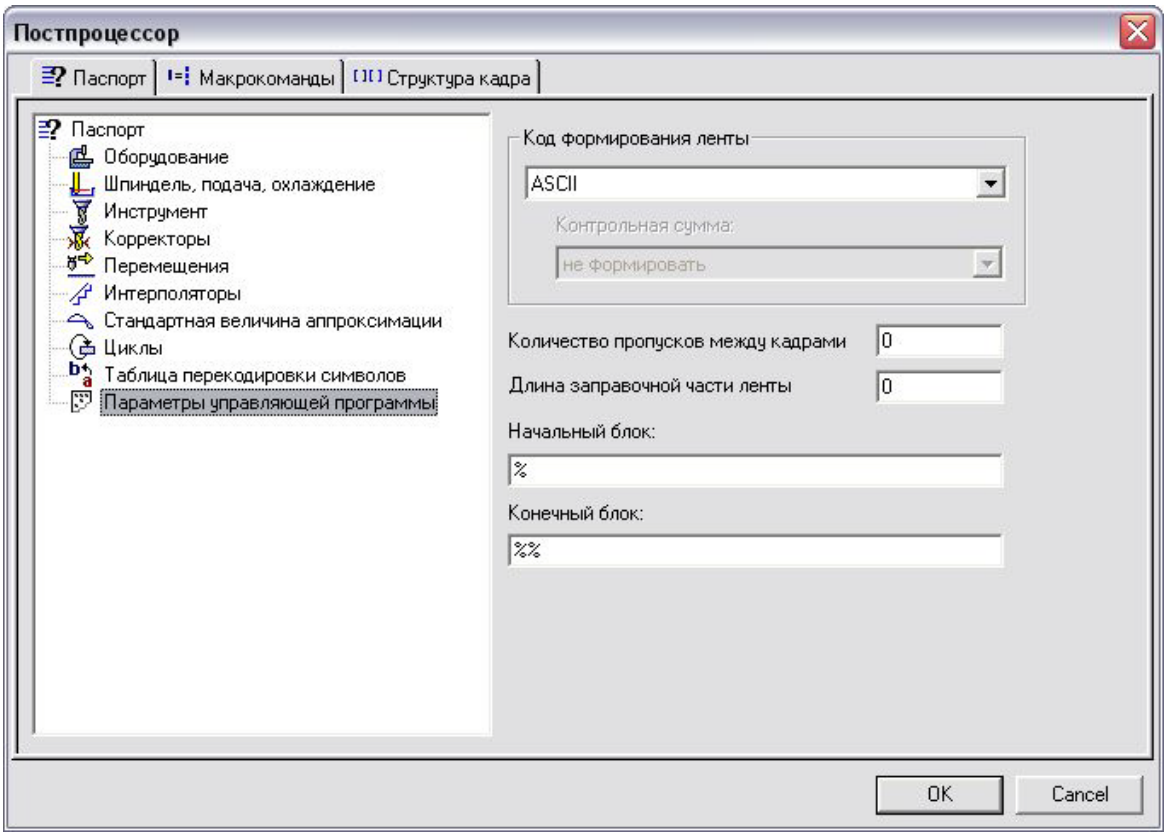
Char — символ, который необходимо перекодировать.

Decimal — десятичное число, содержащее код символа.

Binary — бинарное число, содержащее код символа.

Параметры управляющей программы

В данном разделе паспорта станка устанавливаются общие параметры управляющей программы.



Раздел «Параметры управляющей программы» паспорта станка

Код формирования ленты

Данный параметр устанавливает тип кодировки файла **"PROG.TAP"**.

Управляющая программа сначала формируется в ASCII-кодах, то есть создается обычный текстовый файл с именем **"PLENT.TAP"**. Затем адаптер автоматически перекодирует этот файл в форму, воспринимаемую станком или перфоратором.

Существует три варианта кодировки управляющих программ:

- **ISO четный** — перекодировка производится из **ASCII** в коды **ISO** с контролем по четности.
- **ISO нечетный** — перекодировка производится из **ASCII** в коды **ISO** с контролем по нечетности. Этот тип кодировке встречается крайне редко.
- **Произвольный** — правила перекодировки определяются «Таблицей перекодировки символов» и контролем суммы.

Примечание

При произвольном варианте кодирования УП можно выбрать способ формирования контрольной суммы:

- **не формировать** — контрольная сумма формироваться не будет.
- **формировать по модулю 10** — контрольная сумма будет формироваться таким образом, чтобы общая сумма символов была кратна 10.
- **формировать по четности** — контрольная сумма будет формироваться таким образом, чтобы общая сумма символов была четной.
- **формировать по нечетности** — контрольная сумма будет формироваться таким образом, чтобы общая сумма символов была нечетной.

Количество пропусков между кадрами

Данный параметр устанавливает количество двоичных нулей, которые выведутся в файл при перекодировке, после каждого кадра.

Длина заправочной части ленты

Данный параметр определяет, какое количество двоичных нулей должно записаться в начало файла при перекодировке, чтобы длина пустой ленты была бы равна заданной. Длина задается в метрах.

Начальный блок УП

Это последовательность символов, которую необходимо вывести в начале управляющей программы.

Конечный блок УП

Это последовательность символов, которую необходимо вывести в конце управляющей программы.

Формирование файла макрокоманд

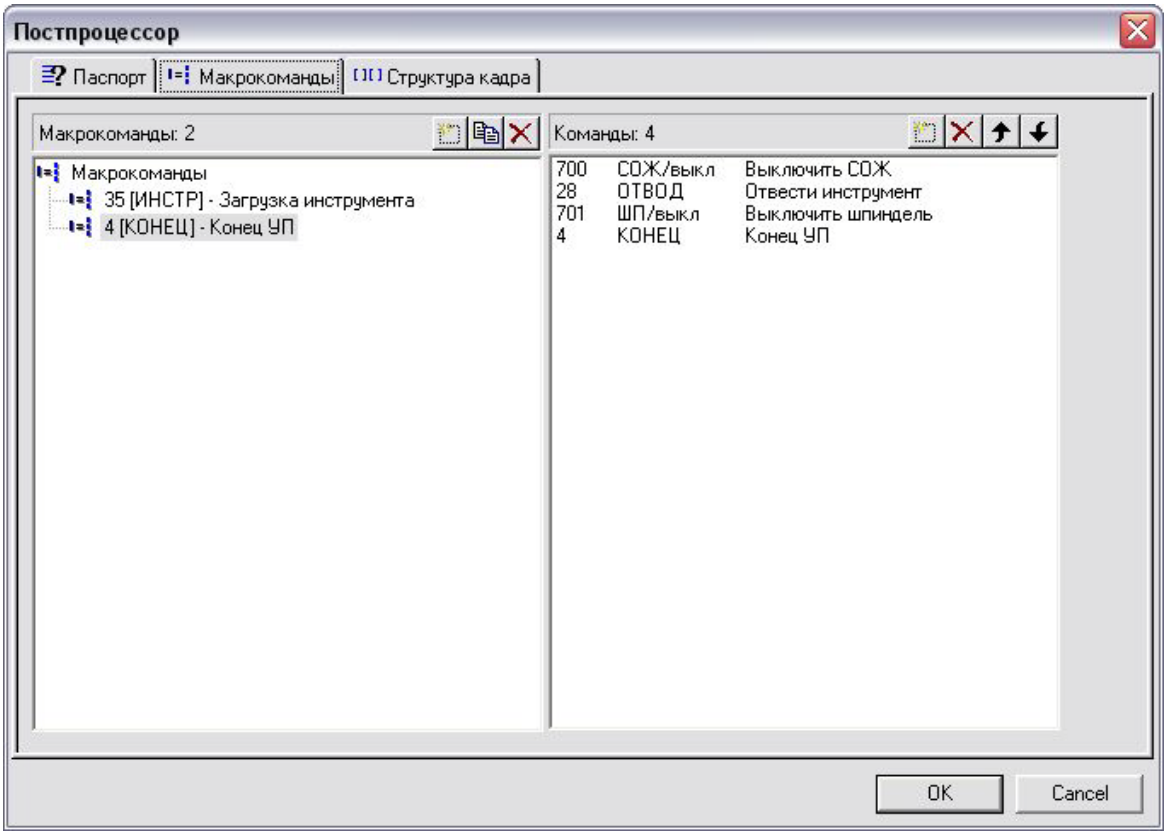
Файл макрокоманд содержит информацию об обработке адаптером тех команд **CLData**, для реализации которых необходимо выполнить несколько дополнительных команд. Количество макрокоманд и входящих в них вспомогательных команд неограниченно.

Более подробные сведения содержатся в разделах:

- Действия пользователя при формирования файла макрокоманд
- Пример формирования файла макрокоманд

Действия пользователя при формировании файла макрокоманд

Запустите модуль подготовки и отладки постпроцессоров. Для этого нажмите кнопку **"Параметры"**  . Откроется окно с активизированной вкладкой **«Паспорт»**. Перейдите на вкладку **«Макрокоманды»**.



Вкладка «Макрокоманды» диалогового окна «Постпроцессор»

Окно разделено на две части. В левой части находятся сформированные макрокоманды, а в правой создаются подчиненные элементы. Для формирования макрокоманды необходимо выбрать из списка в левой половине макрокоманду, которая будет включать в себя вспомогательные команды. Затем в правой половине создайте вспомогательные команды.

Для работы используйте кнопки правой и левой частей. Если кнопка принадлежит левой половине, то она работает с макрокомандами. Если кнопка принадлежит правой половине, то она работает с подчиненными элементами.


Создать
Добавляет новый элемент из списка команд CLData.


Копировать
Копирует выделенный элемент.


Удалить
Удаляет выделенный элемент.


Переместить вверх
Перемещает выделенный элемент вверх по списку.


Переместить вниз
Перемещает выделенный элемент вверх по списку.

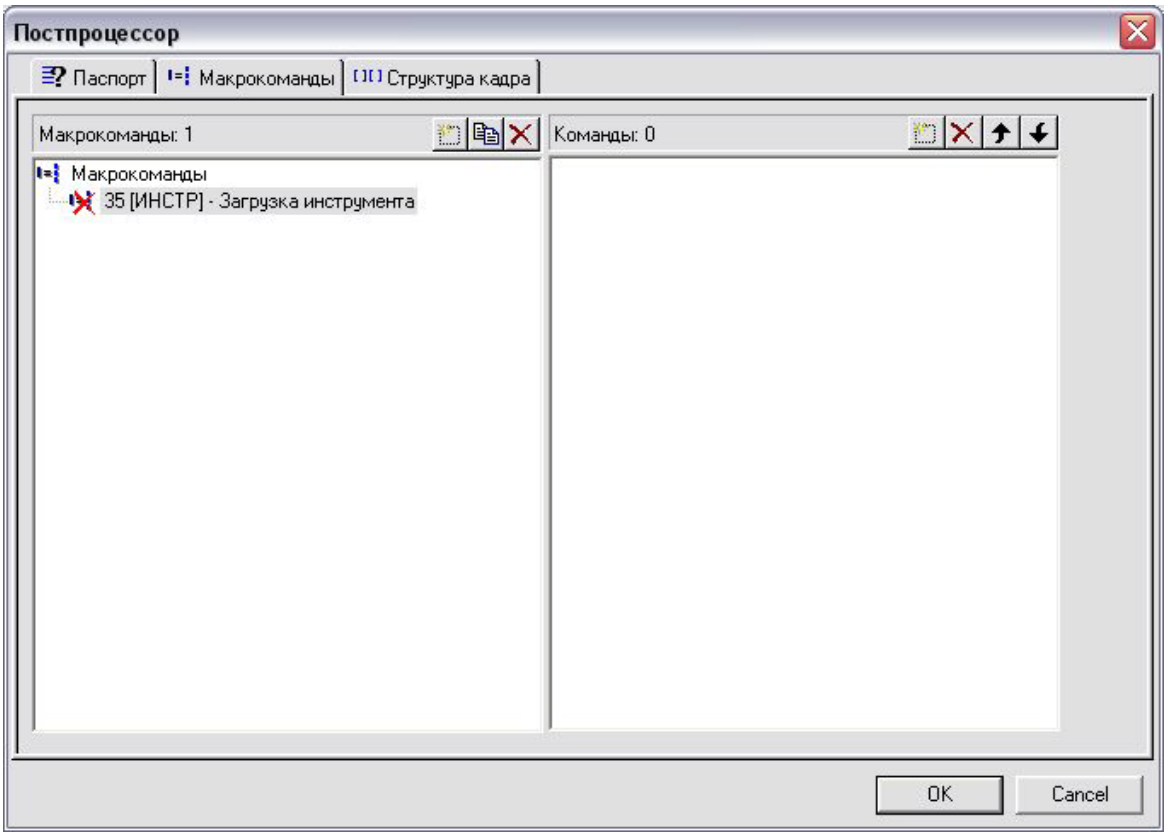
Пример формирования файлов макрокоманд

В качестве примера рассмотрим следующую ситуацию: для корректного формирования УП, необходимо перед трансляцией команды **«Загрузка инструмента»** выполнить отвод инструмента в безопасную позицию, отключить (если были включены ранее) шпиндель и СОЖ, а также отработать дополнительно алгоритм с номером 5000.

Формирование макрокоманды будет выглядеть следующим образом:

Левая половина окна

Нажмите кнопку **«Создать»**  и из появившегося списка выберите команду **«35 - Загрузка инструмента»**. В результате у Вас появится новая пустая макрокоманда.



Создание новой пустой макрокоманды

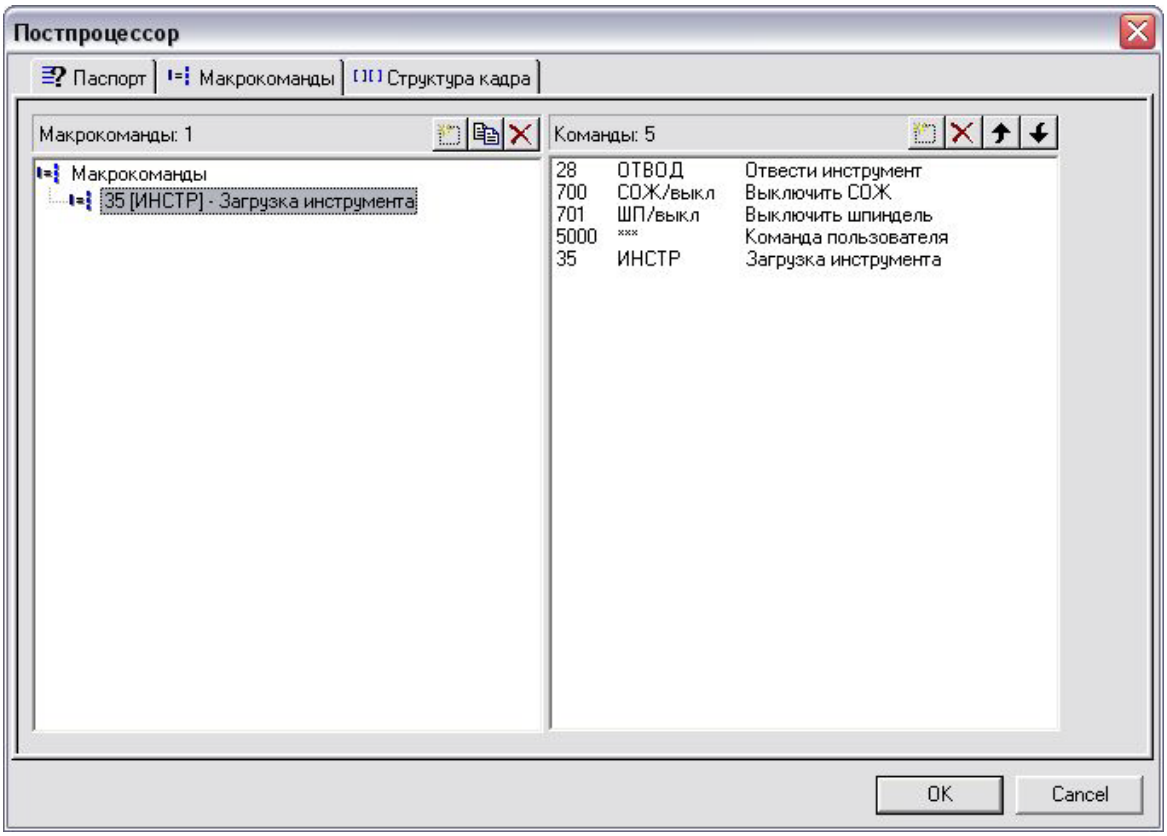
Правая половина окна

1. Нажмите кнопку **«Создать»** . В левой половине окна появится пустая строка. Нажмите на кнопку в конце строки и, из появившегося списка, выберите команду **«28 - Отвести инструмент»**.

2. Повторяя эти действия, добавьте вспомогательные команды **«700 - Выключить СОЖ»** и **«701 - Выключить шпиндель»**.

3. Теперь осталось добавить в список отработку алгоритма 5000. Для этого выделите курсором любой объект в левом окне и нажмите клавишу **Insert** на клавиатуре. Введите номер алгоритма - 5000, после чего нажмите клавишу **Enter**.

В результате у Вас появится список дополнительных команд для трансляции макрокоманды **«Загрузка инструмента»**.



Создание списка команд для макрокоманды

Примечание

Количество макрокоманд и входящих в них вспомогательных команд неограничено.

Формирование макета кадра

Макет кадра — это структура кадра управляющей программы, определяющая взаимное расположение всех возможных окон кадра и описание каждого из них.

Окно кадра описывает слово кадра управляющей программы и состоит из двух частей:

- **Символьная часть** — адрес ЧПУ (может содержать нескольк символов).
- **Формат вывода** — определяет вид выводимой числовой информации.

Пример окна кадра:

```
G[...]
```

В приведённом выше примере **G** — символьная часть окна, а **[...]** — условное обозначение формата вывода.

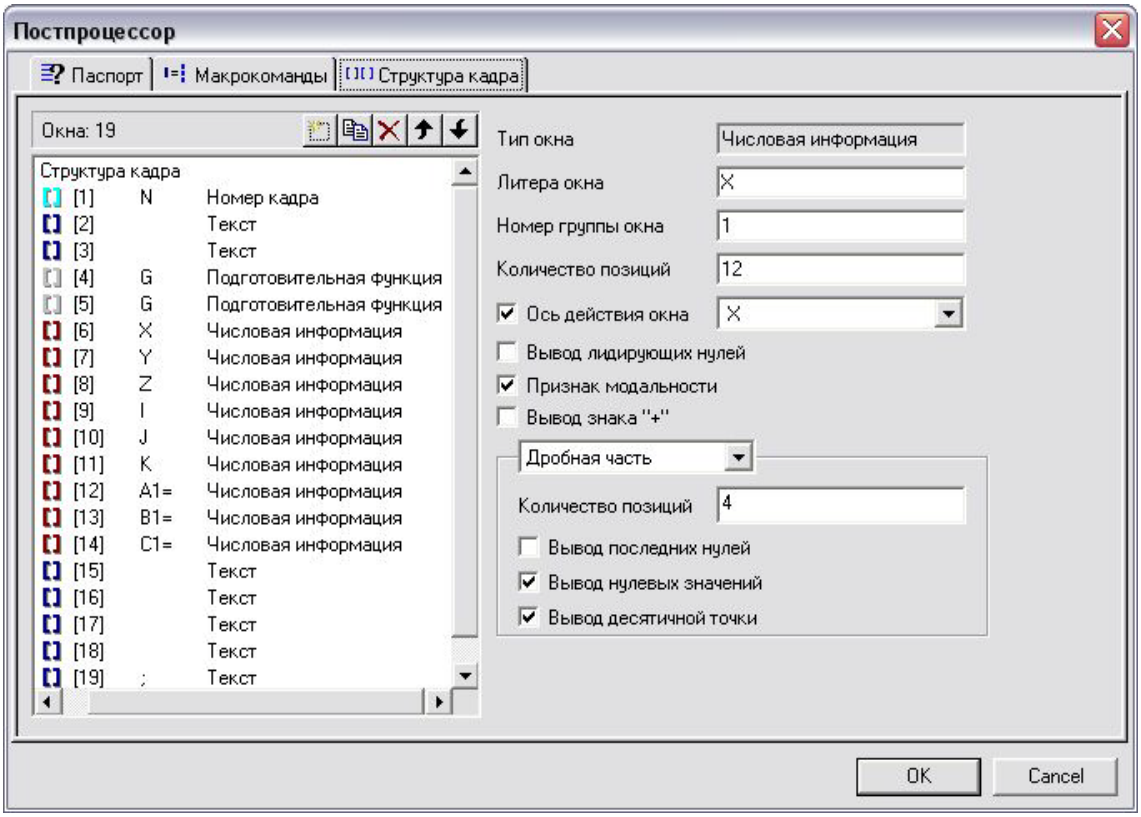
Пример макета кадра:

```
N[...] G[...] G[...] X[...] Y[...] Z[...] I[...] J[...] K[...] F[...]
M[...] L[...]
```

Файл макета кадра имеет имя, например, для постпроцессора с номером **222** - **KADR0222.ANK**. Без этого файла управляющая программа формироваться не будет, адаптер выдаст сообщение **«Нет макета кадра»**.

Действия пользователя при формировании макета кадра

Запустите модуль подготовки и отладки постпроцессоров. Для этого нажмите кнопку **«Параметры»**  . Откроется окно с активизированной вкладкой **«Паспорт»**. Перейдите на вкладку **«Структура кадра»**.



Вкладка «Макрокоманды» диалогового окна «Структура кадра»

В левой части диалогового окна представлена структура кадра. В правой части отображаются параметры выбранного окна кадра.

 **Создать**
Создаёт в кадре новое окно.

 **Копировать**
Копирует выделенный элемент.

 **Удалить**
Удаляет выделенный элемент.

 **Переместить вверх**
Перемещает выделенный элемент вверх по списку.

 **Переместить вниз**

Перемещает выделенный элемент вверх по списку.

Формат вывода

Каждое из окон кадра УП имеет свой набор параметров.

- Тип окна
- Литера окна
- Интервал нумерации
- Признак модальности
- Количество позиций
- Вывод лидирующих нулей
- Количество позиций после десятичной точки
- Вывод последних нулей
- Вывод десятичной точки в целых числах
- Вывод числа в виде целого количества дискрет
- Вывод нулевых значений
- Вывод знака «+» в положительных числах
- Ось действия окна

Тип окна

Тип окна определяет тип выводимой информации. Существуют следующие типы окон:

- **Номер кадра** используется для нумерации кадров управляющей программы.
- **Подготовительная функция** определяет режим и условия работы станка и УЧПУ, например, включение линейной и круговой интерполяции. Параметры подготовительных функций определяются через другие слова кадра.
- **Вспомогательная функция** используется для включения вспомогательных функций станка, например, включение/выключение шпинделя.
- **Числовая информация** используется для вывода параметров подготовительных функций, например, координаты линейного и кругового перемещения, время выстоя, а также для формирования любых других функций, кроме нумерации кадров и формирования конца кадра.
- **Синхродорожка** используется для формирования пропусков в управляющей программе внутри кадра и между кадрами.
- **Текст** используется для формирования любой текстовой информации.

- **Символьное окно** используется для вывода только символьной части окна. Как правило, этот тип окна используют в случаях вывода символов, которые используются в языке описания алгоритмов. Например, знак « **:** » часто обозначает главный кадр УП, но этот же знак в языке **GPP** означает операцию деления.

- **Конец кадра** используется для обозначения конца кадра.

Литера окна

Литера окна — символ или набор символов, обозначающих окно кадра. Например, для функции линейной интерполяции **G1** литерой окна будет символ **G**.

Интервал нумерации

Интервал нумерации — величина интервала нумерации кадров. Например, для интервала нумерации 10, номера кадров будут следующие:

```
N10 G0 X0 Y0 Z100
N20 X10 Y20
N30 Z10
```

Номер группы окна (число от 0 до 49) — определяет принадлежность окна к той или иной группе альтернативных функций станка. Например, альтернативными друг другу являются функции включения круговой и линейной интерполяций. То есть в одном кадре не может одновременно присутствовать **G1** и **G2** или **G3** — все эти функции выводятся в одно и то же окно. Но функции **G90**, **G17** и **G0** могут одновременно присутствовать в одном кадре, поэтому они должны выводиться в разные окна, которые имеют разные номера. Одной группе принадлежат также окна координат перемещений (**X**, **Y** и **Z**), так как литеры этих окон разные.

Признак модальности

Признак модальности — параметр, определяющий модальное действие окна. Если определено, что окно действует модально, система будет запоминать последнее выведенное в это окно значение, и последующий вывод будет осуществляться только в том случае, если выводимое значение отличается от предыдущего.

Количество позиций

Количество позиций — максимальное количество символов выводимого в окно числа. Например, если для окна **G[...]** количество позиций равно 2, то в него можно вывести целые числа от −9 до 99. Все другие числа в заданное количество позиций не поместятся, о чем адаптер выдаст соответствующее сообщение.

Вывод лидирующих нулей

Вывод лидирующих нулей — параметр, определяющий нужно ли выводить левые нули до достижения заданного количества позиций.

Например, для окна **G[..]** количество позиций равно 2 и в него нужно вывести число 1. На языке алгоритма такая команда может быть записана, как **G−>1;**. Если для окна **G[...]** определено, что лидирующие нули нужно выводить, сформируется часть кадра:

```
G01
```

Если выводить нули не нужно:

```
G1
```

Количество позиций после десятичной точки

Количество позиций после десятичной точки — определяет точность вывода числа. Например, если указано количество позиций после десятичной точки 3, число при выводе в УП будет округляться до тысячных долей. Округление производится по общим правилам: 0.5 округляется до 1.

Вывод последних нулей

Вывод последних нулей — параметр, определяющий нужно ли выводить незначащие нули до достижения заданного количества позиций после десятичной точки.

Например, для окна **X[...]** определено, что количество позиций после десятичной точки равно 3 и выполняется команда алгоритма **X–>12.5;**.

Если последние нули выводятся, то будет сформировано окно:

```
X12.500
```

Если последние нули не выводятся:

```
X12.5
```

Вывод десятичной точки в целых числах

Вывод десятичной точки в целых числах — параметр, определяющий нужно ли выводить десятичную точку в целых числах.

Например, выполняется команда алгоритма **X–>12;**. Если для окна **X[...]** определено, что десятичная точка должна быть выведена, сформируется часть кадра:

```
X12.
```

Если определено, что десятичная точка не выводится:

```
X12
```

Вывод числа в виде целого количества дискрет

Вывод числа в виде целого количества дискрет — число представляется в виде целого количества дискрет, имеющих фиксированную величину. Если число состоит не из целого количества дискрет, происходит округление выводимой величины.

Например, для окна **X[...]** определен вывод в виде целого количества дискрет и выполняется команда алгоритма **X–>12.5;**. Если дискретность равна 0.01, после отработки команды сформируется часть кадра:

```
X1250
```

Если дискрета равна 0.005, то будет сформировано:

```
X2500
```

Вывод нулевых значений

Вывод нулевых значений — параметр, определяющий нужно ли выводить нулевые значения в кадр УП.

Например, выполняется команда **X–>0;**. Если для окна **X[...]** определен вывод нулевой величины, сформируется часть кадра:

```
X0
```

Если вывод нулевой величины не определен, сформируется часть кадра, содержащая только символьную часть окна:

```
X
```

Вывод знака «+» в положительных числах

Вывод знака «+» в положительных числах —- параметр, определяющий нужно ли выводить знак **«+»** в кадр УП.

Если определен вывод знака **«+»** и выполняется команда **X–>12.5;** , сформируется часть кадра:

```
X+12.5
```

Если не определен:

```
X12.5
```

Ось действия окна

Ось действия окна — параметр, определяющий по какой оси отрабатывается выводимая в окно величина. Используется только при формировании управляющей программы в приращениях. В этом случае все погрешности, возникшие при округлении чисел, учитываются при выводе в это окно последующих значений, и все перемещения по этой оси автоматически сводятся в ноль.

Формирование окон различных типов

Ниже приведён перечень параметров для различных типов окон.

Номер кадра

- литера окна
- количество позиций
- интервал нумерации кадров
- вывод лидирующих нулей

Подготовительная и вспомогательная функции

- литера окна
- номер группы окна
- количество позиций

- вывод лидирующих нулей
- признак модальности

Числовая информация

- литера окна
- номер группы окна
- количество позиций
- ось действия окна
- вывод лидирующих нулей
- признак модальности
- вывод знака «**+**»

Остальные параметры зависят от того, в каком виде выводится число в кадр: в виде дроби или в виде целого количества дискрет.

Если в раскрывающемся списке выбрано **«Целое число дискрет»**:

- величина дискретности

Если в раскрывающемся списке выбрано **«Дробная часть»**:

- вывод последних нулей
- вывод нулевых значений
- вывод десятичной точки
- количество позиций после десятичной точки

Синхродорожка

- литера окна

Текст

- литера окна

Символьное окно

- литера окна

Конец кадра

- литера окна

Формирование файла алгоритмов

Алгоритмы трансляции команд **CLData** содержатся в файле алгоритмов. Этот файл является частью постпроцессора на станок и имеет имя, например, для анкеты с номером **222 - FTPP0222.ANK**. Без этого файла управляющая программа формироваться не будет, адаптер выдаст сообщение **«Нет файла алгоритмического заполнителя»**.

Алгоритм представляет собой последовательность строк следующего формата:

```
[<метка>:] [ELSE] [IF <условие выполнени>] <команда алгоритма>;
```

,где

IF — указывает, что команда должна быть выполнена только при соблюдении условия, идущего после IF. Только с троки с IF могут иметь альтернативные строки;

ELSE — указывает, что данная строка является альтернативной;

<метка> — идентификатор строки при ссылках (целое положительное число);

<команда алгоритма> — действие по формированию управляющей программы или изменению значения системной или пользовательской переменной.

Разделы по теме:

Действия пользователя при формировании файла алгоритмов

Для того, чтобы создать новый файл алгоритмов нажмите кнопку **«Создать»**  . Откроется окно с пустым текстовым файлом. При создании нового постпроцессора это окно создается автоматически.

Для того, чтобы открыть ранее созданный файл алгоритмов нажмите кнопку **«Открыть»**  и выберите файл алгоритмов. Система откроет выбранный файл. При открытии постпроцессора окно алгоритмов открывается автоматически.

Для того, чтобы сохранить файл алгоритмов нажмите кнопку **«Сохранить»**  . Система сохранит изменения в текущем файле алгоритмов. При сохранении постпроцессора и выполнения трансляции алгоритм файл алгоритмов автоматически сохраняется.

Файл алгоритмов формируется и редактируется в текстовом виде. Перед трансляцией файла система проверяет его на наличие ошибок.

Примечание

Строка не включается в алгоритм, если в ней допущены ошибки! Например, в конце строки нет символа **«;»**.

Пример формирования файла алгоритмов

Необходимо сформировать файл алгоритмов постпроцессора с номером 1 для токарного станка. В файл будут занесены алгоритмы трансляции следующих команд:

- Линейная интерполяция
- Круговая интерполяция
- Включить подачу
- Включить ускоренное перемещение
- Включить шпиндель
- Включить охлаждение
- Загрузка инструмента
- Выключить шпиндель
- Выключить охлаждение
- Конец управляющей программы

Можно начать с составления алгоритма на любую из команд CLData. Мы начнем с инициализации номеров окон макета кадра в алгоритме трансляции команды **«Программа»**, её код равен 1, и она всегда транслируется первой. Полный список команд с их кодами смотрите в главе «Список основных транслируемых команд CLData».

Предположим, макет кадра постпроцессора имеет следующий вид:

```
N[...] G[...] X[...] Z[...] I[...] K[...] F[...] T[...] S[...] M[...]
M[...] L[...] *[...]
```

Пользовательской переменной **_G** присвоим номер окна **G[...]** в макете кадра. Для этого введем в алгоритм строку:

```
_G=2;
```

Таким образом, инициализируем все номера окон макета кадра. Алгоритм после окончания набора примет следующий вид:

```
1;
_G=2;
_z=4;
_I=5;
_K=6;
_F=7;
_T=8;
_SK=9;
_M=10;
_M1=11;
```

```
_L=12;
END;
```

После того, как все номера окон макета инициализированы, перейдем к описанию тарнсляции линейных и интерполяций.

Линейная интерполяция (код 181)

Предположим, что управляющая программа формируется в приращениях. Приращение по оси **X** записывается в кадре с адресом **X**, приращение по оси **Z** записывается в кадре с адресом **Z**. Нулевые приращения по оси не формируются.

Сформируем алгоритм трансляции этой команды. Чтобы сформировать ненулевое перемещение по оси **X**, занесем в окно **X[...]** его величину. Для этого введем строку алгоритма:

```
IF DX!=0 _X->DX;
```

Данная запись означает, что если **(IF)** перемещение по оси X не равно 0 **(DX!=0)**, в окно _X выводится перемещение по оси X **(_X->DX)**.

Аналогично сформируем ненулевое перемещение по оси **Z**:

```
IF DZ!=0 _Z->DZ;
```

Кроме того, при выводе интерполяции необходимо указать, какой именно её вид должен выполняться. Поэтому введем в алгоритм еще одну строку:

```
_G->_FLFEED;
```

Данная запись означает, что в окно _G выводится значение пользовательской переменной _FLFEED.

Пользовательскую переменную **_FLFEED** мы будем инициализировать значениями 1 и 0 в алгоритмах трансляции команд «Включить подачу» и «Включить ускоренное перемещение». То есть, если перед перемещением была команда **«Включить подачу»**, в кадр УП выведется функция **G1**. Если же, перед перемещением была команда **«Включить ускоренное перемещение»**, в кадр УП выведется функция **G0**.

Последней командой алгоритма будет **КАДР**, выводящая всё содержимое буфера памяти в кадры УП.

```
КАДР;
```

Полностью алгоритм трансляции команды "Линейная интерполяция" будет выглядеть следующим образом:

```
181;
_G->_FLFEED;
IF DX!=0 _X->DX;
IF DZ!=0 _Z->DZ;
КАДР;
END;
```

Примечание

В конце каждого алгоритма должна стоять команда **«END;»**, которая обозначает конец алгоритма. А в конце файла алгоритма должна стоять еще одна команда **«END;»**.

Примечание

Каждая строка алгоритма должна заканчиваться символом **«;»**.

Теперь перейдем к трансляции команды **«Круговая интерполяция»**.

Круговая интерполяция (код 183)

Предположим, что круговая интерполяция формируется в приращениях. Приращение по оси **X** записывается в кадре по адресу **X**, приращение по оси **Z** записывается в кадре по адресу **Z**, расстояние по оси **X** от начальной точки дуги до центра дуги записываются по адресу **I**, расстояние по оси **Z** от начальной точки дуги до центра дуги записываются по адресу **K**. Движение по часовой стрелке задается функцией **G2**, против часовой стрелки — функцией **G3**.

Сформированный алгоритм круговой интерполяции:

```
183;
```

Если направление движения по часовой стрелке, в окно _G вывести 2.

```
IF НАПРОКР=ЧС _G->2;
```

Иначе в окно _G вывести 3.

```
ELSE _G->3;
```

Если перемещение по оси X не равно 0, в окно _X вывести величину пермещения.

```
IF DX!=0 _X->DX;
```

Если перемещение по оси Z не равно 0, в окно _Z вывести величину пермещения.

```
IF DZ!=0 _Z->DZ;
```

Если координата X центра дуги не равна координате X начальной точки, в окно _I вывести расстояние по оси X от начальной точки дуги её центра.

```
IF ХЦОКР!=ХС _I->ХЦОКР-ХС;
```

Если координата Z центра дуги не равна координате Z начальной точки, в окно _K вывести расстояние по оси Z от начальной точки дуги её центра.

```
IF ZЦОКР!=ZC _K->ZЦОКР-ZC;
```

Вывод содержимого буфера памяти в кадр УП.

```
КАДР;
```

Конец 183 алгоритма.

```
END;
```

Включить подачу (код 23)

Пусть величина рабочей подачи формируется по адресу **F**. Сформируем алгоритм обработки команды.

```
23;
```

В окно _F выведем значение подачи в мм/мин.

```
_F->S;
```

Переменной _FLFEED присвоим значение 1.

```
_FLFEED->1;
```

Конец 23 алгоритма.

```
END;
```

Обратите внимание, что в алгоритме нет команды **КАДР**. Это значит, что информация в кадр УП в этом алгоритме выведена не будет.

Включить ускоренное перемещение (код 25)

Предположим, что ускоренные перемещения определяются функцией **G0**. Так как эта функция выводится в кадр УП в 181 алгоритме, нам достаточно присвоить переменной **_FLFEED** значение 0, для последующего анализа. Сформируем алгоритм отработки команды:

```
25;
_FLFEED=0;
END;
```

Включить шпиндель (код 24)

Предположим, что величина оборотов шпинделя задается по адресу **S**. Направление вращения по часовой стрелке определяет функция **M3**, против часовой стрелки — **M4**. Сформируем алгоритм отработки этой команды:

```
24;
_S->N;
IF НВШП=ЧС _M3->;
ELSE _M->4;
END;
```

Включить охлаждение (код 26).

Предположим, что охлаждение включается функцией **M8**. Сформируем алгоритм отработки этой команды:

```
26;
_M1->8;
END;
```

Загрузка инструмента (код 35).

Предположим, что номер позиции инструмента устанавливается по адресу **T**, а команда на загрузку инструмента задается функцией **M6**. Вс` должно быть оформлено отдельным кадром. Сформируем алгоритм отработки этой команды:

```
35;
КАДР;
_T->ТИНСТР;
КАДР;
END;
```

Выключить шпиндель (код 701)

Предположим, что выключение шпинделя задаётся функцией **M5** и должно быть оформлено отдельным кадром. Сформируем алгоритм отработки этой команды:

```
701;
КАДР;
_M->5;
КАДР;
END;
```

Выключить охлаждение (код 700).

Предположим, что выключение охлаждения задается функцией **M9** и должно быть оформлено отдельным кадром. Сформируем алгоритм отработки этой команды:

```
700;
КАДР;
_M->9;
КАДР;
END;
```

Конец управляющей программы (код 4)

Предположим, что конец управляющей программы задается функцией **M2** и должно быть оформлено отдельным кадром. Сформируем алгоритм отработки этой команды:

```
4;
КАДР;
_M->2;
КАДР;
END;
```

После завершения процесса формирования, файл алгоритмов должен выглядеть следующим образом:

```
1;
_G=2;
_z=4;
_I=5;
_K=6;
_F=7;
_T=8;
_SK=9;
_M=10;
_M1=11;
```

```
_L=12;
END;

4;
КАДР;
_M->2;
КАДР;
END;

23;
_F->S;
_FLEED=1;
END;

24;
_S->N;
IF НВШП=ЧС _M3->;
ELSE _M->4;
END;

25;
_FLFEED=0;
END;

26;
_M1->8;
END;

35;
КАДР;
_T->ТИНСТР;
КАДР;
END;

181;
_G->_FLFEED;
IF DX!=0 _X->DX;
IF DZ!=0 _Z->DZ;
КАДР;
END;

183;
IF НАПРОКР=ЧС _G->2;
ELSE _G->3;
IF DX!=0 _X->DX;
IF DZ!=0 _Z->DZ;
IF ХЦОКР!=ХС _I->ХЦОКР-ХС;
IF ZЦОКР!=ХС _K->ZЦОКР-ZС;
КАДР;
END;

700;
КАДР;
_M->9;
КАДР;
END;
```

```
701;
КАДР;
_M->5;
КАДР;
END;
```

Примечание

Для облегчения поиска и редактирования, советуем заранее располагать алгоритмы в порядке возрастания их номеров.

Основные команды и функции

При описании алгоритмов трансляции команд CLData, можно использовать различные арифметические и тригонометрические функции. Наряду с самыми простыми действиями в алгоритмах можно организовывать циклы, вызывать другие алгоритмы, а также использовать команды трансформации для пересчета координат.

Разделы по теме:

- Арифметичесие действия и функции в алгоритмах
- Команды алгоритмов

Кроме того, советуем обратить внимание на использовании некоторых сложных команд и функций, описание которых можно найти в разделе «Примеры».

Арифметические действия и функции в алгоритмах

Действия и функции, используемые при проектировании алгоритмов, можно разделить на несколько групп.

Группы

- Арфметические действия
- Арифметические и тригонометрические функции
- Дополнительные функции
- Функции для работы с трансформами

Арифметические действия

При составлении выражений допускается использование следующих действий:

Арифметические операции

Оператор	Операция
+	сложение
-	вычитание
*	умножение
:	деление
**	возведение в степень
&	логическое умножение (И)
|	логическое сложение (ИЛИ)
<	меньше
>	больше
<=	меньше или равно
>=	больше или равно
=	равенство
!=	неравенство

Арифметические и тригонометрические функции

Для выполнения различных расчетов можно использовать следующие функции:

Арифметические и тригонометрические функции

Функция	Операция
SQRT()	квадратный корень
EXP()	экспонента
ABS()	абсолютная величина
LOGD()	десятичный логарифм
LOGE()	натуральный логарифм
DEV()	вычисление целойт части с округлением
SIN()	синус угла
COS()	косинус угла
TAN()	тангенс угла
CTAN()	котангенс угла
ASIN()	арксинус угла
ACOS()	арккосинус угла
ATAN()	арктангенс угла
ACTN()	арккотангенс угла

Примечание

В тригонометрических функциях угол должен быть задан в радианах!

Дополнительные функции

Наряду с арифметическими и тригонометрическими функциями, при описании алгоритмов можно использовать дополнительные функции, которые в некоторых случаях значительно облегчают процесс проектирования постпроцессора.

COKH()

Величина, выведенная в окно. Аргументом функции является номер окна в макете кадра.

Пример:

```
IF СОКН(_X)!=XT _X->XT;
```

Если содержимое окна, номером которого инициализирована переменная _X, не равно текущей координате по оси X, в окно _X выводится выводится координата X.

NCD()

Поиск фразы с заданным кодом от текущего положения до команд «Загрузка инструмента» или «Конец программы». Аргументом функции является код команды CLData. Выход функции: 1— фраза CLData с заданным кодом найдена, 0 — фраза CLData с заданным кодом не найдена.

Пример:

```
_ID=NCD(28);
IF _ID!=1 do;
   I=BMES('Не задан отвод');
   EXIT;
```

```
ENDDO;
```

BMES()

Сообщение, выдаваемое на экран при работе адаптера. Аргументом является текст сообщения. Имеет две кнопки "Ок" и "Cancel". Выход функции: 1 – Ok, 2 – Cancel.

Пример:

```
I=BMES('Продолжить?');
IF I=2 DO;
   I=BMES('Формирование УП было прервано');
   EXIT;
ENDDO;
```

WINCODE()

Перекодировка текстовой строки из DOC в Windows. Аргументом является текстовая строка. Обычно используется для вывода в текст УП комментария, написанного кириллицей.

Пример:

```
_P1='любой текст в DOC-кодировке';
_P1=WINCODE(_P1);
_TXT->_P1;
```

GETCODE()

Установить код команды CLData. Аргументом является порядковый номер команды CLData. Возвращает код команды. Используется только в случае формирования новой CLData на основе существующей.

GETLONG()

Установить длину команды CLData. Аргументом является порядковый номер команды CLData. Возвращает длину внутренней структуры команды в двойных словах (одно двойное слово — 8 байт). Используется только в случае формирования новой CLData на основе существующей.

GETQC()

Установить количество фраз CLData в основной программе или подпрограмме. Аргументом является код: 0 — основная программа, 1 — подпрограмма.

Пример:

```
_QC=GETQC(0);
```

В приведённом выше примере в пользовательскую переменную записывается _QC запишется количество фраз CLData в основной программе.

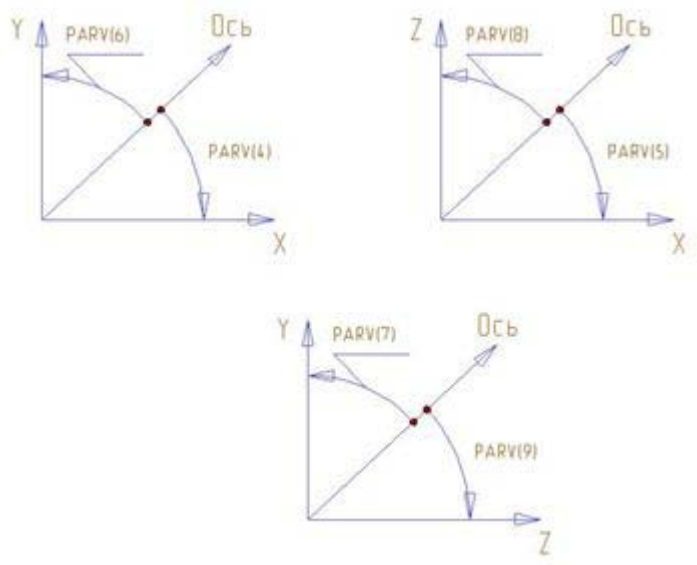
Функции для работы с трансформами

Если плоскость обработки не параллельна плоскости **XY** системы координат детали, в CLData появляется команда «Трансформ» (код 10123), определяющая положение плоскости обработки относительно системы координат детали. Для описания этой команды используется функция **PARV()**, которая возвращает либо значение угла в радианах, либо величину в миллиметрах.

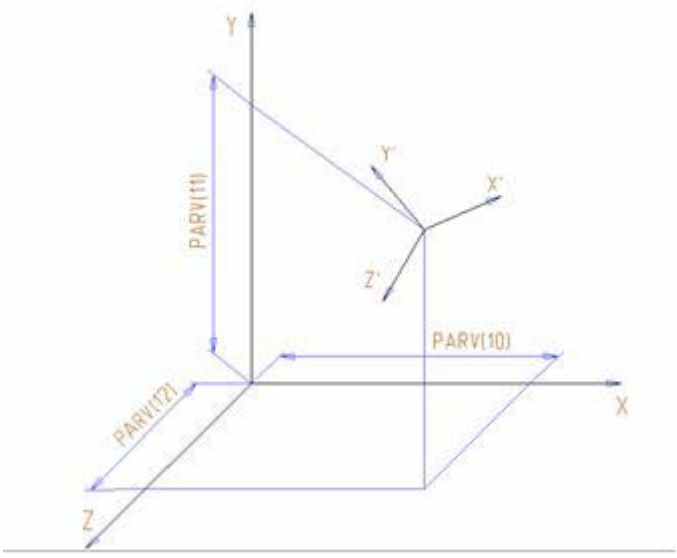
Аргументом функции **PARV()** является код.

Функции для работы с трансформами

Функция	Результат
PARV(1)	направляющий косинус выбранной оси текущей системы координат к оси X базовой системы координат
PARV(2)	направляющий косинус выбранной оси текущей системы координат к оси Y базовой системы координат
PARV(3)	направляющий косинус выбранной оси текущей системы координат к оси Z базовой системы координат
PARV(4)	Угол, на который нужно повернуть текущую систему координат для того, чтобы проекция выбранной оси текущей системы координат на плоскость **XY** базовой системы координат совместилась с осью **X** базовой системы координат.
PARV(5)	Угол, на который нужно повернуть текущую систему координат для того, чтобы проекция выбранной оси текущей системы координат на плоскость **XZ** базовой системы координат совместилась с осью **X** базовой системы координат.
PARV(6)	Угол, на который нужно повернуть текущую систему координат для того, чтобы проекция выбранной оси текущей системы координат на плоскость **XY** базовой системы координат совместилась с осью **Y** базовой системы координат.
PARV(7)	Угол, на который нужно повернуть текущую систему координат для того, чтобы проекция выбранной оси текущей системы координат на плоскость **YZ** базовой системы координат совместилась с осью **Y** базовой системы координат.
PARV(8)	Угол, на который нужно повернуть текущую систему координат для того, чтобы проекция выбранной оси текущей системы координат на плоскость **XZ** базовой системы координат совместилась с осью **Z** базовой системы координат.
PARV(9)	Угол, на который нужно повернуть текущую систему координат для того, чтобы проекция выбранной оси текущей системы координат на плоскость **YZ** базовой системы координат совместилась с осью **Z** базовой системы координат.
PARV(10)	Смещение нуля текущей системы координат по оси **X** базовой системы координат
PARV(11)	Смещение нуля текущей системы координат по оси **Y** базовой системы координат
PARV(12)	Смещение нуля текущей системы координат по оси **Z** базовой системы координат



Углы поворота текущей системы координат относительно базовой СК



Велечины смещений нуля текущей системы координат (X'Y'Z') относительно нуля базовой СК (XYZ)

Примечание

Функцию **PARV()** можно использовать только после команды выбора оси **DATAV**.

Команды алгоритмов

Команды, используемые при проектировании алгоритмов, можно разделить на несколько групп:

Основные команды

К основным командам относятся наиболее часто используемые команды.

Начало алгоритма

Команда определяет начало алгоритма и имеет числовой формат:

```
<номер алгоритма>;
```

Примечание

Как правило номер алгоритма совпадает с номером транслируемой команды CLData, но можно использовать и другие номера для написания собственных алгоритмов. При этом лучше начинать нумеровать свои алгоритмы начиная с 1000.

Конец алгоритма

Команда определяет конец алгоритма или файла алгоритмов и имеет формат:

```
КОНЕЦ;
```

или

```
END;
```

Комментарий

Команда определяет строку комментария в тексте алгоритма. Все, что идет после нее до символа **«;»**, будет игнорироваться при трансляции алгоритма. Команда имеет формат:

```
//<строка>;
```

Пример:

```
//НАЧИНАЕТСЯ ВЫВОД ДАННЫХ В УП;
_G->1;
_X->100;
КАДР;
```

Присвоение значения

Команда изменяет значение системной или пользовательской переменной и имеет формат:

```
 =
```

Пример:

```
_P1=10;
_P2=20;
_P3=_P1+_P2+15;
```

После отработки этих команд пользовательские переменные _P1, _P2 и _P3 примут значения 10, 20 и 45 соответственно.

Занесение информации в окно макета кадра

Команда имеет следующий формат:

```
< номер окна в макете кадра > -> < информация >;
```

Пример.

Инструмент находится в точке с координатами X=10, Y=20, Z=30. Макет кадра:

```
N[...] G[...] G[...] X[...] Y[...] Z[...] M[...]
```

Отрабатываются команды алгоритма:

```
4->XT;
5->YT;
6->ZT;
2->1;
```

В буфере памяти сформируется:

```
G1 X10. Y20. Z30.
```

Примечание

Для облегчения ориентирования в алгоритмах, вместо номера окна можно использовать переменную, инициализированную этим номером.

Вывод в УП символьной части окна

Команда используется для вывода в кадр УП литеры окна и имеет следующий формат:

```
АОКНО < номер окна >;
```

или

```
WIND/ON < номер окна >;
```

Пример.

Макет кадра:

```
N[...] /REM_TOOL[...] G[...] G[...] X[...] Y[...] Z[...] M[...]
```

Отрабатываются команды алгоритма:

```
АОКНО2;
```

В буфере памяти сформируется:

```
/REM_TOOL
```

Формирование конца кадра

Команда используется для вывода в кадр УП информации из буфера памяти. Автоматически формируется номер кадра и конец кадра. Если в буфере не было никакой информации, эта команда игнорируется. Команда КАДР имеет следующий формат:

```
КАДР;
```

или

```
BLOCK;
```

Пример:

```
_G->1;
КАДР;
_G->2;
КАДР;
КАДР;
```

Сформируются кадры:

```
N001 G01
N002 G02
```

Обратите внимание на то, что последняя команда КАДР была проигнорирована.

Подавление номера кадра

Команда отменяет автоматическое формирование номера в текущем кадре, при этом порядок нумерации сохраняется. Команда имеет следующий формат:

```
ПНКАДР;
```

или

```
NBL/OFF;
```

Пример:

```
_G->10;
КАДР;
_G->20;
ПНКАДР;
КАДР;
```

```
_G->30;
КАДР;
```

Сформируются кадры:

```
N001 G10
G20
N003 G30
```

Инициализация номера кадра

Команда устанавливает значение, с которого пойдет дальнейшая нумерация кадров. Команда имеет следующий формат:

```
ИНКАДР <число>;
```

или

```
NBL/SET <число>;
```

Пример:

```
_G->10;
КАДР;
_G->20;
ИНКАДР 100;
КАДР;
_G->30;
КАДР;
```

Сформируются кадры:

```
N001 G10
N101 G20
N102 G30
```

Подавление модальности окна

Команда устанавливает признак игнорирования модальности окна, действует только в текущем кадре. Команда имеет следующий формат:

```
МОДВЫКЛ <номер окна>;
```

или

```
MODOFF <номер окна>;
```

или

```
MOD/OFF <номер окна>;
```

Пример.

Предположим, что окно _G[...] модальное.

```
MOD/OFF <номер окна>;
_X->100;
```

```
КАДР;
_G->1;
_Y->200;
КАДР;
__G->1;
_X->300;
MODOFF _G;
КАДР;
_G->1;
_Y->400;
КАДР;
```

Сформируются кадры:

```
N001 G01 X100
N002 Y200
N003 G01 X300
N004 Y400
```

Вызов алгоритма

Команда используется для вызова и последующего выполнения какого-либо алгоритма в качестве подпрограммы. Номером алгоритма считается код команды CLDATA или число, которое вы определили вместо кода команды CLDATA при составлении алгоритма. Номера для собственных алгоритмов лучше выбирать больше 1000. Команда имеет следующий формат:

```
CALL < номер алгоритма >;
```

Пример.

Предположим, обрабатывается следующий алгоритм.

```
_G->1;
_X->100;
CALL 6000;
КАДР;
```

Алгоритм с номером 6000 выглядит следующим образом:

```
6000;
_L->801;
END;
```

Сформируется кадр:

```
N001 G01 X100 L801
```

Прерывание формирования УП

Команда прекращает формирование управляющей программы и имеет формат:

```
EXIT;
```

Дополнительные команды.

К дополнительным командам относятся:

Включение кругового интерполятора

Команда используется для включения кругового интерполятора, если он был отключен ранее. После включения все круговые интерполяции в CLData будут интерпретироваться как круговые интерполяции. Команда имеет следующий формат:

```
CIRC/ON;
```

Выключение кругового интерполятора

Команда используется для выключения кругового интерполятора. После выключения все круговые интерполяции в CLData будут аппроксимированы линейными перемещениями с точностью, определенной в паспорте станка. Команда имеет следующий формат:

```
CIRC/OFF;
```

Выключить интервал между словами кадра

После выполнения этой команды все слова в кадре не будут отделены пробелами. Команда имеет следующий формат:

```
INT/OFF;
```

Включить интервал между словами кадра

После выполнения этой команды все слова в кадре будут отделены пробелами. Команда имеет следующий формат:

```
INT/ON;
```

Включить нумерацию кадров

Команда используется для включения нумерации кадров, если она была отключена ранее. После выполнения этой команды все кадры будут нумероваться. Команда имеет следующий формат:

```
GNBL/ON;
```

Выключить нумерацию кадров

Команда используется для выключения нумерации кадров. После выполнения этой команды кадры не будут нумероваться. Команда имеет следующий формат:

```
GNBL/OFF;
```

Вызывать текущий алгоритм перед трансляцией каждой команды CLData

Команда используется для выполнения определенных действий перед отработкой каждой команды CLData, независимо от того, есть ли алгоритм описания этой команды. Команда имеет следующий формат:

```
EALG;
```

Примечание

Использование данной команды возможно только в одном алгоритме постпроцессора.

Пример:

```
5000;
EALG;
BLOCK;
_TXT->'НОВАЯ КОМАНДА CLDATA';
BLOCK;
END;
```

Перед тем, как обработать алгоритм, адаптер выполнит алгоритм под номером 5000. То есть, если в CLData сформирован набор линейных перемещений, кадры УП будут выглядеть следующим образом:

```
N90 НОВАЯ КОМАНДА CLADATA
N100 G1 X10 Y20
N110 НОВАЯ КОМАНДА CLDATA
N120 X40
N130 НОВАЯ КОМАНДА CLADATA
N140 Y50
```

Вызывать текущий алгоритм перед трансляцией каждой команды CLData в случае, если для кода команды нет соответствующего алгоритма

Команда используется для выполнения определенных действий перед отработкой каждой команды CLData, трансляция которой не определена в алгоритме с соответствующим номером. Команда имеет следующий формат:

```
DEFAULT;
```

Циклы, условия и движение по CLData

Организация различных циклов и условий осуществляется с помощью следующих команд:

Переход на метку

Команда используется для передачи управления на метку. Команда имеет следующий формат:

```
GOTO < номер метки >;
```

Сама метка имеет следующий формат:

```
< номер метки >: < выражение >
```

Пример:

```
1: _P2=1;
_P2=0;
IF _P2=0 GOTO 1;
```

Организация условия

Команда используется организации условий и имеет следующий формат:

```
IF < условие > < выражение >; или ЕСЛИ < условие > < выражение >;
```

Кроме условия можно организовывать ложные условия. Для этого используется команда:

```
ELSE < выражение >; или ИНАЧЕ < выражение >;
```

Пример:

```
_P2=5;
_P3=10;
IF _P1=1 _P2=_P3;
ELSE _P2=0;
```

Если переменная _P1=1, то в результате выполнния условия переменная _P2 примет значение 10. Иначе переменная _P2 примет значение 0.

Операторные скобки

Команда используется для выделения строк алгоритма, которые должны быть выполнены при общем условии. Каждая открывающая операторная скобка должна быть закрыта. Эта команда имеет следующий формат:

```
...ОПСК;
...
...КОПСК;
```

или

```
...DO;
...
...ENDDO;
```

Например, предположим, что отрабатывается следующий алгоритм:

```
IF _P1=1 DO;
   _X->100;
   _Y->100;
ENDDO;
ELSE IF _P1=2 DO;
   _X->200;
   _Y->200;
ENDDO;
ELSE DO;
   _X->300;
   _Y->300;
ENDDO;
BLOCK;
```

Если _P1=1, то сформируется кадр:

```
N001 X100 Y100
```

Если _P1=2, то сформируется кадр:

```
N001 X200 Y200
```

Во всех остальных случая сформируется кадр:

```
N001 X300 Y300
```

Организация циклов

Команда используется для организации различных циклов и имеет следующий формат:

```
FOR <инициализация переменной> <условие выполенния> <выражение итерации>
   <тело цикла>
ENDDO;
Пример:
_K=0;
FOR _I=1 _I=3 _I=I+1;
   _K=K+2;
ENDDO;
```

После отработки цикла переменная _K примет значение 6.

Начать преобразование команд CLData в кадры УП заново

Команда используется в случае, когда необходимо выполнить чтение данных из файла CLData перед тем, как начать формировать УП. Например, может возникнуть необходимость в начале УП перечислить все точки, на которых будут отрабатываться стандартные сверлильно-расточные циклы. По этой команде отработка текущей команды CLData останавливается, и преобразование начинается заново, с первого алгоритма. Команда имеет следующий формат:

```
BEGIN;
```

Примечание

Сформированные к моменту отработки команды BEGIN, кадры УП не уничтожаются.

Примечание

Адаптер не ограничивает количество таких преобразований, поэтому избегайте зацикливания.

Передположим, отрабатывается следующий алгоритм:

```
IF _P1=0 DO;
   _P1=1;
   BEGIN;
ENDDO;
```

Как видно из примера, переменная _P1 меняет своё значение для того, чтобы при следующей трансляции CLData не произошло зацикливания.

Команды для работы с файлами

При проектировании постпроцессора можно назначать имя файла, в который будут выводиться кадры УП и копировать сформированные файлы. Для этого используются следующие команды:

Установить имя файла

Команда используется для определения имени файла, в который будут выводиться кадры управляющей программы. По умолчанию устанавливается файл с именем PLENT.TAP. Если формируется одновременно несколько файлов, переключение между ними осуществляется этой командой. При этом, если в указанный файл были сделаны записи, все последующие будут добавлены. Имя файла может содержать полный путь к этому файлу. Если путь не указан, файл формируется в рабочем директории. Команда имеет следующий формат:

```
SET/FILE 'имя файла с расширением';
```

Примечание

Адаптер не может создавать каталоги, поэтому перед тем, как указать путь к файлу, убедитесь, что конечный каталог существует.

Пример:

```
SET/FILE 'c:/tmp/abc.iso';
```

По этой команде адаптер создаст файл с именем **ABC.ISO** в каталоге **C:/tmp**.

Скопировать файл

Команда используется для копирования ранее созданного файла. Имя файла может содержать полный путь к этому файлу. Если путь не указан, адаптер будет искать файл в рабочем директории. Команда имеет следующий формат:

```
COPYFILE 'имя файла с расширением';
```

Примечание

Перед копированием файла, содержащего пронумерованные кадры УП, отключите нумерацию кадров командой **GNBL/OFF**. Иначе адаптер повторно пронумерует все строки УП.

Пример:

```
SET/FILE 'c:/tmp/abc.iso';
GNBL/OFF;
COPYFILE 'PLENT.TAP';
```

Адаптер сначала создаст файл с именем **ABC.ISO** в каталоге **C:/tmp**, а затем скопирует в этот файл содержимое файла **PLENT.TAP**.

Команды для работы с трансформами

В файле CLData координаты всех перемещений инструмента откладываются в системе координат конструктивного элемента (СК КЭ). Если плоскость XY системы координат КЭ не параллельна плоскости XY системы координат детали, в CLData появляется команда **«Трансформ»** (код 10123), определяющая положение СК КЭ относительно системы координат детали.

Трансформ — это матрица, с помощью которой адаптер переводит координаты движения инструмента из одной СК в другую.

Для преобразования координат, используются следующие команды:

Вычислить параметры оси

Команда вычисляет положение выбранной оси одной СК относительно другой. После ее выполнения, параметры, определяющие положение оси, доступны через функцию PARV(). Команда имеет следующий формат:

```
DATAV {A, B или C} TR1 TR2;
```

где

TR1 — матрица, содержащая данные о СК выбранной оси;

TR2 — матрица, содержащая данные о СК, относительно которой вычисляется положение оси;

{A, B или C} — ось, положение которой необходимо определить.

Вычисли положение оси C системы координат КЭ относительно системы координат детали.

```
DATAV C CLDCOOR SYSCOOR;
```

Повернуть систему координат

Команда поворачивает одну систему координат относительно оси другой СК. После ее выполнения, параметры, определяющие положение осей поворачиваемой СК необходимо определять заново. Команда имеет следующий формат:

```
ROTECOOR {A, B или C} TR1 V;
```

где

TR1 — матрица, содержащая данные о системе координат, которую нужно повернуть;

V — угол поворота в радианах;

{A, B или C} — ось системы координат, относительно которой происходит поворот.

Повернём систему координат КЭ вокруг оси C системы координат детали на угол _CRAD.

Примечание

Система координат, вокруг которой происходит поворот, по умолчанию совпадает с СК детали. Но её положение можно изменять с помощью соответствующих команд.

```
ROTECOOR C COORUS1 _CRAD;
```

Вычислить положение одной системы координат относительно другой

Команда позволяет вычислить текущее положение одной системы координат относительно другой. Команда имеет следующий формат:

```
CLCOOR TR1 TR2;
```

где

TR1 — матрица, содержащая данные о системе координат, относительно которой производится вычисление;

TR2 — матрица, содержащая данные о системе координат, положение которой вычисляется.

Примечание

Результаты вычислений записываются в матрицу TR2.

Вычислим положение системы координат COORUS2 относительно COORUS2

```
CLCOOR COORUS1 COORUS2;
CALCOOR=COORUS2;
```

Установить систему координат для трансформа поворота

Команда позволяет установить координаты X, Y, Z начала системы координат, относительно которой будут поворачиваться все остальные СК. Команда имеет следующий формат:

```
SET/RC X Y Z;
```

или

```
SET/RC _P[];
```

где

X Y Z — координаты начала СК;

_P[] — массив, содержащий координаты начала СК.

Сместим **ось вращения A** (поворот вокруг оси X) на **50 мм** вниз относительно СК ноля детали.

```
SET/RC 0 0 -50;
```

Команды для пересчета 5-ти координатных перемещений

При проектировании постпроцессора, может возникнуть необходимость пересчета 5-ти координатных перемещений.

К командам для пересчета относятся:

Вычислить параметры оси инструмента

Команда вычисляет положение оси инструмента относительно СК детали. После её выполнения, параметры, определяющие положение оси, доступны через функцию PARV() аналогично трансформам. Описание функции PARV() можно посмотреть в разделе «Арифметические действия и функции в алгоритмах». Команда имеет следующий формат:

```
DATAORT VSP/X VSP/Y VSP/Z;
```

где

VSP/X, VSP/Y и VSP/Z — направляющие косинусы, определяющие положение оси инструмента.

Предположим, что инструмент качается по оси B (вокруг оси Y). Необходимо вычислить угол наклона инструмента относительно оси Z СК детали.

```
DATAORT VSP/X VSP/Y VSP/Z;
_BRAD=PARV(8);
```

Переменная _BRAD примет значение угла наклона инструмента в радианах.

Повернуть точку вокруг оси

Команда поворачивает точку с указанными координатами вокруг выбранной оси СК детали на определенный угол. При этом, в переменные или массив, содержащие координаты точки, после поворота запишутся пересчитанные значения. Команда имеет следующий формат:

```
RPOINT <координата X> <координата Y> <координата Z> <ось> <угол>;
```

или

```
RPOINT <имя массива>[] <ось> <угол>;
```

где

<координата X, Y, Z> — переменные, содержпщие координаты точки;

<имя массива>[] — массив, содержащий координаты точки;

<ось> — ось СК детали, вокруг которой поворачивается точка;

<угол> — значение угла поворота в радианах.

Примечание

В параметрах команды не допускается использование констант и выражений.

Нужно повернуть точку с координатами [X=10; Y=20; Z=0] вокруг оси A системы координат детали на угол 90 градусов.

```
_XT=10;
_YT=20;
_ZT=0;
_ARAD=ASIN(1);
RPOINT _XT _YT _ZT A _ARAD;
```

После выполнения команды переменные примут значения _XT=10; _YT=0; _ZT=20.

Команды для работы с массивами данных

При проектировании постпроцессора, может возникнуть необходимость в создании массива данных. Например, для того, чтобы в начале УП вывести данные обо всех инструментах, используемых в маршруте обработки, можно сначала оттранслировать всю CLData, без вывода кадров УП, собрать в один массив номера инструментов, в другой - тип инструментов и т.д. А потом, при повторной трансляции CLData (организованной с помощью команды BEGIN), вывести считанные данные в заголовок УП.

К командам для работы с массивами данных относятся:

Инициализация массива элементов

Команда инициализирует массив элементов указанным значением. Команда имеет следующий формат:

```
< имя массива >[]=< значение >;
```

Инициализируем массив данных с именем _P1, все элементы которого равны 10.

```
_P1[]=10;
```

Примечание

Массив имеет переменную длину, то есть автоматически изменяет свой размер при добавлении нового элемента.

Присваивание значения одному элементу массива

Команда присваивает элементу массива с указанным номером заданного значения. Команда имеет следующий формат:

```
< имя массива >[номер элемента массива]=< значение >;
```

Второму элементу массива _P присвоим значение 5.

```
_P1[2]=5;
```

Присваивание переменной значения элемента массива

Команда присваивает переменной значение элемента массива с указанным номером. Команда имеет следующий формат:

```
< имя переменной >=< имя массива >[номер элемента массива];
```

Переменной _NAME присвоим значение второго элемента массива _P1.

```
_NAME=_P1[2];
```

Команды для прямого доступа и преобразования CLData

Команды для прямого доступа и преобразования CLData необходимы для организации доступа к данным, без использования системных переменных, и внесения изменений в существующую CLData. К ним относятся:

Установить код фразы CLData

Команда устанавливает код транслируемой команды CLData. Команда имеет следующий формат:

```
GETCODE(< номер фразы CLData >);
```

Установить длину фразы CLData

Команда устанавливает длину транслируемой команды CLData. Команда имеет следующий формат:

```
GETLONG(< номер фразы CLData >);
```

Сведения об используемых подпрограммах

Команда позволяет получить сведения об используемых подпрограммах. Команда имеет следующий формат:

```
SEPSUB < имя переменной > < имя массива 1>[] < имя массива2 >[] < имя массива3 >[];
```

Данные о подпрограммах заносятся в:

<имя переменной> — количество используемых подпрограмм;

<имя массива 1>[] — массив, содержащий имена используемых подпрограмм;

<имя массива 2>[] — массив, содержащий порядковый номер первой команды каждой из используемых подпрограмм;

<имя массива 3>[] — массив, содержащий порядковый номер последней команды каждой из используемых подпрограмм.

Прочитать команду CLData в массив

Команда считывает текущую команду CLData в массив с установленным именем. Команда имеет следующий формат:

```
GETFR < номер фразы CLData > < имя массива >[];
```

Установка на обработку основной программы

Команда устанавливает на обработку файл **FCLD.wrk**, содержащий основную программу. Команда имеет следующий формат:

```
SETPROG;
```

Установка на обработку основной программы

Команда устанавливает на обработку файл **CLDP.wrk**, содержащий подпрограмму. Команда имеет следующий формат:

```
SETSUBP;
```

Установить файл для записи CLData

Команда устанавливает файл для записи CLData с указанным именем. Команда имеет следующий формат:

```
SETWRF <'имя файла'>;
```

Установить файл записи как CLData основной программы

Команда устанавливает файл с указанным именем как файл **CLData** основной программы. Команда имеет следующий формат:

```
SETPF <'имя файла'>;
```

Установить файл записи как CLData подпрограммы

Команда устанавливает файл с указанным именем как файл **CLData** подпрограммы. Команда имеет следующий формат:

```
SETSF <'имя файла'>;
```

Сформировать команду CLData

Команда формирует команду **CLData**, используя массив данных с указанным именем. Команда имеет следующий формат:

```
WRCOM <'имя массива'>[];
```

Отработать как команду CLData

Команда отрабатывает как команду CLData с указанным кодом, любую структуру, содержащуюся в массиве с указанным именем. Команда имеет следующий формат:

```
RUNCOM <'имя массива'>[] [код команды];
```

Блокировка файла макрокоманд

Команда блокирует использование файла макрокоманд при трансляции **CLData**. Команда имеет следующий формат:

```
BLMACRO;
```

Разблокировка файла макрокоманд

Команда разблокирует использование файла макрокоманд при трансляции **CLData**, если блокировка была установлена ранее. Команда имеет следующий формат:

```
SETMACRO;
```

Чтение файла макрокоманд

Команда предназначена для чтения файла макрокоманд, при этом в массив с указанным именем заносятся коды используемых подкоманд. Команда имеет следующий формат:

```
LMACRO < код фразы > < имя переменной > < имя массива >;
```

где

< код фразы > — код считываемой макрокоманды;

< имя переменной > — имя переменной, в которую записывается количество подкоманд;

< имя массива > — имя массива, содержащего список подкоманд.

Получить команду CLData

Команда предназначена для установки на трансляцию команды из нового, сформированного файла **CLData** основной программы или подпрограммы. Команда имеет следующий формат:

```
GETWRFR;
```

Системные переменные

Системные переменные имеют фиксированные имена и автоматически принимают значения параметров команд **CLData**. В ранних версиях адаптера использовались фиксированные имена также и для пользовательских переменных. Сейчас достаточно поставить перед именем переменной знак подчеркивания «_» и она будет интерпретироваться адаптером как пользовательская. Кроме того, необходимо помнить, что только пользовательская переменная может иметь тип **«Символ» (текст)**.

Для присваивания символьных значений используется следующий формат инициализации:

```
< имя переменной >='< текстовая строка >';
```

Кроме того, можно выводить значения любых переменных в текстовые окна. Формат вывода имеет вид:

```
< имя/номер текстового окна >='< текст > @[< имя переменной >] < текст >';
```

К параметрам команд CLData можно обращаться напрямую, минуя системные переменные. Обращение к параметрам имеет формат:

```
< имя пользовательской переменной >=FR[< номер параметра команды >];
```

Структура некоторых основных команд CLData с номерами параметров приведена в приложении «Структура основных транслируемых команд CLData».

Для удобства использования все системные переменные распределены по разделам:

- Координаты инструмента
- Круговая интерполяция
- Последующие перемещения инструмента
- Совмещенные перемещения
- Геометрия и номер позиции инструментов
- Включение/выключение корректоров
- Выстой
- Положение металла
- Управление шпинделем
- Управление подачей
- Резьба (токарная)
- Учетные параметры программы, детали и станка
- Переменные для работы с постоянными циклами
- Координаты безопасной позиции
- Координаты точки прижима

- Номер стола
- Номер трубопровода СОЖ
- Начало цикла
- Переменные для работы с подпрограммами
- Системные переменные для работы с контурами и элементами CLData
- Переменные для работы с пользовательскими командами
- Переменные для работы с трансформами
- 5-ти координатные перемещения
- Вспомогательные переменные
- Пользовательские переменные, используемые в ранних версиях адаптера

Координаты инструмента

Ниже, в таблице, перечислены системные переменные, принимающие определённое значение при отработке следующих команд CLData:

- Линейная интерполяция (код 181)
- Круговая интерполяция (код 183)
- Совмещенное перемещение (код 41)
- Поворот (код 40)

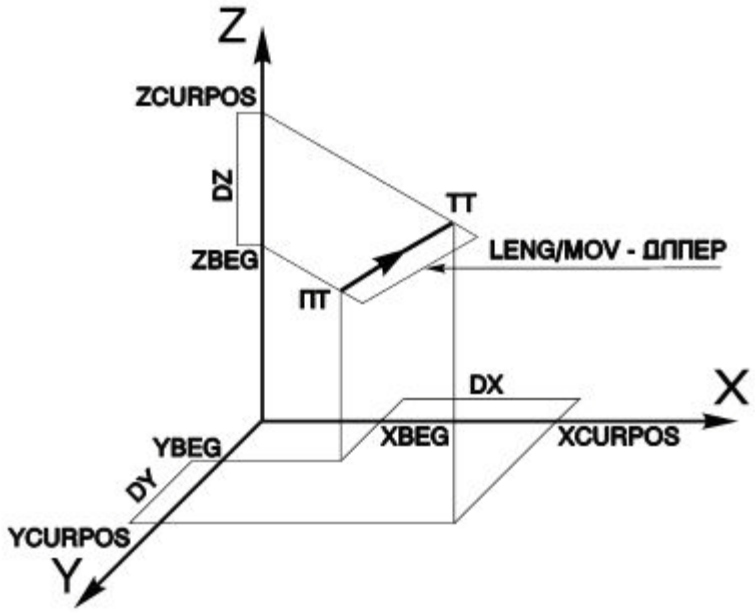
Для расшифровки величин, записываемых в переменные, введём следующие понятия:

Текущая точка (ТТ) — это точка, в которой находится инструмент.

Предыдущая точка (ПТ) — это точка, из которой произошло перемещение в **ТТ**.

Перемнная	Значение
XT или **XCURPOS**	координата X ТТ
YT или **YCURPOS**	координата Y ТТ
ZT или **ZCURPOS**	координата Z ТТ
XC или **XBEG**	координата X ПТ
YC или **YBEG**	координата Y ПТ
ZC или **ZBEG**	координата Z ПТ
DX	перемещение по оси X, равное XT-XC
DY	перемещение по оси Y, равное YT-YC
DZ	перемещение по оси Z, равное ZT-ZC
AT или **ACURPOS**	угловая координата по оси A ТТ
BT или **BCURPOS**	угловая координата по оси B ТТ

СТ или **CCURPOS**	угловая координата по осиС ТТ
DA	угловое перемещение по оси А, равное АТ-АС
DB	угловое перемещение по оси В, равное ВТ-ВС
DC	угловое перемещение по оси С, равное СТ-СС
ДЛПЕР или **LENG/MOV**	угловая координата по осиС ТТ



Величины переменных

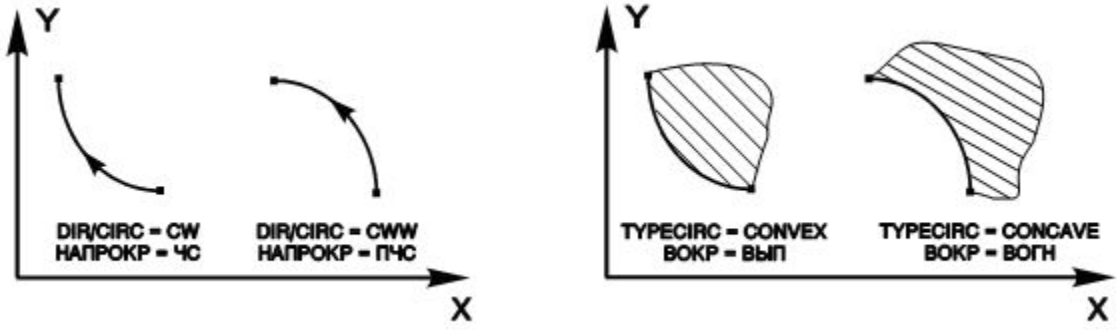
Круговая интерполяция

Ниже, в таблице, перечислены системные переменные, принимающие определённое значение при отработке команды CLData — Круговая интерполяция (код 183).

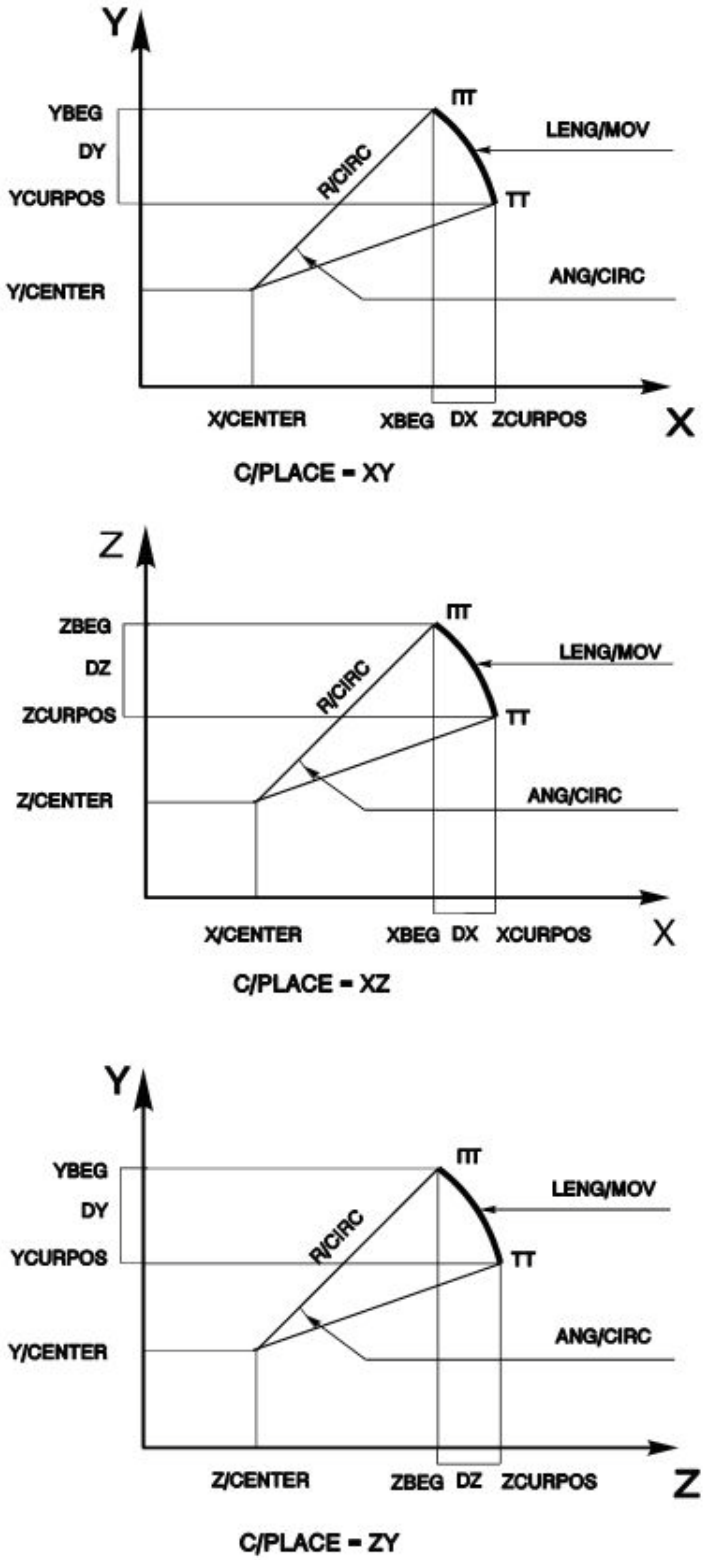
Для расшифровки величин, записываемых в переменные, введём понятие **«точки центра окружности»**.

Точка центра окружности (ТЦО) — это центр дуги, по которой произошло перемещение из предыдущей точки (**ПТ**) в текущую (**ТТ**).

Перемнная	Значение
ХЦОКР или **X/CENTER**	координата X ТЦО
YT или **Y/CENTER**	координата Y ТЦО
ZT или **Z/CENTER**	координата Z ТЦО
РОКР или **R/CIRC**	радиус дуги, по которой произошло перемещение и ПТ в ТТ
НАПРОКР или **DIR/CIRC**	направление движения по дуге — принимает значения ЧС (CW) или ПЧС (CWW)
ЧС или **CW**	движение по часовой стрелке
ПЧС или **CWW**	движение против часовой стрелки
ВОКРили **TYP/CIRC**	вид дуги, по которой происходит перемещение — принимает значения ВЫП или ВОГН
ВЫП или **CONVEX**	выпуклая дуга
ВОГН или **CONCAVE**	вогнутая дуга
ЦУГОЛ или **ANG/CIRC**	центральный угол дуги, по которой происходит круговая интреполяция
КПЛОК или **ПЛОКР** или **C/PLACE**	плоскость движения по дуге окружности — принимает значения XY, YZ, ZX
XY или **YX**	плоскость движения по дуге окружности — XY
YZ или **ZY**	плоскость движения по дуге окружности — YZ
ZX или **XZ**	плоскость движения по дуге окружности — ZX



Слева: направление движения по дуге. Справа: тип дуги.



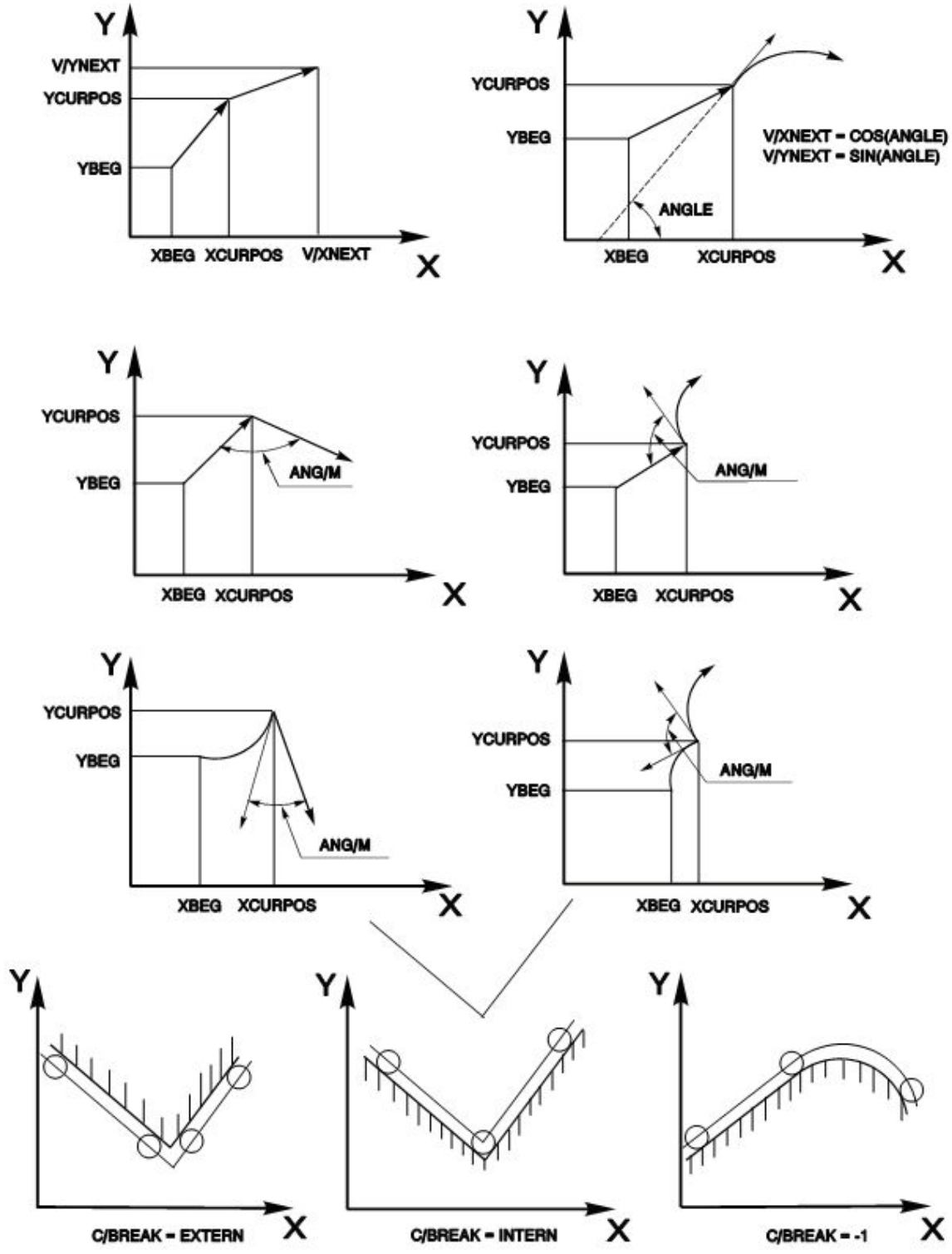
Числовые характеристики дуг в плоскостях XY, XZ и YZ соответственно

Последующие перемещения инструмента

ТСЛ (следующая точка) — точка обработки детали, в которую произойдет следующее перемещение инструмента.

Перемнная	Значение
ВХСЛили **V/XNEXT**	вектор перемещения по оси в следующую точку обработки детали. Если следующее перемещение по траектории обработки — линейная интерполяция, **ВХСЛ** принимает значение координаты **X ТСЛ**. В случае круговой интерполяции, **ВХСЛ** принимает значение косинуса касательной к окружности, по которой будет происходить перемещение, в точке начала дуги
ВУСЛ или **V/YNEXT**	Вектор перемещения по оси в следующую точку обработки детали. Если следующее перемещение по траектории обработки — линейная интерполяция, **ВУСЛ** принимает значение координаты **Y ТСЛ**. В случае круговой интерполяции, **ВУСЛ** принимает значение косинуса касательной к окружности, по которой будет происходить перемещение, в точке начала дуги
ВZСЛ или **V/ZNEXT**	вектор перемещения по оси в следующую точку обработки детали. Если следующее перемещение по траектории обработки — линейная интерполяция, **ВZСЛ** принимает значение координаты **Z ТСЛ**. В случае круговой интерполяции, **ВZСЛ** принимает значение косинуса касательной к окружности, по которой будет происходить перемещение, в точке начала дуги
УГНАП или **ANG/M**	угол перелома траектории (в радианах)
К/ПЕРЕЛ или **C/BREAK**	код перелома траектории. Принимает значение **ВНЕШ (EXTERN)** при внешнем обходе угла, **ВНУТ (INTERN)** при обработке внутреннего угла, в случае сопряжения **К/ПЕРЕЛ=-1**
XNEXT	координата X следующего перемещения
YNEXT	координата Y следующего перемещения
ZNEXT	координата Z следующего перемещения
XCNEXT	координата X центра дуги следующего перемещения, если следующее перемещение — круговая интерполяция
YCNEXT	координата X центра дуги следующего перемещения, если следующее перемещение — круговая интерполяция
ZCNEXT	координата X центра дуги следующего перемещения, если следующее перемещение — круговая интерполяция
RNEXT	радиус дуги следующего перемещения, если следующее перемещение — круговая интерполяция
CNEXTM	код следующего перемещения. Принимает следующие значения: **181** — линейное перемещение, **183** — круговая интерполяция, **35** — после текущего перемещения встречена смена инструмента, **4** — после текущего перемещения обнаружен конец программы.
CCIRCM	направление движения по дуге следующего перемещения, если следующее перемещение – круговая интерполяция.

Принимает значения **ЧС (CW)** или **ПЧС (CWW)**



Последующие перемещения инструмента

Совмещенные перемещения

Совмещенные перемещения характерны для электроэрозионной обработки и представляют собой одновременные перемещения двух рабочих органов, в которых закреплен инструмент.

Совмещенное перемещение — дополнительное перемещение для второго рабочего органа (для электроэрозионной обработки это движение верхней головки). В отличии от основных перемещений, команды совмещенных перемещений имеют другие коды CLData: **совмещенное линейное перемещение** имеет код **185**, **совмещенное перемещение по дуге** имеет код **187**.

Примечание

Совмещенные перемещения формируются только для оборудования, чей тип в паспорте постпроцессора как определен **«EDM (2 контура)»**. Для типа **«EDM»** формируются мультиперемещения (код 41), а для типа **«EDM (точка и вектор)»** — 5-ти координатные перемещения (код 182).

Перемнная	Значение
XCUR2	координата X точки совмещенного перемещения
YCUR2	координата Y точки совмещенного перемещения
ZCUR2	координата Z точки совмещенного перемещения
XOLD2	координата X начальной точки совмещенного перемещения
YOLD2	координата Y начальной точки совмещенного перемещения
ZOLD2	координата Z начальной точки совмещенного перемещения
X/CENT2	координата X центра дуги совмещенного перемещения, если совмещенное перемещение проходит по дуге
Y/CENT2	координата Y центра дуги совмещенного перемещения, если совмещенное перемещение проходит по дуге
Z/CENT2	координата Z центра дуги совмещенного перемещения, если совмещенное перемещение проходит по дуге
DIR/CIR2	направление движения по дуге совмещенного перемещения, если совмещенное перемещение проходит по дуге. Принимает значения **ЧС (CW)** или **ПЧС (CWW)**

Геометрия и номер позиции инструментов

Этим системным переменным присваиваются значения параметров при отработке команды CLData **«Загрузка инструмента (код 35)»**.

Перемнная	Значение
ТИНСТР или **N/TOOL**	номер позиции загружаемого инструмента
CURTOOL	номер позиции текущего инструмента
ИНСТР1 или **FIRSTOOL**	номер позиции первого инструмента в программе
СЛИНСТР или **NEXTOOL**	номер позиции инструмента, который будет загружен следующим по ходу программы. Если загружен последний инструмент в программе, значение **СЛИНСТР** формируется в зависимости от постпроцессора: при необходимости загрузки первого инструмента в конце программы **СЛИНСТР=ИНСТР1**, в противном случае **СЛИНСТР=0**
РИНСТР или **RAD/TOOL**	радиус загружаемого документа
РСЛИНСТР или **NEXT/RTL**	радиус инструмента, который будет загружен следующим в программе
ВЫЛЕТХ или **XOVERH**	вылет инструмента по оси X
ВЫЛЕТУ или **YOVERH**	вылет инструмента по оси Y
ВЫЛЕТZ или **ZOVERH**	вылет инструмента по оси Z
ANG/TOOL	угол инструмента в радианах
L/TOOL	длина режущей части инструмента
TL/TOOL	общая длина иснтрумента
COD/TOOL	код загружаемого инструмента, может принимать значения: 1 — фреза; 2 — сверло; 3 — центровка; 4 — зенкер; 5 — развёртка; 6 — метчик; 7 — резец; 8 — пуансон; 9 — проволока; 10 — лазер; 11— резак.
N/TUR	номер туреты для двухтуретной обработки
N/CP	номер контрольной точки при многотуретной обработке

Включение/выключение корректоров

Этим системным переменным присваиваются значения параметров при отработке следующих команд CLData:

- загрузка инструмента (код 35)
- включить корректор по оси X (код 703)
- включить корректор по оси Y (код 704)
- включить корректор по оси Z (код 705)
- включить радиусный корректор (код 706)
- выключить корректор по оси X (код 707)
- выключить корректор по оси Y (код 708)
- выключить корректор по оси Z (код 709)
- выключить радиусный корректор (код 710)

Перемнная	Значение
ВКЛКОРХ или **XCOMPON**	номер включаемого линейного корректора по оси X. После отработки команды **«Включить корректор по оси X»** значение **ВКЛКОРХ** сбрасывается
ВКЛКОРУ или **YCOMPON**	номер включаемого линейного корректора по оси X. После отработки команды **«Включить корректор по оси Y»** значение **ВКЛКОРУ** сбрасывается
ВКЛКОРZ или **ZCOMPON**	номер включаемого линейного корректора по оси X. После отработки команды **«Включить корректор по оси Z»** значение **ВКЛКОРZ** сбрасывается
ВКЛКОРR или **RCOMPON**	номер включаемого линейного корректора по оси X. После отработки команды **«Включить радиусный корректор»** значение **ВКЛКОРR** сбрасывается
ВЫКЛКОРХ или **RCOMPOFF**	номер включаемого линейного корректора по оси X. После отработки команды **«Выключить корректор по оси X»** значение **ВЫКЛКОРХ** сбрасывается
ВЫКЛКОРУ или **RCOMPOFF**	номер включаемого линейного корректора по оси X. После отработки команды **«Выключить корректор по оси Y»** значение **ВЫКЛКОРУ** сбрасывается
ВЫКЛКОРZ или	номер включаемого линейного корректора по

RCOMPOFF	оси X. После отработки команды **«Выключить корректор по оси Z»** значение **ВЫКЛКОРZ** сбрасывается
ВЫКЛКОРR или **RCOMPOFF**	номер включаемого линейного корректора по оси X. После отработки команды **«Выключить радиусный корректор»** значение **ВЫКЛКОРR** сбрасывается

Выстой

Этим системным переменным присваиваются значения параметров при отработке команды CLData **«Выстой» (код 27)**.

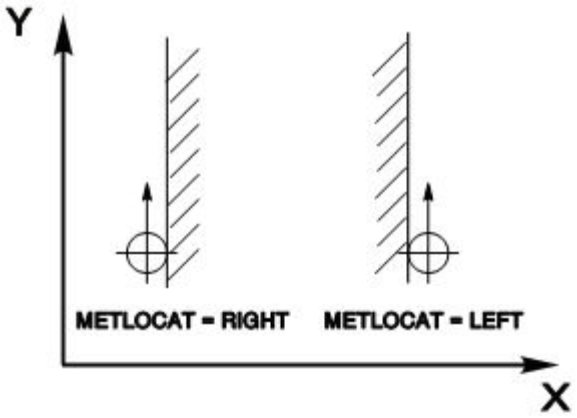
Перемнная	Значение
ВЫСТОБ или **SPIN/DW**	величина выстоя, выраженная в количестве оборотов шпинделя
ВЫСТВР или **TIME/DW**	величина выстоя, выраженная в секундах
КОДВЫСТ или **C/DWELL**	показывает, каким образом была задана величина выстоя при формировании перехода технологического объекта. Принимает значения **0**, если выстой был задан в секундах или **1**, если выстой был задан в оборотах шпинделя. Вне зависимости от способа задания выстоя, его величина рассчитывается и оборотах и в секундах и записывается в системные переменные **ВЫСТОБ(SPIN/DW)** и **ВЫСТВР(TIME/DW)** соответственно

Положение металла

Этим системным переменным присваиваются значения параметров при отработке команды CLData — «Включить радиусный корректор» (код 706).

Перемнная	Значение
ПЛМ или **METLOCAT**	положение металла. Переменная принимает значения **СЛВ** или **СПР**
СЛВ или **LEFT**	металл слева [1]
СПР или **RIGHT**	металл справа [0]

Поясняющий рисунок



Слева: металл справа от инструмента, справа: металл слева от инструмента

Управление шпинделем

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Включить шпиндель» (код 24).

Перемнная	Значение
N или **NOБ** или **SPIN**	частота вращения шпинделя, об/мин
V или **VРЕЗ** или **VC**	скорость резания, м/мин
NДИАП или **N/RANGE**	номер заданного диапазона шпинделя
НВШПили **DIR/SPIN**	Направление вращения шпинделя, принимает значение **ЧС (CW)** или **ПЧС (CWW)**
КОДШПИНД или **COD/SPIN**	код в ряду шпинделя
СЛNОБ или **NEXTSPIN**	величина оборотов шпинделя, которая будет включена далее по ходу программы, об/мин
СЛVРЕЗ или **NEXTVC**	величина оборотов шпинделя, которая будет включена далее по ходу программы, пересчитанная в мм/ми
СЛНДИАП или **NEXT/RNG**	номер диапазона шпинделя, который будет задействован далее по ходу программы
СЛНВШП или **NX/DIRSP**	направление вращения шпинделя, которое будет включено далее по ходу программы, принимает значение **ЧС (CW)** или **ПЧС (CWW)**
СЛКОДШП или **NEXTC/SP**	следующий код в ряду шпинделя
УГОРШП или **ANG/SPIN**	угол ориентации шпинделя при ориентированном останове
СЛККШП	показывает, каким образом будут заданы обороты шпинделя при смене текущей загрузки шпинделя. Принимает значения **об/мин** или **м/мин**. Вне зависимости от способа задания оборотов шпинделя его величина

	рассчитывается и в оборотах в минуту и в метрах в минуту и записывается в системные переменные **СЛNОБ(NEXTSPIN)** и **СЛVРЕЗ(NEXTVC)** соответственно
ККШП или **CUTCOND**	показывает, каким образом были заданы обороты шпинделя при формировании технологического объекта. Принимает значения **об/мин** или **м/мин**. Вне зависимости от способа задания оборотов шпинделя его величина рассчитывается и в оборотах в минуту и в метрах в минуту и записывается в системные переменные **СЛNОБ(NEXTSPIN)** и **СЛVРЕЗ(NEXTVC)** соответственно
MAX/SPIN	максимально допустимая величина оборотов шпинделя (используется в токарной обработке)

Управление подачей

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Включить рабочую подачу (код 23)».

Перемнная	Значение
S или **ПОДМИН** или **FEED**	подача, мм/мин
SОБ или **ПОДОБ** или **FEEDS/T**	подача, мм/об
СЛПОДМИН или **NEXT/F/T**	величина подачи, которая будет включена следующей, мм/мин
СЛПОДОБ или **NEXT/FD**	величина подачи, которая будет включена позже, мм/об
КЗПОД или **CSETF**	показывает, каким образом была задана подача при формировании перехода технологического объекта. Принимает значения **мм/мин** или **мм/об**. Вне зависимости от способа задания подачи ее величина рассчитывается и в миллиметрах в минуту и в миллиметрах на оборот и записывается в системные переменные **S (FEED)** и **SОБ (FEEDS/T)** соответственно

Резьба (токарная)

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Нарезать резьбу (код 94)».

Перемнная	Значение
П/РЕЗЬБЫ или **PR/THRD**	профиль резьбы, принимает значение **МЕТРИ**, **ТРАПЕЦ**, **УПОРНАЯ**, **ТРУБНАЯ** или **ПРЯМОУГ**
МЕТРИЧ или **METRIC**	метрическая [0]
ТРАПЕЦ или **TRAPEZ**	трапецеидальная [1]
УПОРНАЯ или **BUTTRESS**	упорная [2]
ТРУБНАЯ или **PIPE**	трубная [3]
ПРЯМОУГ или **SQUARE**	прямоугольная [4]
В/РЕЗЬБЫ или **KINDTHRD**	вид резьбы, принимает значения **НАРУЖ** или **ВНУТ**
НАРУЖ или **EXTERN**	наружная [0]
ВНУТ или **INTERN**	внутренняя [1]
Т/РЕЗЬБЫ или **TYPETHRD**	тип резьбы, принимает значения **ПРАВАЯ** или **ЛЕВАЯ**
ПРАВАЯ или **RIGHT**	правая [0]
ЛЕВАЯ или **LEFT**	левая [1]
СБЕГ или **RUN/OUT**	сбег резьбы, принимает значения **ВКЛ** или **ВЫКЛ**
ВКЛ или **ON**	включен [0]
ВЫКЛ или **OFF**	выключен [1]
Ф/РЕЗЬБЫ/ или **FORMTHRD**	форма резьбы, принимает значения **ЦИЛ** или **КОНИЧ**
ЦИЛ или **STRAIGHT**	цилиндрическая [0]
КОНИЧ или **TAPER**	коническая [1]
УГР или **ANG/THRD**	коническая [1]
ШАГР или **PITCH/T**	коническая [1]
ДЛИНР или **LENGTH**	длина резьбы
КПРОХР или **NOF/THST**	количество проходов при нарезании]

Учетные параметры программы, детали и станка

Этим системным переменным присваиваются значения параметров при отработке следующих команд CLData:

- Программа (код 1)
- Деталь (код 2)
- Станок (код 3)

Перемнная	Значение
НПРОГ или **N/PROG**	номер программы, формируемый из имени программы. Если имя программы не является числом, то **НПРОГ=1**
ИПРОГ или **NAMEPROG**	имя программы (текстовая информация)
ИДЕТ или **NAMEPART**	наименование детали (текстовая информация)
НДЕТ или **N/PART**	обозначение детали (текстовая информация)]

Переменные для работы с постоянными цикалми

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Цикл» (код 36).

Перемнная	Значение
NЦИКЛ или **N/CYCLE**	номер цикла
NEXTCYCL	номер цикла, который будет обрабатываться после текущей команды CLData
КПАРЦ или **NOF/PARC**	количество параметров цикла
ПАРЦ<номер параметра> или **PAR/C<номер параметра>**	параметры цикла
ПНЦТ или **IND/CP**	порядковый номер текущей команды **ЦИКЛ** (код 36). Учитываются только стандартные циклы с номерами от 69 до 77 и от 81 до 89
NEXTCYCL	номер следующего цикла
NUM/PC или **IND/CP**	порядковый номер цикла в маршруте обработки

Ниже в таблице приведены параметры, установленные для стандартных циклов.

Параметр	Значение
ПАРЦ1 или **PAR/C1**	подача цикла
ПАРЦ2 или **PAR/C2**	координата Z вывода (плоскость отвода инструмента). При просмотре CLData эта величина относительная
ПАРЦ3 или **PAR/C3**	общая глубина отверстия, включает в себя недобег и перебег

ПАРЦ4 или **PAR/C4**	величина выстоя
ПАРЦ5 или **PAR/C5**	число заглублений минус единица
ПАРЦ6 или **PAR/C6**	реверс направления вращения
ПАРЦ7 или **PAR/C7**	восстановление направления вращения
ПАРЦ8 или **PAR/C8**	общая глубина отверстия, включает в себя недобег и перебег
ПАРЦ9 или **PAR/C9**	коэффициент уменьшения глубины сверления
ПАРЦ10 или **PAR/C10**	величина недобега
ПАРЦ11 или **PAR/C11**	величина перебега
ПАРЦ12 или **PAR/C12**	величина вывода
ПАРЦ13 или **PAR/C13**	величина шага резьбы
ПАРЦ14 или **PAR/C14**	величина угла наклона инструмента к оси X
ПАРЦ15 или **PAR/C15**	величина угла наклона инструмента к оси Y
ПАРЦ16 или **PAR/C16**	величина угла ориентации шпинделя
ПАРЦ17 или **PAR/C17**	величина отвода инструмента от оси отверстия
ПАРЦ18 или **PAR/C18**	стартовая глубина начала обработки
ПАРЦ19 или **PAR/C19**	код задания подачи. Если **PAR/C19=1** — подача задана в мм/мин, если **PAR/C19=2** — подача задана в об/мин
ПАРЦ20 или **PAR/C20**	код задания выстоя. Если **PAR/C20=1** — выстой задан в секундах, если **PAR/C20=2** — выстой задан в оборотах
ПАРЦ21 или **PAR/C21**	код задания шпинделя. Если **PAR/C21=2** — задано число оборотов, если **PAR/C21=3** — задана скорость резания
ПАРЦ22 или **PAR/C22**	число оборотов шпинделя или величина скорости резания
ПАРЦ23 или **PAR/C23**	код координаты выхода. Если **PAR/C23=0** — координата Z, если **PAR/C23=1** — координата X
ПАРЦ24 или **PAR/C24**	код задания торца. Если **PAR/C24=0** – торец не задан, если **PAR/C24=1** — задан левый торец, если **PAR/C24=2** — задан правый торец
ПАРЦ25 или **PAR/C25**	направляющий косинус к оси X
ПАРЦ26 или **PAR/C26**	направляющий косинус к оси Y
ПАРЦ27 или **PAR/C27**	направляющий косинус к оси Z
ПАРЦ28 или **PAR/C28**	направление обработки. **Если PAR/C28=0** — прямая обработка, если **Если PAR/C28=1** — обратная обработка
ПАРЦ29 или **PAR/C29**	вывод инструмента. Если **PAR/C29=0** — вывод на холостом ходу, если **PAR/C29=1** — вывод на подаче, если **PAR/C29=2** — вывод ручной

Примечание

Для параметров, чей номер больше 20, используется прямой доступ через формат **< имя пользовательской переменной >=FR[< номер параметра команды >];**. Структура команды CLData «Цикл» (код 36) приведена в приложении «Структура основных транслируемых команд CLDat»а.

Координаты безопасной позиции

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Безопасная позиция» (код 451).

Перемнная	Значение
ХБЕЗПОЗ или **X/GOHOME**	координата X безопасной позиции
YБЕЗПОЗ или **Y/GOHOME**	координата Y безопасной позиции
ZБЕЗПОЗ или **Z/GOHOME**	координата Z безопасной позиции

Координаты точки прижима

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Перезахват» (код 29).

Перемнная	Значение
ХПРИЖ или **XPRESS**	координата X точки прижима
YПРИЖ или **YPRESS**	координата Y точки прижима
СМЗАЖ или **CLAMPMOV**	смещение зажима

Номер стола

Этой системной переменной присваивается значение при отработке команды CLData «Смена стола» (код 34).

Перемнная	Значение
НСТОЛ или **N/TABLE**	номер стола

Номер трубопровода СОЖ

Этой системной переменной присваивается значение при отработке команды CLData «Включить СОЖ» (код 26).

Перемнная	Значение
НСОЖ или **N/COOL**	номер трубопровода СОЖ

Начало цикла

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Начало цикла» (код 401).

Если начало цикла (НЦ) задано координатами, следующие переменные принимают значения:

Перемнная	Значение
ХНЦ или **XHOME**	координата X точки прижима
YНЦ или **YHOME**	координата Y точки прижима
ZНЦ или **ZHOME**	смещение зажима

Если начало цикла задано номером регистра, присваивается значение переменной:

Перемнная	Значение
NНЦ или **N/HOME**	номер регистра с координатами **НЦ**

Если начало цикла задано корректорами, присваиваются значения переменным:

Перемнная	Значение
KXНЦ или **XCHOME**	номер корректора по оси X
KYНЦ или **YCHOME**	номер корректора по оси Y
KZНЦ или **ZCHOME**	номер корректора по оси Z

Переменные для работы с подпрограммами

Этим системным переменным присваиваются значения параметров при отработке следующих команд CLData:

- Вызов подпрограммы (код 223)
- Начало подпрограммы (код 252)

Перемнная	Значение
НПОДПР или **NПОДПР** или **N/SUB**	номер вызываемой подпрограммы. Формируется на основе имени подпрограммы. Если имя не число, эта переменная принимает значение 1
КПОВТППГ	количество повторений подпрограммы при вызове
НФППГ или **NCRSUB**	номер формируемой подпрограммы. Формируется на основе имени подпрограммы. Если имя не число, эта перемнная принимает значение 1
КПАРППГ	количество параметров вызываемой подпрограммы
ПАРПП<номер параметра>	величины параметров вызываемой подпрограммы
XFSUB	координата X первого перемещения вызываемой подпрограммы
YFSUB	координата Y первого перемещения вызываемой подпрограммы
ZFSUB	координата Z первого перемещения вызываемой подпрограммы
ИВППГ	имя вызываемой подпрограммы. Формируется на основе имени подпрограммы (текстовая информация)
ИФППГ	имя формируемой подпрограммы. Формируется на основе имени подпрограммы (текстовая информация)

Системные переменные для работы с контурами и элементами CLData

Эти переменные устанавливаются непосредственно перед трансляцией CLData.

Перемнная	Значение
COEFMM	коэффициент, показывающий текущие единицы измерения. Если **COEFMM=1**, то текущие единицы измерения миллиметры, а если **COEFMM=2,54** – дюймы
XCLDMIN	минимальная координата X, полученная в CLData
YCLDMIN	минимальная координата Y, полученная в CLData
ZCLDMIN	минимальная координата Z, полученная в CLData
XCLDMIN	максимальная координата X, полученная в CLData
YCLDMIN	максимальная координата Y, полученная в CLData
ZCLDMIN	максимальная координата Z полученная в CLData
RTOOLMAX	максимальный радиус инструмента, используемого в маршруте обработки

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Контур» (код 121). Команда формируется в CLData, если был задан «Цикл пользователя» (код 121) на контуре.

Перемнная	Значение
XCONTMIN	минимальная координата X контура обработки
YCONTMIN	минимальная координата Y контура обработки
XCONTMAX	максимальная координата X контура обработки
YCONTMAX	максимальная координата Y контура обработки
PLANEMAX	максимальная плоскость конструктивного элемента, заданная в маршруте обработки
QTY/EL	количество элементов в контуре
CODE/EL	код элемента контура: 2 — отрезок, 3 — дуга
XBEG/EL	начальная координата X элемента контура
YBEG/EL	начальная координата Y элемента контура
ZBEG/EL	начальная координата Z элемента контура
XEND/EL	конечная координата X элемента контура
XEND/EL	конечная координата Y элемента контура
XEND/EL	конечная координата Z элемента контура
DIRC/EL	направление движения по дуге, принимает значения ЧС и ПЧС
RCIRC/EL	радиус дуги

XCEN/EL	координата X центра дуги
YCEN/EL	координата Y центра дуги
ZCEN/EL	координата Z центра дуги

Переменные для работы с пользовательскими командами

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Команда пользователя» (код 459).

Перемнная	Значение
N/USFUNC	номер команды пользователя
NEXTUSC	номер команды пользователя, которая будет отрабатываться после текущей команды CLData
PUSFUN<номер параметра>	параметры команды пользователя
X/COORD	координата X пользовательской системы координат
Y/COORD	координата Y пользовательской системы координат
Z/COORD	координата Z пользовательской системы координат
A/COORD	угол пользовательской системы координат

Переменные для работы с трансформами

Этим системным переменным присваиваются значения параметров при отработке команды CLData «Трансформ» (код 10123), а также в процессе преобразования трансформа, который может быть организован пользователем в любой момент.

Перемнная	Значение
SYSCOOR	матрица, содержащая данные об СК детали
CLDCOOR	матрица, содержащая данные об СК КЭ
COORUS<номер>	матрица, содержащая данные о промежуточных системах координат, используемых при вычислении. Всего можно назначить 10 промежуточных СК
CALCOOR	матрица, содержащая данные для пересчета координат точек CLData в координаты точек управляющей программы

5-ти координатные перемещения

Этим системным переменным присваиваются значения параметров при отработке следующих команд CLData:

- Линейные векторные перемещения (код 182)
- Векторные круговые перемещения (код 184)

- Векторные линейные перемещения с коррекцией (код 189)
- Криволинейное перемещение (код 190)
- Дополнительное криволинейное перемещение (код 191)

Перемнная	Значение
VSP/X	косинус угла, определяющего положение оси инструмента относительно оси X СК детали
VSP/Y	косинус угла, определяющего положение оси инструмента относительно оси Y СК детали
VSP/Z	косинус угла, определяющего положение оси инструмента относительно оси Z СК детали
VSNP/X	косинус угла, определяющего положение нормали к поверхности в точке касания ее инструментом относительно оси X СК детали
VSNP/Y	косинус угла, определяющего положение нормали к поверхности в точке касания ее инструментом относительно оси Y СК детали
VSNP/Z	косинус угла, определяющего положение нормали к поверхности в точке касания ее инструментом относительно оси Z СК детали

Вспомогательные переменные

Эти переменные устанавливаются непосредственно перед трансляцией **CLData** и контролируются на протяжении всего процесса трансляции.

Перемнная	Значение
НОМКДР или **N/BLOCK**	подача, мм/мин
CDEFAULT	подача, мм/об
ТКОМ или **REM**	величина подачи, которая будет включена следующей, мм/мин
КСЛКОМ или **NEXT/COD**	величина подачи, которая будет включена позже, мм/об
КОТВОД или **C/GOHOME**	код отвода. Если **КОТВОД=ВКЛ**, то в данный генерируются команды отвода инструмента. Если **КОТВОД=ВЫКЛ**, то идет обычная отработка команды CLData
TIME	переменная, которая содержит время обработки при выключенном условии автоматического расчета времени адаптером. Для отключения автоматического времени расчета необходимо дать команду **РАСЧВР=ВЫКЛ;**, а в переменной **TIME** накапливать время работы УП
РАСЧВР или **AUTOTIME**	переменная, которая содержит признак автоматического расчета времени обработки.

	Начальное значение переменной **РАСЧВР=ВКЛ**. Чтобы иметь возможность рассчитывать время работы УП в постпроцессоре, необходимо дать команду **РАСЧВР=ВЫКЛ;** или **AUTOTIME=OFF;**, а время работы сохранять в переменной **TIME**
ТВСТ или **INS/TEXT**	текст вставки
SIZEFILE	переменная, в которой накапливается количество символов, выведенных в кадры управляющей программы
DAY	текущий день недели
MONTH	текущий месяц
YEAR	текущий год
HOUR	текущий час
MINUTE	текущая минута
SECOND	текущая секунда

Пользовательские переменные, используемые в ранних версиях адаптера

Имена паременных, перечисленные ниже, допускалось использовать в качестве пользовательских в ранних версиях адаптера. В актуальных версиях, для того, чтобы инициализировать пользовательскую переменную, ей достаточно присвоить имя, которое начинается с символа «_». Например, _FEED.

```
P1...P10
G1(G)...G20
NK
E
X(X1), X2
Y(Y1), Y2
Z(Z1), Z2
D(D1)...D3
I, J, K
T, F, SK
L(L1)...L3
A, B, C
H(H1)...H3
M(M1)...M3
R, Q, LF
```

Пользовательские команды и циклы обработки

В системе **ADEM** имеется возможность в дополнение к существующим средствам проектирования маршрута обработки (технологические команды, технологические переходы и конструктивные элементы) создавать свои собственные команды и циклы обработки. Пользовательские команды и циклы создаются на основе так называемых «**ini-файлов**». Эти файлы должны находиться в папке **«...ADEM/GMD/INI/CommonINI/CNC/»** и иметь формат **< имя файла >.ini**. Кроме отдельных команд и циклов можно также добавлять пользовательские параметры к основным параметрам технологических переходов и команд, входящих в стандартную поставку системы **ADEM**.

Примеры отработки в постпроцессоре пользовательских команд, циклов и параметров можно посмотреть в разделе «Примеры».

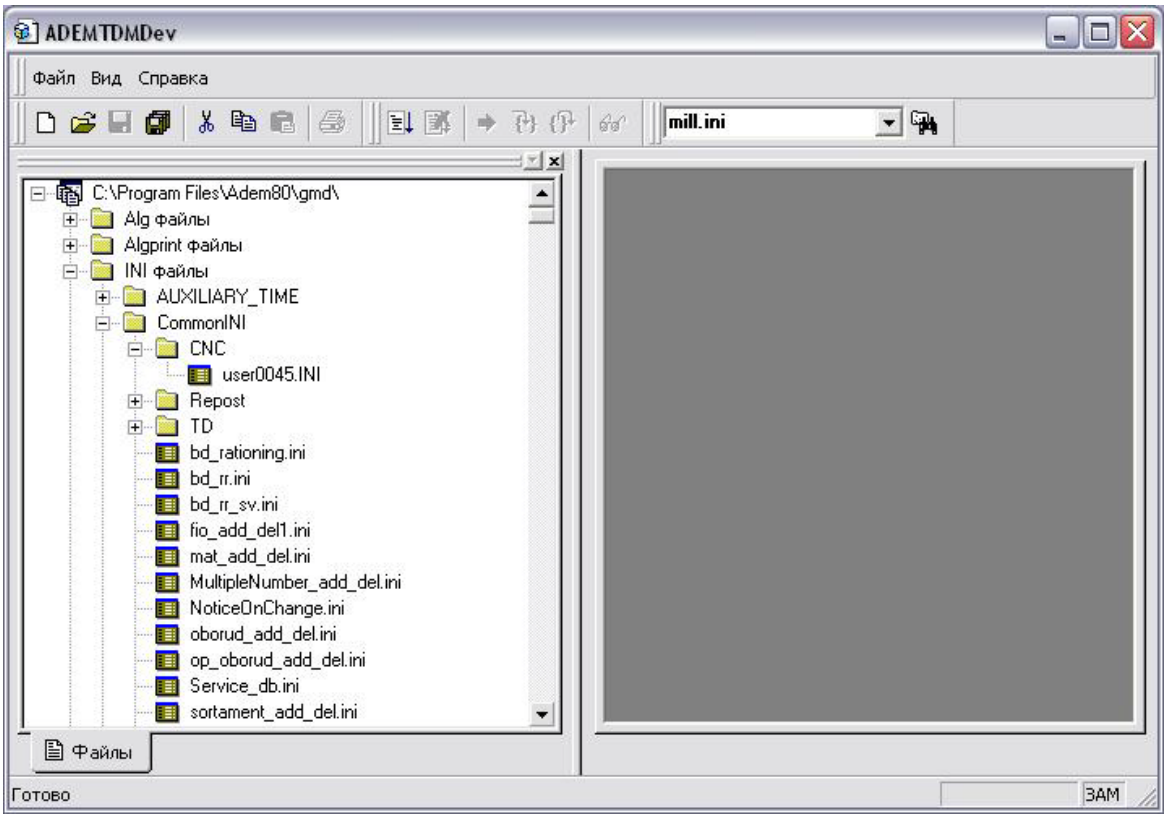
Подробное описание способов создания пользовательских команд, циклов и параметров содержится в разделах:

- создание пользовательской команды
- создание пользовательского цикла обработки
- добавление созданных команд и циклов в меню выбора
- добавление пользовательских параметров к основным параметрам технологического перехода

Создание пользовательской команды

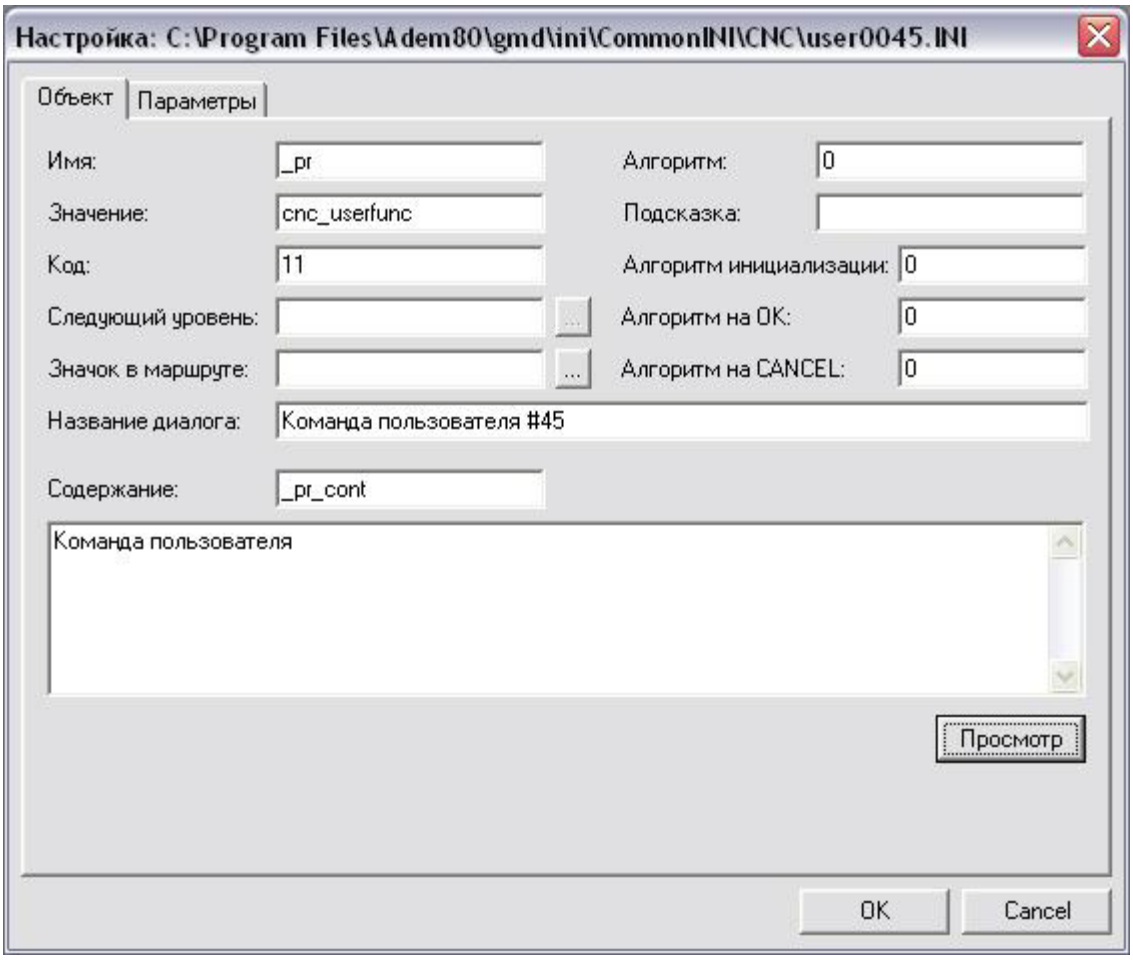
Диалог ТК **«Пользовательская команда»** формируется в соответствии с шаблоном, содержащимся в настроечном файле с расширением ***.ini**. Файл расположен в каталоге **"...\GMD\INI\CommonINI\CNC\"**. В качестве содержится файл **"user0045.ini"**, который может быть использован в качестве образца.

Чтобы изменить шаблон или создать новый, необходимо выполнить команду меню **«Модуль»** — **«Adem CAPP Developer»**. На вкладке **«Файлы»** раскройте папку **"INI файлы\CommonINI\CNC\"**, выберите требуемый **INI**-файл и откройте его двойным щелчком мыши.



Диалоговое окно «ADEM TDMDev»

После выбора имени файла откроется диалоговое окно **«Настройка: объект»**, позволяющее работать с параметрами создаваемой команды.



Диалоговое окно создания пользовательской команды

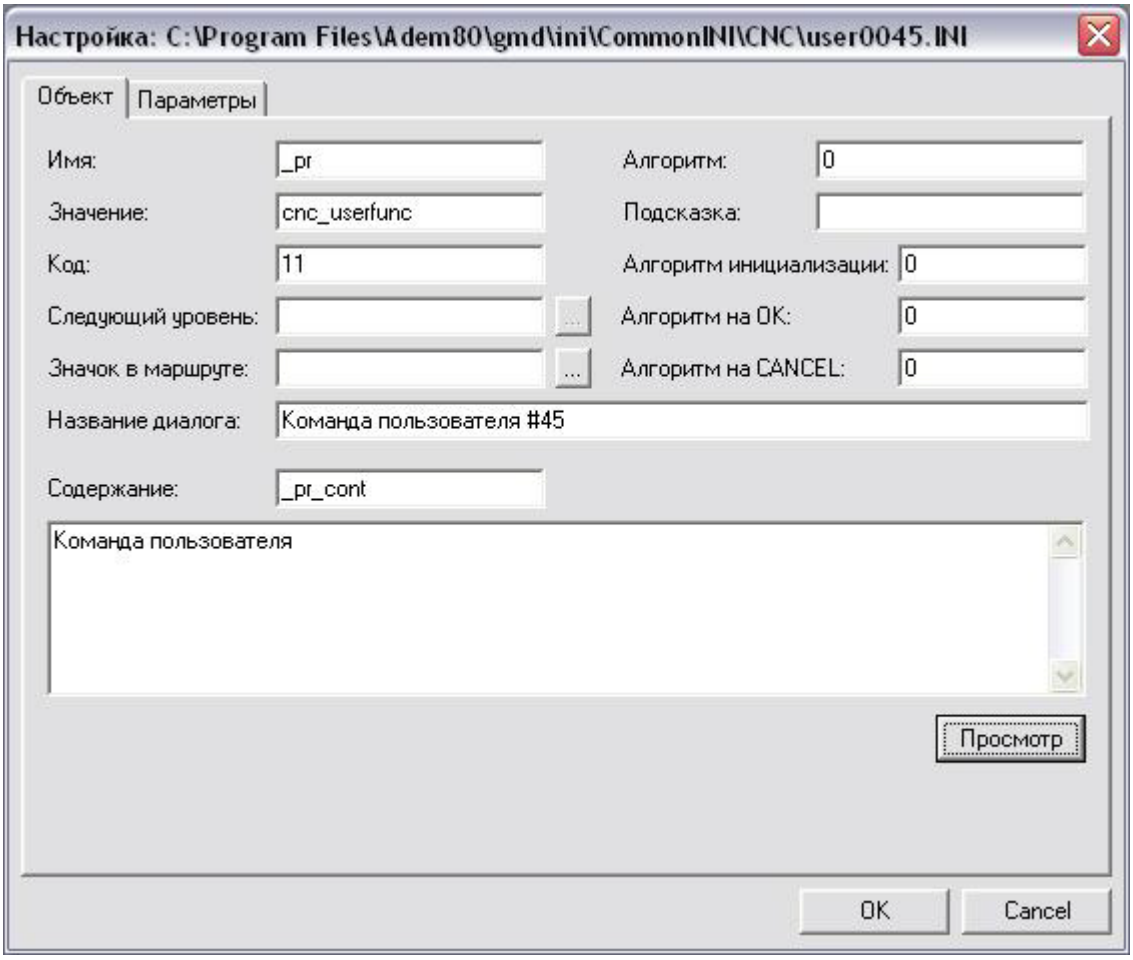
Для закрытия диалога с сохранением внесенных изменений нажмите на кнопку **«OK»**. Для закрытия диалога без сохранения внесенных изменений нажмите на кнопку **«Cancel»**.

Вкладки:

- «Объект»
- «Параметры»

Вкладка «Настройка»

Вкладка **«Объект»** диалогового окна **«Настройка»** предназначена для настройки основных параметров создаваемой пользовательской команды.



Вкладка «Объект» диалогового окна «Настройка»

Для закрытия диалога с сохранением внесенных изменений нажмите на кнопку **«OK»**. Для закрытия диалога без сохранения внесенных изменений нажмите на кнопку **«Cancel»**.

Имя

Имя переменной, которая соответствует текущему объекту. Используется в алгоритмах настройки техпроцесса.

Значение

Значение, которым инициализируется переменная, описанная параметром **«Имя»**.

Код

Код объекта. Если команда пользователя будет отрабатываться с помощью макропроцедуры, этот код должен быть равен 11. Если же команда пользователя отрабатывается в постпроцессоре, код должен иметь значение 459.

Алгоритм

Номер алгоритма. Если установлено не нулевое значение, активизируется кнопка **«Алгоритм»** на объекте в правом верхнем углу, при нажатии на которую выполняется требуемый алгоритм. Содержится алгоритм в файле с именем **0000< номер алгоритма >.alg**.

Подсказка

Текст для всплывающей подсказки кнопки **«Алгоритм»**.

Алгоритм инициализации

Номер алгоритма инициализации. Если установлено не нулевое значение, при создании объекта выполняется требуемый алгоритм, который инициализирует параметры диалога. Содержится алгоритм в файле с именем: **0000< номер алгоритма >.alg**.

Алгоритм на ОК

Номер алгоритма, выполняемого при нажатии на кнопку **"ОК"** в диалоге. Если установлено не нулевое значение, то при нажатии на кнопку **"ОК"** в диалоге выполняется требуемый алгоритм. Содержится алгоритм в файле с именем: **0000< номер алгоритма >.alg**.

Алгоритм на Cancel

Номер алгоритма, выполняемого при нажатии на кнопку **"Cancel"** в диалоге. Если установлено не нулевое значение, то при нажатии на кнопку **"Cancel"** в диалоге выполняется требуемый алгоритм. Содержится алгоритм в файле с именем: **0000< номер алгоритма >.alg**.

Название диалога

Заголовок диалогового окна команды.

Следующий уровень

Имя настроечного файла (***.ini**) или файла меню (***.mnu**). Устанавливает последовательность действий, которые будут произведены пользователем при создании объектов на следующем уровне. Если установлено имя настроечного файла, то при выполнении команды Новый на следующем уровне будет создан объект, который формируется по шаблону данного настроечного файла. Если установлено имя файла меню, то при выполнении команды **Новый** на следующем уровне откроется меню выбора, созданное на основе данного файла меню.

Содержание

Имя переменной содержания. Используется в алгоритмах.

Текст содержания

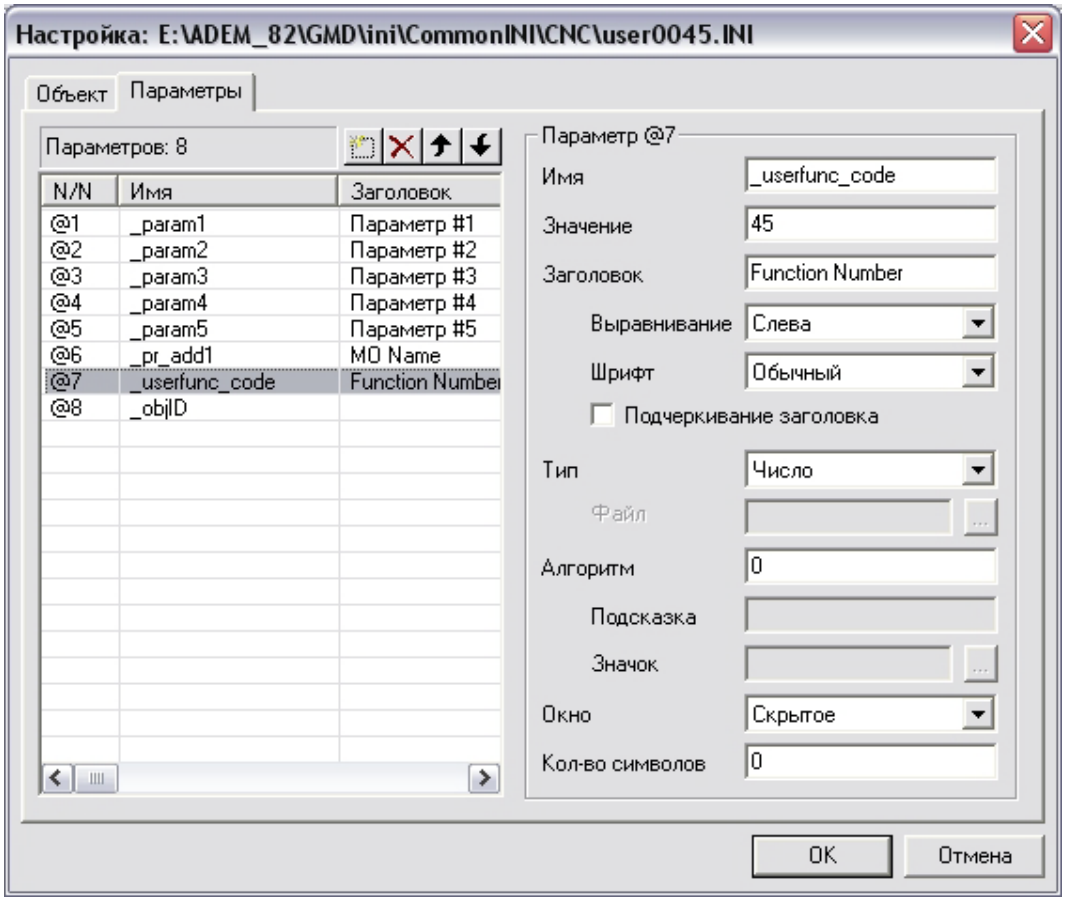
Содержание используется для описания объекта. Текст является параметрическим. Связь с параметрами объекта устанавливается с помощью специального символа **@**, после которого устанавливается либо порядковый номер параметра, либо имя параметра, заключенное в **[ ]** скобки.

Просмотр

Используется для предварительного просмотра полученного диалога объекта.

Вкладка «Параметры»

На вкладке **«Параметры»** осуществляется работа со свойствами параметров создаваемой пользовательской команды.



Вкладка «Параметры» диалогового окна «Настройка»

В левой части диалогового окна расположен список параметров пользовательской команды. Для управления списком предусмотрены кнопки:

Кнопки


Добавить
Добавление в список нового параметра. Параметр добавляется в конец списка.


Удалить
Удаление выбранного параметра из списка.


Вверх
Перемещение выбранного параметра вверх по списку.


Вниз
Перемещение выбранного параметра вниз по списку.

У каждого параметра есть свой набор свойств, которые пользователь может изменять.

Имя
Имя переменной текущего параметра. Используется в алгоритмах.

Значение
Значение, которым инициализируется переменная параметра.

Заголовок
Заголовок параметра.

Выравнивание
Выравнивание заголовка. Выравнивание может производиться:

- по левому краю
- по правому краю
- по центру

Шрифт
Начертание шрифта.

Тип
Тип параметра. Может принимать одно из следующих значений:

- **Число** — в качестве значения параметра могут использоваться математические выражения. Результат выполнения будет занесен в переменную параметра.
- **Дата** — отображается стандартный управляющий элемент диалога для ввода даты.
- **Меню** и **меню2** — комбинированный список, элементами которого являются строки текстового файла. Если определен тип **«меню»**, то после выбора строки в переменную параметра занесется число, соответствующее порядковому номеру строки в файле. Если определен тип **«меню2»**, то — сама строка.

- **Вкладка** — добавляет вкладку в диалог. В поле **«Заголовок»** необходимо ввести название вкладки. На вкладку помещаются все параметры находящиеся между двумя параметрами **«Вкладка»** или все параметры от текущего параметра **«Вкладка»** до конца. Если объект не имеет вкладок, то все элементы помещаются на вкладку **«Параметры»**.

- **Разделитель** — добавляет разделитель в диалог. В поле **«Заголовок»** необходимо ввести заголовок разделителя.

- **Только заголовок** — добавляет параметр в диалог соответствующего типа, т.е. без возможности ввода в него информации. В поле **«Заголовок»** необходимо ввести заголовок параметра. Поле для его вывода складывается из поля для вывода заголовка и поля для ввода информации.

- **Флажок** — добавляет параметр в диалог соответствующего типа. В поле **«Заголовок»** необходимо ввести заголовок флажка.

Файл

Имя текстового файла, связанного с данным параметром, тип которого установлен как **«меню»** или **«меню2»**.

Алгоритм

Номер алгоритма. Если установлено не нулевое значение, справа от параметра появится кнопка, при нажатии на которую выполняется требуемый алгоритм. Содержится алгоритм в файле с именем: **0000< номер алгоритма >.alg**.

Подсказка

Текст всплывающей подсказки для кнопки с выполнением алгоритма.

Значок

Имя файла с графическим изображением, которое будет размещено на кнопке с выполнением алгоритма.

Окно

Тип окна вывода параметра. Может принимать одно из 5-и значений: **обычное**, **большое**, **скрытое**, **обычное только чтение**, **большое только чтение**. Тип окна **«большое»** могут принимать только нечетные параметры: 1-й, 3-й… Если установлен тип **«скрытое»**, параметр не будет отображаться в диалоге и не может корректироваться пользователем. Параметры, имеющие тип окна **«обычное только чтение»** и **«большое только чтение»** могут изменять свои значения только из алгоритмов, в режиме редактирования их значения изменить нельзя. Данный вид параметра диалога распространяется только на тип данных **«число»**, **«строка»** и **«меню2»**. На другие типы данных параметр окна **«только чтение»** игнорируется.

Количество символов

Максимальное количество символов, которое возможно будет ввести в создаваемый параметр. Если значение в поле не определено, то количество символов, которое можно ввести в параметр, не ограничено.

Создание пользовательского цикла обработки

Этот раздел документации находится в разработке. По всем вопросам обращайтесь к производителю.

Добавление пользовательских параметров к основным параметрам технологического перехода

Механизм добавления пользовательских параметров к основным параметрам технологических переходов и команд аналогичен механизму создания пользовательских команд, описанному в разделе «Создание пользовательской команды».

Для того, чтобы добавить пользовательские параметры к диалогу соответствующего перехода или команды, необходимо изменить соответствующий ini-файл. По умолчанию эти файлы располагаются в директории **...ADEM/ncm/NCALG/INI/** . Для того, чтобы вкладка **«Параметры пользователя»** появилась на соответствующем диалоге, необходимо нужные Вам ini-файлы перенести в директорий **...ADEM/GMD/INI/CommonINI/CNC/** .

Перечень ini-файлов, содержащих параметры закладки **«Параметры пользователя»** для диалогов соответствующих технологических переходов:

- **milluser.ini** — закладка «Параметры пользователя» во всех технологических переходах **«Фрезеровать»**.
- **centuser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Центровать»**.
- **drilluser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Сверлить»**.
- **enlarguser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Зенкеровать»**.
- **reamuser.ini**— закладка «Параметры пользователя» в технологическом переходе **«Развернуть»**.
- **boreuser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Расточить (фрезерный)»**.
- **tapuser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Нарезать резьбу (фрезерный)»**.
- **turnuser.ini** — закладка «Параметры пользователя» в технологическом переходе "**Точить (токарный)»**.
- **lboreuser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Расточить (токарный)»**.

- **lfaceuser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Подрезать (токарный)»**.
- **lcutuser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Отрезать (токарный)»**.
- **lthreaduser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Нарезать резьбу (токарный)»**.
- **cutuser.ini** — закладка «Параметры пользователя» в технологическом переходе **«Резать»**. Этот диалог распространяется на электроэрозионную, лазерную и газово-плазменную резку.
- **punchuser.in i**— закладка «Параметры пользователя» в технологическом переходе **«Пробить (пресс)»**.

Перечень ini-файлов, содержащих параметры закладки **«Параметры пользователя»** для диалогов соответствующих технологических команд:

- **homeuser.ini** – закладка «Параметры пользователя» в технологической команде **«Начало цикла»**.
- **safeposuser.ini** – закладка «Параметры пользователя» в технологической команде **«Безопасная позиция»**.
- **clearpuser.ini** – закладка «Параметры пользователя» в технологической команде **«Плоскость холостых ходов»**.
- **tooluser.ini** – закладка «Параметры пользователя» в технологической команде **«Инструмент»**.
- **reclawuser.ini** – закладка «Параметры пользователя» в технологической команде **«Перезахват»**.
- **rotationuser.ini** – закладка «Параметры пользователя» в технологической команде **«Поворот»**.

Добавление созданных команд и циклов в меню выбора

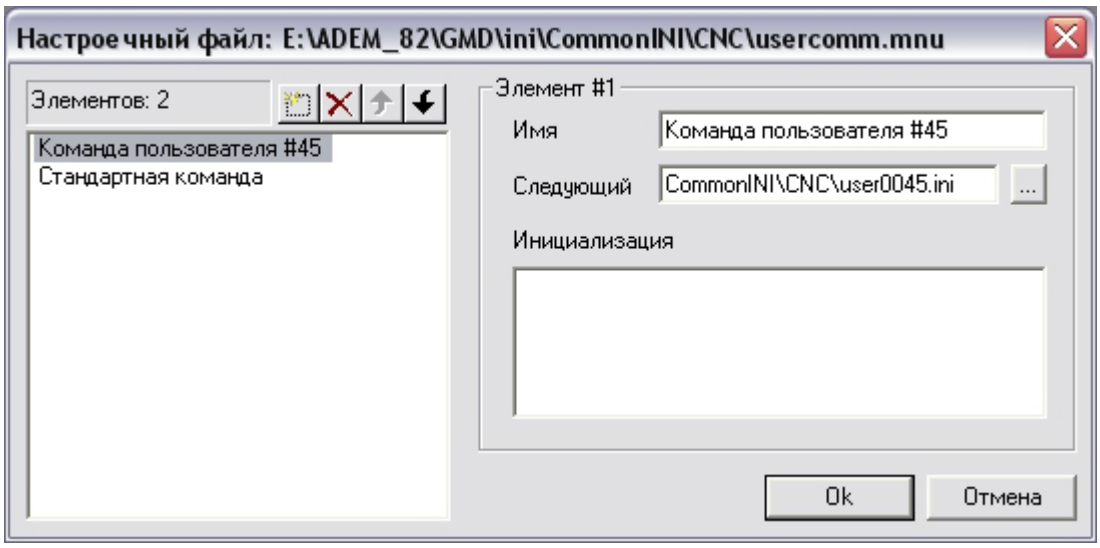
Меню выбора ТК **«Пользовательская команда»** и **«Пользовательский цикл»** формируются в соответствии с шаблоном, содержащимся в настроечном файле с расширением ***.mnu** и находятся в каталоге **...\GMD\INI\CommonINI\CNC**. Файл, содержащий меню выбора пользовательских команд, имеет имя **"usercomm.mnu"**, а файл, содержащий меню выбора пользовательских циклов — **"cycle.mnu"**

Чтобы изменить шаблон или создать новый, необходимо выполнить команду меню **«Модуль»** — **«Adem CAPP Developer»**. На вкладке **«Файлы»** раскройте папку **"MNU файлы"**, выберите требуемый **MNU**-файл и откройте его двойным щелчком мыши.

В общем виде MNU-файл представляет собой форматированный текстовый файл, каждая строчка которого является пунктом меню.

Формат строки **MNU**-файла:

```
<имя элемента меню>, <имя настроечного файла (*.ini)> или <файла меню
(*.mnu)>
[?<имя параметра 1>=<значение параметра 1>;
  <имя параметра 2>=<значение параметра 2>[;...]]
```



Диалоговое окно «Настроечный файл»

Редактирование MNU-файла осуществляется в специализированном диалоге **«Настроечный файл»**. После внесения изменения нажмите кнопку **«Ok»** в диалоге.

MNU-файла можно редактировать в явном виде, открыв его как текстовый файл. Для этого выберите требуемый файл в диалоговом окне **ADEMTDMDev** и в его контекстном меню укажите пункт **«Открыть как текст»**.

-->

Примеры

В этом разделе представлены примеры трансляции некоторых сложных команд CLData. А также приведены примеры пересчета углов и координат для различных кинематических схем и многое другое, что может понадобиться вам при проектировании постпроцессоров.

Статьи:

Пример отработки пользовательского цикла обработки

Этот раздел документации находится в разработке. По всем вопросам обращайтесь к разработчикам.

Пример отработки пользовательских параметров технологических переходов и команд

В качестве примера рассмотрим алгоритм трансляции двух пользовательских параметров из любого перехода:

- Параметр №1: числовой, его имя **_param1=10**
- Параметр №2: символьный, его имя **_param2=ADEM**

Алгоритм трансляции будет выглядеть следующим образом:

```
460  // начало алгоритма трансляции команды №460 - Параметры пользователя;
IF  _PARAM1!=0  _AAA=_PARAM1  // присваиваем переменной _AAA значение пользовательского параметра _param1;
IF  _PARAM2!=0  _BBB=_PARAM2  // присваиваем переменной _BBB значение пользовательского параметра _param2;
_PARAM1=0 // обнуление значения пользовательского параметра _param_1;
_PARAM2=0 // обнуление значения пользовательского параметра _param_2;
END;
```

После отработки алгоритма, пользовательские переменные приобретут значение соответствующих пользовательских параметров: _AAA=10; _BBB=ADEM.

Примечание

После инициализации пользовательских переменных значениями пользовательских параметров, лучше всего эти параметры обнулить, так как пользовательские параметры с одинаковыми именами могут присутствовать в нескольких переходах и командах. А при инициализации необходимо проводить проверку на нулевое значение.

Примеры отработки команды "Цикл" (код 36)

Отработка стандартного сверлильно-расточного цикла (а также и пользовательского) в CLData представляется в виде команды **«Цикл»** (код 36), стоящей сразу после описания точки, в которой этот цикл выполняется. Как правило, в УП цикл описывается с помощью какой-либо подготовительной функции (например, G81), а после описания вызывается в каждой точке отработки цикла автоматически до отмены (например, по G80).

Описание цикла, как правило, содержится в 36 алгоритме, либо в алгоритмах, которые вызываются из него. Вывод в УП точек, на которых отрабатывается цикл, тоже лучше делать в это алгоритме. Отмену отработки цикла обычно прописывают в алгоритме №181 по коду отвода.

Так может выглядеть часть алгоритма №181:

```
181;
IF C/GOHOME=ON DO // если идёт отвод инструмента, то есть цикл, если он
был, уже отработан на всех точках;
  IF _FLCYLCL=1 DO // если цикл обрывается;
    _G->80 // вывод в УП функции отмены цикла;
    _FLCYCL=0 // обнуление переменной;
  ENDDO;
IF NEXT/COD=36 _FLCYCL=1 GOTO 1 // переход на метку если идёт обработка
цикла в точке или следующей командой в CLData будет команда «Цикл»;
...
...
1; _AAA=1;
END;
```

Так может выглядеть алгоритм №36:

```
36;
IF _FLCYCL=0 DO // описание цикла, делается, как правило, только один
раз - при первом вызове;
  BLOCK;
  // так как вариантов описания огромное множество, само описание мы не
приводим;
  BLOCK;
  ENDDO
// далее происходит вывод в УП координат текущей точки, на которой будет
отрабатываться цикл, а также выводятся параметры цикла, которые должны
присутствовать на каждой точке отработки (например, параметр R);
_R->PAR/C10;
_X->XT;
```

```
_Y->YT;
_Z->ZT;
BLOCK;
_FLCYCL=1;
END;
```

Примечание

Для удобства последующего сопровождения постпроцессора, советуем Вам в алгоритме №36 анализировать цикл только по его номеру, а само описание каждого цикла формировать в отдельных алгоритмах.

Пример подсчета времени работы УП

Подсчитывать время работы УП можно разными способами. В этом разделе представлен один из них, подходящий к большинству постпроцессоров.

Сам подсчет времени содержится в одном алгоритме, но, для его корректной работы, вызов этого алгоритма необходимо вставить в алгоритмы трансляции следующих команд: 1, 4, 23, 24, 25, 27, 35, 36, 40, 94, 181, 183, 223, 252, 713.

Примечание

Если нарезание резьбы резцом (код 94) осуществляется с помощью универсального станочного цикла (например, G76), то расчет времени его отработки обычно подсчитывают отдельно, ориентируясь по диаметру резьбы, количеству проходов и т.д.

Алгоритм подсчета времени будет выглядеть следующим образом:

```
5000;
__CHP=ACOS(-1);
__CLD=FR[1] // установка кода танслируемой команды;
IF C/GOHOME=ON __CLD=181;
IF __CLD=1 DO // инициализация переменных;
  AUTOTIME=OFF;
  ___TIME=0;
  ___FASTFD=3000 // максимальная подача для ускоренных перемещений;
  __CHTOOL=5:60 // время замены инструмента, мин;
  __SPINTM=2:60 // время разгона шпинделя, мин;
  __SPA=90:180*__CHP // подача для поворота вокруг оси A;
  __SPB=90:180*__CHP // подача для поворота вокруг оси B;
  __SPB=90:180*__CHP // подача для поворота вокруг оси C;
  SEPSUB __QTYSUB __SNAME[] __SBEG[] __SEND[] // установка массивов для
работы с подпрограммами;
  FOR __II=1 __II<=__QTYSUB __II=+II+1;
    __SCALL[__II]=0;
    __STIME[__II]=0;
    ENDDO;
  ENDDO;
ELSE IF __CLD=4 DO // конец УП;
  TIME=_TIME;
  ENDDO;
```

```
ELSE IF __CLD=23 DO // включение подачи;
  __PRFEED=1;
  ENDDO;
ELSE IF __CLD=24 DO // включение шпинделя;
  __TIME=__TIME+__SPINTM;
  ENDDO;
ELSE IF __CLD=25 DO // включение ускоренного хода;
  _PRFEED=0;
  ENDDO;
ELSE IF __CLD=27 DO // выстой
  __IF FR[3]=0 __TIME=__TIME+FR[4]:60 // если выстой задан в секундах;
  ELSE __TIME=__TIME+FR[4]*N // если выстой задан в оборотах;
  ENDDO;
ELSE IF __CLD=35 DO // загрузка инструмента;
  __TIME=__TIME+__CHTOOL;
  ENDDO;
ELSE IF __CLD=36 DO // стандартный сверлильно-расточной цикл;
  IF  N=0  N=(1000*V):(__CHP*RAD/TOOL*2)  // расчёт N, если была задана
скорость резания;
  IF FR[28]!=1 __FEEDC=FR[10]*N // расчёт подачи в мм/мин, если она была
задана в мм/об;
  ELSE __FEEDC=FR[10];
  __TIMER=FR[12]:__FEEDC // время рабочего хода;
  __TIMEH=FR[12]:(__FEEDC*5) // время ускоренного хода;
  __DEPTH1=FR[17] // текущая глубина прохода;
  FOR __II=1 __II<=FR[14] __II=__II+1;
    __TIMEH=__TIMEH+(__DEPTH1-1):(__FEEDC*5);
    IF __DEPTH!=0 DO;
      IF (__DEPTH1+__DEPTH)>FR[12] __DEPTH=FR[12]-__DEPTH1;
      __TIMER=__TIMER+(__DEPTH+1):__FEEDC;
      __DEPTH1=__DEPTH1+__DEPTH;
    ENDDO;
    ELSE DO;
      IF (__DEPTH1+FR[17])>FR[12] FR[17]=FR[12]-__DEPTH1;
      __TIMER=__TIMER+(FR[17]+1):__FEEDC;
      __DEPTH1=__DEPTH1+FR[17];
    ENDDO;
  __TIMEH=__TIMEH+__DEPTH1:(__FEEDC*5);
  __ENDDO;
  __TIME=__TIME+__TIMER+__TIMEH;
  ENDDO;
ELSE IF __CLD=40 DO // поворот;
  __TIME=__TIME+ABS(FR[5]*__SPA)+ABS(FR[7]*__SPB)+ABS(FR[9]*__SPC);
  ENDDO;
ELSE IF __CLD=94 DO // нарезание резьбы резцом;
  IF FR[11]!=0 __TIME=__TIME+ABS(FR[12]):(FR[11]*N);
  ENDDO;
ELSE IF __CLD=181|__CLD=182|__CLD=189 DO // любые линейные перемещения,
кроме дополнительных;
  __FD=__PRFEED;
  __LENGM=LENG/MOV;
  IF __LENGM>=1000 __LENGM=1000;
  IF __FD=0 __TIME=__TIME+__LENGM:__FASTFD;
  ELSE IF S!=0 __TIME=__TIME+__LENGM:S;
  ENDDO;
```

```
ELSE IF __CLD=183|__CLD=184 DO // дуговые перемещения, кроме
дополнительных;
  __LENGM=ABS(FR[8]*ANG/CIRC);
  IF S!=0 __TIME=__TIME+__LENGM:S;
  ENDDO;
ELSE IF __CLD=223 DO // вызов подпрограммы;
  FOR __II=1 __II<=__QTYSUB __II=__II+1;
    IF ИВППГ=__SNAME[__II] __SCALL[__II]=__SCALL[__II]+КПОВТППГ;
    ENDDO;
  ENDDO;
ELSE IF __CLD=252|__CLD=713 DO // начало или конец подпрограммы;
  IF FR[0]!=1 __TIME=0 // начало формирования подпрограммы;
  ELSE DO // конец формирования подпрограммы;
    FOR __II=1 __II<=__QTYSUB __II=__II+1;
      IF ИФППГ=__SNAME[__II] __STIME[__II]=__TIME;
      ENDDO;
    IF IFR=__SEND[__QTYSUB] DO;
      FOR __JJ=1 __JJ<=__QTYSUB __JJ=__JJ+1;
        TIME=TIME+__STIME[__JJ]*__SCALL[__JJ];
        ENDDO;
      ENDDO;
    ENDDO;
  ENDDO;
END;
```

Примечание

Имена всех пользовательских переменных в этом алгоритме начинаются с двойного подчеркивания "__" , чтобы не пересечься с именами переменных, назначаемых в остальных алгоритмах постпроцессора.

Пример формирования цикла нарезания резьбы резцом (G76)

Нарезание резьбы резцом в CLData представляется в виде отработки команды нарезания резьбы (код 94) на каждом проходе. Если в УП программе этот вид обработки должен быть представлен в виде одного универсального цикла, то вывод цикла в УП нужно формировать в отдельном пользовательском алгоритме. А алгоритм трансляции №94 можно использовать для предварительного сбора какой-либо информации.

В приведенном ниже примере, формирование самого цикла нарезания резьбы происходит в алгоритме №9400. Причем, этот алгоритм, в свою очередь, вызывается из алгоритма трансляции №181 после того, как все резьба уже будет получена.

Так может выглядеть часть алгоритма №181:

```
181;
IF C/GOHOME=ON DO // если идет отвод инструмента, то есть резьба, если
была, сформировалась;
  IF _FLREZB=1 CALL 9400 // вызов алгоритма формирования цикла, если
формировалась резьба;
  ENDDO;
ELSE IF _FLREZB=1 DO // если еще формируется резьба;
```

```
  GOTO 1 // переход на метку, чтобы не отрабатывать остальную часть
алгоритма, например, вывод координат в кадр УП;
  ENDDO;
  ...
  ...
1: _AAA=1;
END;
```

Так может выглядеть алгоритм №94:

```
94;
IF FR[9]>1&FR[10]!=1 GOTO 1 // переход на метку, если это не первый
резьбовой заход;
IF _FLREZB=0 DO // самый первый проход;
  _YREZN=YT // запоминаем координату Y первого прохода;
  _XKREZ=XNEXT // запоминаем координату X конца резьбы;
  ENDDO;
IF _YREZP!=YT DO // черновой проход;
  _YREZP1=_YREZP // координата Y предыдущего чернового прохода;
  _YREZP=YT // координата Y очередного чернового прохода;
  IF _PRDN=0&_FLREZB=1 DO // анализ второго чернового прохода;
    IF _VIDR=0 DO // резьба наружная;
      _PRIPN=_YREZN-_YREZP // припуск на первом проходе;
      _YNACH=YT+_PRIPN*2+0.012 // координата Y начала резьбы;
      ENDDO;
    ELSE DO // резьба внутренняя;
      _PRIPN=_YREZP-_YREZN // припуск на первом проходе;
      _YNACH=YT-_PRIPN*2-0.012 // координата Y начала резьбы;
      ENDDO;
    _PRDN=1;
    ENDDO;
  _KCHP1=_KCHP1+1 // количество черновых проходов плюс один;
  ENDDO;
ELSE DO // чистовой проход;
  _YREZK=YT // координата Y чистового прохода;
  _KCHP=_KCHP+1 // количество чистовых проходов минус один;
  ENDDO;
_SHAGR=PITCH/T // шаг резьбы;
_FLREZB=1;
_VIDR=FR[4] // вид резьбы;
_LENGTH=FR[12] // длина резьбы, при конической резьбе длина по оси X;
_KZH=FR[9] // количество заходов резьбы;
1: _AAA=1;
END;
```

А так выглядит алгоритм №9400, в котором формируется и выводится в УП цикл нарезания резьбы:

```
9400;
_HIGHT=0.65*_SHAGR // высота резьбы может быть подсчитана исходя из
координаты Y начала резьбы и координаты Y чистового прохода, но в
большинстве устройств ЧПУ для вычисления высоты резьбы используется
формула: h=0.65 * шаг резьбы;
IF _VIDR=0 DO // резьба наружная;
  _PRIPK=_YREZP1-_YREZK // припуск на последнем проходе;
  _DEND=(_YNACH-_HIGHT)*2 // диаметр впадин;
```

```
  ENDDO;
ELSE DO // резьба внутренняя ;
  _PRIPK=_YREZK-_YREZP1 // припуск на последнем проходе;
  _DEND=(_YNACH+_HIGHT)*2 // диаметр впадин;
  ENDDO;
// дальше идет формирование цикла и вывод в УП;
IF  _KCHP+1<10  _KCHP='0@[_kchp+1]'  // перевод из числа в символы
количества чистовых проходов, если их меньше 10;
ELSE _KCHP=_KCHP+1;
IF PR/THRD=METRIC _UREZB=60 // определение угла резьбы по ее типу;
ELSE IF PR/THRD=PIPE _UREZB=55;
ELSE IF PR/THRD=SQUARE _UREZB=90;
ELSE IF PR/THRD=TRAPEZ _UREZB=80;
ELSE _UREZB=0;
_RAZN=ABS(ATAN(ANG/THRD)*_LENGTH)  // вычисление катета, если резьба
коническая;
BLOCK;
_TXT1->'G76  P@[_KCHP]00@[_UREZB]  Q@[DEV(_PRIPK*10000*2)]  R@[_PRIPK*2]'
// вывод в УП первой строки цикла;
BLOCK;
_PRIPN=_HIGHT:SQRT(_KCHP1-1)  // мы в алгоритме №94 уже вычислили
начальный припуск, но в большинстве устройств ЧПУ для вычисления высоты
резьбы используется эта формула;
IF    ANG/THRD!=0    _TXT1->'G76    X@[_DEND]    Z@[_XKREZ]    R@[_RAZN]
P@[dev(_HIGHT*10000)]  Q@[DEV(_PRIPN*10000)]  F@[_SHAGR]'  // вывод в УП
второй строки цикла, если резьба коническая;
ELSE    _TXT1->'G76    X@[_DEND]    Z@[_XKREZ]    P@[DEV(_HIGHT*10000)]
Q@[DEV(_PRIPN*10000)]  F@[_SHAGR]'  // вывод в УП второй строки цикла,
если резьба цилиндрическая;
BLOCK;
_KCHP1=0 // обнуление количества черновых проходов;
_KCHP=0 // обнуление количества чистовых проходов;
_FLREZB=0 // обнуление переменной, которая показывает была ли резьба;
// далее идет расчет времени отработки резьбового цикла;
_TIME1=_LENGTH*(_KCHP1-1):(_SHAGR*N)   //   время   отработки   черновых
проходов;
_TIME2=_LENGTH*(_KCHP+1):(_SHAGR*N)   //   время   отработки   чистовых
проходов;
_TIME3=_LENGTH*(_KCHP+_KCHP1):_FFEED // время отработки холостых ходов,
где _FFEED - величина ускоренных перемещений;
_TIMEO=(_TIME1+_TIME2+_TIME3)*_KZH  // общее время отработки цикла в
зависимости от количества заходов резьбы;
END;
```

Примеры работы с трансформами

При трансляции команды **«Трансформ»** (код 10123), обычно устанавливают только признак того, что эта команда была. А в алгоритмы №35 и №181 вставляют проверку на этот признак и вызов алгоритма, в котором и описаны все необходимые действия.

В зависимости от математического аппарата устройства ЧПУ и кинематической схемы станка, в этом алгоритме рассчитывают углы поворота вращающихся осей, а также при необходимости производят пересчет линейных координат.

Примеры трансляции 5x обработки

Этот раздел документации находится в разработке. По всем вопросам обращайтесь к разработчикам.

Приложения

В этом разделе приведен список основных команд **CLData**, а также структура некоторых из них, для организации прямого доступа к параметрам команд.

Приложения:

Список основных транслируемых команд CLData

Код	Наименование
1	Программа
2	Деталь
3	Станок
4	Конец программы
6	Начало
22	Стоп программы
23	Включить рабочую подачу
24	Включить шпиндель
25	Включить ускоренную подачу
26	Включить СОЖ
27	Выстой
28	Отвести инструмент. Зарезервированная команда (её алгоритм никогда не отрабатывается)
29	Перезахват (листоштамповка)
33	Условный останов программы
34	Смена стола
35	Загрузка инструмента
36	Цикл
39	Коррекция инструмента. Зарезервированная команда (её алгоритм никогда не отрабатывается)
40	Поворот
41	Совмещенные мультипермещения
45	Контрольная точка
94	Резьба токарная
181	Линейная интерполяция
182	Линейное векторное перемещение (отрабатывается всегда как 181 команда)
183	Круговая интерполяция
184	Векторное круговое перемещение
185	Совмещённое линейное перемещение (дополнительное линейное перемещение)
187	Совмещённое перемещение по дуге (дополнительно круговое перемещение)
189	Векторное линейное перемещение с коррекцией
190	Криволинейное перемещение (основное)
191	Криволинейное перемещение (дополнительное)

223	Вызов подпрограммы
227	Строка
252	Начало подпрограммы
301	Технологический переход «Фрезеровать»
302	Технологический переход «Точить»
303	Технологический переход «Расточить (токарный)»
304	Технологический переход «Нарезать резьбу метчиком»
305	Технологический переход «Отрезать»
306	Технологический переход «Пробить»
307	Технологический переход «Центровать»
308	Технологический переход «Сверлить»
309	Технологический переход «Развернуть»
310	Технологический переход «Подрезать»
311	Технологический переход «Зенкеровать»
312	Технологический переход «Гравировать»
314	Технологический переход «Расточить (сверлильный)»
315	Технологический переход «Резать»
316	Технологический переход «Нарезать резьбу резцом (токарный)»
401	Начало цикла
402	Трансформ зоны
404	Ноль УП для текущей зоны обработки
405	Трансформ осей поворота
406	Начальная точка обработки
451	Безопасная позиция
452	Плоскость холостого хода
458	Вставка УП
459	Команда пользователя
460	Параметры пользователя (параметры, расположенные на вкладке «Параметры пользователя» в технологическом объекте)
491	Технологическая команда «Величина аппроксимации»
493	Турета
582	Комментарий программы
700	Выключить СОЖ
701	Выключить шпиндель
703	Включить корректор по оси X
704	Включить корректор по оси Y
705	Включить корректор по оси Z
706	Включить радиусный корректор
707	Выключить корректор по оси X
708	Выключить корректор по оси Y
709	Выключить корректор по оси Z
710	Выключить радиусный корректор
711	Ориентированный останов шпинделя
713	Конец подпрограммы
799	Зарезервированная команда (её алгоритм никогда не отрабатывается)
900	Глубина резания (только для токарной обработки)
901	Плоскость интерполяции (для всех видов обработки кроме токарной)
10123	Трансформ

Структура основных транслируемых команд CLData

Список команд:

- Параметры пользователя - закладка «Параметры пользователя» в диалоге тех.перехода или команды (код 460)
- Технологическая команда «Величина аппроксимации» (код 491)
- Турета (код 493)
- Комментарий (код 582)
- Глубина резания (код 900)
- Плоскость интерполяции (код 901)
- Трансформ (код 10123)
- Технологический переход «Фрезеровать» (код 301)

Структура команды «Учетные данные программы» (код 1)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=1
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	имя программы
FR[4]	Ф.И.О. разработчика
FR[5]	дата
FR[6]	цех
FR[7]	участок
FR[8]	номер операции

Структура команды «Учетные данные детали» (код 2)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=2
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код для адаптера
FR[4]	наименование детали
FR[5]	наименование материала детали
FR[6]	обозначение детали

Структура команды «Учетные данные станка» (код 3)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=3
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код станка: 0 — токарный 1 — обрабатывающий центр 2 — фрезерный 3 — сверлильный 4 — пресс 5 — фрезерный 2.5X 6 — эррозионный 7 — эррозионный по двум контурам

	8 — эррозионный с векторами нормали
FR[4]	количество параметров (от 0 до 10)
FR[5]	наименование станка
FR[6]	массив, содержащий пользовательские параметры. Доступ к параметрам осуществляется через FR[i+100], где i — порядковый номер параметра

Структура команды «Конец УП» (код 4)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=4
FR[2]	порядковый номер объекта в маршруте обработки

Структура команды «Стоп» (код 22)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=22
FR[2]	порядковый номер объекта в маршруте обработки

Структура команды «Включение рабочей подачи» (код 23)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=23
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код задания единиц измерения подачи: 1 — подача в мм/мин 2 — подача в мм/об
FR[4]	порядковый номер объекта в маршруте обработки

Структура команды «Включение шпинделя» (код 24)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=24
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код шпинделя: 0 — шпиндель включен 1 — шпиндель выключен 2 — задана частота вращения шпинделя 3 — задана постоянная скорость резания 4 — задан ориентированный останов шпинделя
FR[4]	величина скорости резания или частоты вращения шпинделя
FR[5]	угол ориентации шпинделя
FR[6]	код диапазона шпинделя
FR[7]	номер заданного механического диапазона шпинделя
FR[8]	код DXYZ: 0 — не задан

	1 — задан D 2 — задан X 3 — задан Y 4 — задан Z
FR[9]	величина DXYZ
FR[10]	код задания способа ограничения вращения шпинделя 0 — ограничение не задано 1 — задано максимальная частота вращения 2 — задана максимальная скорость резания
FR[11]	заданная величина ограничения вращения шпинделя

Структура команды «Включение холостого хода» (код 25)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=25
FR[2]	порядковый номер объекта в маршруте обработки

Структура команды «Включение СОЖ» (код 26)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=26
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код включения СОЖ: 0 — подача СОЖ выключена 1 — подача СОЖ включена
FR[4]	номер трубопровода

Структура команды «Выстой» (код 27)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=27
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код способа задания выстоя: 0 — выстой задан в секундах 1 — выстой задан в оборотах
FR[4]	величина выстоя

Структура команды «Перехват» (код 29)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=29
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	координата X перехвата
FR[4]	координата Y перехвата
FR[5]	координата Z перехвата

Структура команды «Условный останов» (код 33)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=33
FR[2]	порядковый номер объекта в маршруте обработки

Структура команды «Смена стола» (код 34)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=34
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	номер стола

Структура команды «Загрузка инструмента» (код 35)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=35
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код загружаемого инструмента: 0 — инструмент не задан 1 — фреза 2 — сверло 3 — центровка 4 — зенкер 5 — развёртка 6 — метчик 7 — резец 8 — пуансон 9 — проволока 10 — лазер 11 — резак
FR[5]	код задания корректора по оси X: 0 — не задан 1 — задан
FR[6]	номер корректора по оси X
FR[7]	код задания корректора по оси Y:

	0 — не задан 1 — задан
FR[8]	номер корректора по оси Y
FR[9]	код задания корректора по оси Z: 0 — не задан 1 — задан
FR[10]	номер корректора по оси Z
FR[11]	код задания радиусного корректора: 0 — не задан 1 — задан
FR[12]	номер радиусного корректора
FR[13]	код задания вылета по оси X: 0 — не задан 1 — задан
FR[14]	значение вылета по оси X
FR[15]	код задания вылета по оси Y: 0 — не задан 1 — задан
FR[16]	значение вылета по оси Y
FR[17]	код задания вылета по оси Z: 0 — не задан 1 — задан
FR[18]	значение вылета по оси Z
FR[19]	код задания геометрии инструмента: 1 — геометрия задана через радиус 2 — геометриязадана через диаметр 3 — геометрия задана через сечение 4 — геометрия задана через ширину
FR[20]	величина радиуса или диаметра или размера "A" сечения
FR[21]	ширина или величина размера "B"
FR[22]	величина радиуса скругления иснтрумента
FR[23]	длина иснтрумента
FR[25]	код ориентации инструмента: 0 — 0^0 1 — 45^0 2 — 90^0 3 — 135^0 4 — 180^0 5 — 225^0 6 — 270^0 7 — 315^0

FR[26]	код задания пользовательского инструмента: 0 — не задан 1 — задан
FR[27]	адрес контура пластинки
FR[28]	адрес контура резцедержателя
FR[29]	имя каталожного файла инструмента
FR[30]	угол инструмента
FR[31]	подтип инструмента: 101 — фреза концевая 102 — фреза концевая скруглённая 103 — фреза концевая сферическая 104 — фреза коническая 105 — фреза коническая скруглённая 106 — фреза коническая сферическая 107 — фреза угловая 108 — фреза угловая скруглённая 109 — фреза угловая сферическая 110 — фреза дисковая 111 — фреза дисковая скруглённая 112 — фреза дисковая сферическая 113 — фреза каплевидная 701 — ромбическая пластина 702 — квадратная пластина 703 — треугольная пластина 704 — прорезная пластина 705 — круглая пластина

Структура команды «Цикл» (код 36)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=36
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	количество параметров цикла
FR[4]	код подпрограммы: 0 — цикл используется в основной программе 1 — цикл используется в подпрограмме
FR[5]	имя подпрограммы, в которой используется цикл
FR[6]	номер цикла
FR[i+10]	массив, содержащий параметры цикла. Доступ к параметрам осуществляется через FR[i+10], где i — порядковый номер параметра

Структура команды «Поворот» (код 40)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=40
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код способа задания углов поворота: 0 — заданы абсолютные углы 1 — заданы относительные углы
FR[4]	код задания углов поворота относительно оси A: 0 — не задан 1 — задан
FR[5]	величина поворота вокруг оси A
FR[6]	код задания углов поворота относительно оси B: 0 — не задан 1 — задан
FR[7]	величина поворота вокруг оси B
FR[8]	код задания углов поворота относительно оси C: 0 — не задан 1 — задан
FR[9]	величина поворота вокруг оси C

Структура команды «Мультиперемещения» (код 41)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=41
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код способа задания углов поворота: 0 — заданы абсолютные углы 1 — заданы относительные углы
FR[4]	величина координаты X
FR[5]	величина координаты Y
FR[6]	величина координаты Z
FR7]	код задания углов поворота относительно оси A: 0 — не задан 1 — задан
FR[8]	величина поворота вокруг оси A
FR[9]	код задания углов поворота относительно оси B: 0 — не задан 1 — задан
FR[10]	величина поворота вокруг оси B
FR[11]	код задания углов поворота относительно оси C:

	0 — не задан 1 — задан
FR[12]	величина поворота вокруг оси C

Структура команды «Контрольная точка» (код 45)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=45
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	номер контрольной точки

Структура команды «Резьба токарная» (код 94)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=94
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	профиль резьбы: 0 — метрическая 1 — трапецеидальная 2 — упорная 3 — трубная 4 — прямоугольная
FR[4]	вид резьбы: 0 — наружная 1 — внутрення
FR[5]	сбег: 0 — сбег не задан 1 — сбег задан
FR[6]	тип резьбы: 0 — цилиндрическая 1 — коническая
FR[7]	угол резьбы. Если тип резьбы — «Цилиндрическая», то угол равен 0
FR[8]	направление вращения шпинделя: 0 — по часовой стрелке 1 — против часовой стрелки
FR[9]	угол резьбы. Если тип резьбы — «Цилиндрическая», то угол равен 0
FR[10]	количество заходов резьбы
FR[11]	шаг резьбы
FR[12]	длина резьбы. При конической резьбе — длина по оси X
FR[13]	количество резьбовых проходов

Структура команды «Линейные перемещения» (код 181)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=181
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	текущая координата X
FR[4]	текущая координата Y
FR[5]	текущая координата Z

Структура команды «Круговые перемещения» (код 183)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=183
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	координата X конечной точки дуги
FR[4]	координата Y конечной точки дуги
FR[5]	координата Z конечной точки дуги
FR[6]	координата X центра дуги
FR[7]	координата Y центра дуги
FR[8]	радиус дуги. Если движение происходит по часовой стрелке, то радиус дуги <0, если движение против часовой стрелки — радиус дуги >0
FR[9]	код плоскости расположения окружности: 0 — текущая плоскость 1 — плоскость XY 2 — плоскость YZ 3 — плоскость ZX

Структура команды «Векторные круговые перемещения» (код 184)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=184
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	координата X конечной точки дуги
FR[4]	координата Y конечной точки дуги
FR[5]	координата Z конечной точки дуги
FR[6]	координата X центра дуги
FR[7]	координата Y центра дуги
FR[8]	радиус дуги. Если движение происходит по часовой стрелке, то радиус дуги <0, если движение против часовой стрелки — радиус дуги >0
FR[9]	угол раствора дуги окружности
FR[10]	единичный вектор оси инструмента к оси X в конечной точке
FR[11]	единичный вектор оси инструмента к оси Y в конечной точке
FR[12]	единичный вектор оси инструмента к оси Z в конечной точке
FR[13]	код плоскости расположения окружности:

0 — текущая плоскость 1 — плоскость XY 2 — плоскость YZ 3 — плоскость ZX

Структура команды «Дополнительное линейное перемещение» (код 185)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=185
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	текущая координата X дополнительного перемещения
FR[4]	текущая координата Y дополнительного перемещения
FR[5]	текущая координата Z дополнительного перемещения

Структура команды «Дополнительное круговое перемещение» (код 187)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=187
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	координата X конечной точки дуги
FR[4]	координата Y конечной точки дуги
FR[5]	координата Z конечной точки дуги
FR[6]	координата X центра дуги
FR[7]	координата Y центра дуги
FR[8]	радиус дуги. Если движение происходит по часовой стрелке, то радиус дуги <0, если движение против часовой стрелки — радиус дуги >0
FR[9]	код плоскости расположения окружности: 0 — текущая плоскость 1 — плоскость XY 2 — плоскость YZ 3 — плоскость ZX

Структура команды «Векторные линейные перемещения с коррекцией» (код 189)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=189
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	текущая координата X
FR[4]	текущая координата Y
FR[5]	текущая координата Z
FR[6]	направляющий косинус единичного вектора оси инструмента к оси X
FR[7]	направляющий косинус единичного вектора оси инструмента к оси Y
FR[8]	направляющий косинус единичного вектора оси инструмента к оси Z
FR[9]	направляющий косинус единичного вектора в точке касания к оси X
FR[10]	направляющий косинус единичного вектора в точке касания к оси Y
FR[11]	направляющий косинус единичного вектора в точке касания к оси Z

Структура команды «Криволинейное перемещение» (код 190)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=190
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код кривой: 3 — задана пространственная дуга 4 — задан кубический полином 7 — задан NURBS-сплайн
FR[4]	координата X конечной точки движения по кривой
FR[5]	координата Y конечной точки движения по кривой
FR[6]	координата Z конечной точки движения по кривой
	Если FR[3]=3 (пространственная дуга)
FR[7]	координта X центра окружности
FR[8]	координта Y центра окружности
FR[9]	координта Z центра окружности
FR[10]	координта Z центра окружности
FR[11]	направляющий косинус единичного вектора плоскости дуги к оси X
FR[12]	направляющий косинус единичного вектора плоскости дуги к оси Y
FR[13]	направляющий косинус единичного вектора плоскости дуги к оси Z
FR[14]	угол дуги (если этот параметр равен 0, то угол не рассчитан)
	Если FR[3]=4 (кубический полином)
FR[7]	KF[0]
FR[8]	KF[1]
FR[9]	KF[2]
FR[10]	KF[3]
FR[11]	KF[4]
FR[12]	KF[5]
FR[13]	KF[6]
FR[14]	KF[7]
FR[15]	KF[8]
FR[16]	KF[9]
FR[17]	KF[10]
FR[18]	KF[11]
	$X=KF[0]+KF[1]*U+KF[2]*U^2+KF[3]*U^3$ $Y=KF[4]+KF[5]*U+KF[6]*U^2+KF[7]*U^3$ $X=KF[8]+KF[9]*U+KF[10]*U^2+KF[11]*U^3$ где $0<=U<=1$
	Если FR[3]=7 (NURBS-сплайн)
FR[7]	количество контрольных точек сплайна
FR[8]	степень сплайна
FR[9]	признак рационального сплайна — равенство 1. Если признак равен 0, то массив контрольных точек отсутствует.
FR[10]	значение параметрической координаты начальной точки движения по сплайну

FR[11]	значение параметрической координаты конечной точки движения по сплайну
FR[i+100]	массив, содержащий контрольные точки. Доступ к параметрам осуществляется через FR[i+100], где i — порядковый номер контрольной точки
FR[i+100000]	массив, содержащий n весов контрольных точек. Для нерациональных сплайнов массив отсутствует. Доступ к параметрам осуществляется через FR[i+100000], i — порядковый номер контрольной точки
FR[i+200000]	массив, содержащий m коэффициенты: m=n+d-1. Доступ к параметрам осуществляется через FR[i+200000], где i — порядковый номер контрольной точки

Структура команды «Дополнительное криволинейное перемещение» (код 191)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=191
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	код кривой: 3 — задана пространственная дуга 4 — задан кубический полином 7 — задан NURBS-сплайн
FR[4]	координата X конечной точки движения по кривой
FR[5]	координата Y конечной точки движения по кривой
FR[6]	координата Z конечной точки движения по кривой
	Если FR[3]=3 (пространственная дуга)
FR[7]	координта X центра окружности
FR[8]	координта Y центра окружности
FR[9]	координта Z центра окружности
FR[10]	координта Z центра окружности
FR[11]	направляющий косинус единичного вектора плоскости дуги к оси X
FR[12]	направляющий косинус единичного вектора плоскости дуги к оси Y
FR[13]	направляющий косинус единичного вектора плоскости дуги к оси Z
FR[14]	угол дуги (если этот параметр равен 0, то угол не рассчитан)
	Если FR[3]=4 (кубический полином)
FR[7]	KF[0]
FR[8]	KF[1]
FR[9]	KF[2]
FR[10]	KF[3]
FR[11]	KF[4]
FR[12]	KF[5]
FR[13]	KF[6]
FR[14]	KF[7]
FR[15]	KF[8]
FR[16]	KF[9]

FR[17]	KF[10]
FR[18]	KF[11]
	$X=KF[0]+KF[1]*U+KF[2]*U^2+KF[3]*U^3$ $Y=KF[4]+KF[5]*U+KF[6]*U^2+KF[7]*U^3$ $X=KF[8]+KF[9]*U+KF[10]*U^2+KF[11]*U^3$ где 0<=U<=1
	Если FR[3]=7 (NURBS-сплайн)
FR[7]	количество контрольных точек сплайна
FR[8]	степень сплайна
FR[9]	признак рационального сплайна — равенство 1. Если признак равен 0, то массив контрольных точек отсутствует.
FR[10]	значение параметрической координаты начальной точки движения по сплайну
FR[11]	значение параметрической координаты конечной точки движения по сплайну
FR[i+100]	массив, содержащий контрольные точки. Доступ к параметрам осуществляется через FR[i+100], где i — порядковый номер контрольной точки
FR[i+100000]	массив, содержащий n весов контрольных точек. Для нерациональных сплайнов массив отсутствует. Доступ к параметрам осуществляется через FR[i+100000], i — порядковый номер контрольной точки
FR[i+200000]	массив, содержащий m коэффициенты: m=n+d-1. Доступ к параметрам осуществляется через FR[i+200000], где i — порядковый номер контрольной точки

Структура команды «Вызов подпрограммы» (код 223)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=223
FR[2]	порядковый номер объекта в маршруте обработки
FR[4]	количество повторов

Структура команды «Начало/конец подпрограммы» (код 252)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=252
FR[2]	порядковый номер объекта в маршруте обработки

Структура команды «Начало цикла» (код 401)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=401
FR[2]	порядковый номер объекта в маршруте обработки
	Если FR[0]=3 (явное задание начала цикла (координатами))
FR[30]	номер системы координат
FR[11]	координта Y начала цикла
FR[12]	координта Z начала цикла
	Если FR[0]=2 (начало цикла задано корректорами по осям)

FR[20]	корректор по оси X
FR[21]	корректор по оси Y
FR[22]	корректор по оси Z
	Если FR[0]=1 (начало цикла задано номером системы координат)
FR[30]	номер системы координат

Структура команды «Безопасная позиция» (код 451)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=451
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	код задания координаты X безопасной позиции 0 — не задана 1 — задан
FR[11]	координата X безопасной позиции
FR[12]	код задания координаты Y безопасной позиции 0 — не задана 1 — задан
FR[13]	координата Y безопасной позиции
FR[14]	код задания координаты Z безопасной позиции 0 — не задана 1 — задан
FR[15]	координата Z безопасной позиции

Структура команды «Плоскость холостых ходов» (код 452)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=452
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	код задания плоскости холостого хода 0 — выключена 1 — плоскость холостого хода XY 2 — плоскость холостого хода XZ 3 — плоскость холостого хода YZ 10 — выключение модальности, не отключает плоскость холостого хода 11 — модальная плоскость холостого хода XY 12 — модальная плоскость холостого хода XZ 13 — модальная плоскость холостого хода YZ
FR[4]	величина координаты плоскости холостого хода

Структура команды «Команда пользователя» (код 459)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=459
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	номер команды пользователя (от 0 до 9999)
FR[i+10]	массив, содержащий параметры команды пользователя. Доступ к параметрам осуществляется через FR[i+10], где i — порядковый номер параметра

Структура команды «Параметры пользователя - закладка «Параметры пользователя» в диалоге тех.перехода или команды» (код 460)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=460
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	количество параметров
FR[i+100]	массив, содержащий параметры параметры пользователя. Доступ к параметрам осуществляется через FR[i+100], где i — порядковый номер параметра

Структура команды «Величина аппроксимации» (код 491)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=491
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	величина аппроксимации

Структура команды «Турета» (код 493)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=493
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	номер туреты

Структура команды «Комментарий» (код 582)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=582
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	текст комментария

Структура команды «Глубина резания» (код 900)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=900
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	величина глубины резания

Структура команды «Плоскость интерполяции» (код 901)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=901
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	код задания плоскости интерполяции 0 — задана плоскость XY 1 — задана плоскость YZ 2 — задана плоскость ZX
FR[11]	величина соответсвукющей координаты, определяющей положение плоскости

Структура команды «Трансформ» (код 10123)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=10123
FR[2]	порядковый номер объекта в маршруте обработки
FR[3]	компонента вектора оси X СК КЭ в СК детали
FR[4]	компонента вектора оси Y СК КЭ в СК детали
FR[5]	компонента вектора оси Z СК КЭ в СК детали
FR[6]	компонента вектора оси X СК КЭ в СК детали
FR[7]	компонента вектора оси Y СК КЭ в СК детали
FR[8]	компонента вектора оси Z СК КЭ в СК детали
FR[9]	компонента вектора оси X СК КЭ в СК детали
FR[10]	компонента вектора оси Y СК КЭ в СК детали
FR[11]	компонента вектора оси Z СК КЭ в СК детали
FR[12]	координата X центра СК КЭ в СК детали
FR[13]	координата Y центра СК КЭ в СК детали
FR[14]	координата Z центра СК КЭ в СК детали

Структура команды «Фрезеровать» (код 301)

FR[0]	длина внутренней структуры фразы
FR[1]	код фразы=301
FR[2]	порядковый номер объекта в маршруте обработки
FR[10]	адрес инструмента
FR[11]	параметр НТК 0 — выключен 1 — включен
FR[12]	радиусная коррекция 0 — выключена 1 — включен 3 — заданы касательные и перпендикулярные отрезки
FR[13]	направление фрезерования 0 — встречное 1 — попутное

FR[14]	схема резания 0 — выключен 1 — по нормали 2 — линейное 3 — линейное+наклон 4 — радиусное 5 — радиусное+наклон 6 — контурное спиральное 7 — контурное+наклон
FR[15]	тип обработки 0 — нет 1 — эквидистанта 2 — обратная эквидистанта 3 — петля эквидистантная 4 — зигзаг эквидистантный 5 — спираль простая 6 — петля 7 — зигзаг 8 — петля UV 9 — зигзаг UV 10 — петля контурная 11 — зигзаг контурный 12 — «карандашная» 13 — петля контурная II 14 — зигзаг контурный II 15 — эквидистанта II 16 — обратная эквидистанта II 17 — спираль 18 — спираль обратная 19 — спираль с XX
FR[16]	вид фрезерования 0 — 2.5X обработка 1 — обратка по Z-уровням 2 — 2X обработка 3 — 3X обработка 4 — 4X обработка с постоянным углом инструмента по оси Y 5 — 4X обработка с постоянным углом инструмента по оси X 6 — 5X обработка торцем фрезы 7 — 5X обработка боковой поверхностью инструмента 10 — обработка наклонной рабочей плоскости с установкой шпинделя инструмента 12 — обработка с поворотом стола по одной оси 15 — лазерная 2.5X обработка 16 — лазерная 5X обработка 17 — лазерная 5X обработка боковой поверхностью цилиндра 20 — плунжерное фрезерование

FR[17]	признак подбора 0 — подбор не задан 1 — подбор задан
FR[19]	код глубины врезания 0 — не задан 1 — задана глубина резания 2 — задано количество врезаний
FR[20]	код скругления траектории 0 — не скруглять 1 — скругление дуговое 2 — скругление кубическим сплайном 3 — скругление NURBS-сплайном
FR[22]	удаление «пеньков» 0 — «пеньки» не удалять 1 — удаление «пеньков» без учета геометрии инструмента 2 — удаление «пеньков»с учетомгеометрии инструмента
FR[23]	тип подхода к контуру 0 — эквидистантный 1 — линейный 2 — выключен 3 — радиальный 4 — задан в приращениях по 3 координатам dx, dy, dz 5 — задан в приращениях по 2 координатам dx, dy
FR[24]	код системы координат подхода 0 — в плоскости: вектор касательной-вектор нормали 1 — в вертикальной плоскости
FR[25]	тип отхода от контура 0 — эквидистантный 1 — линейный 2 — выключен 3 — радиальный 4 — задан в приращениях по 3 координатам dx, dy, dz 5 — задан в приращениях по 2 координатам dx, dy
FR[26]	код системы координат отхода 0 — в плоскости: вектор касательной-вектор нормали 1 — в вертикальной плоскости
FR[27]	код задания основной подачи 0 — не задана 1 — задана в мм/мин

	2 — задана в мм/об
FR[28]	код задания скорости вращения шпинделя 0 — не задана 1 — задана в об/мин 2 — задана скорость резания в м\мин
FR[29]	код задания подачи врезания 0 — не задана 1 — задана
FR[30]	код включения подачи СОЖ 0 — не включена 1 — включена
FR[32]	код задания глубины резания 0 — глубина резания не задана 1 — задано значение глубины резания 2 — задан коэффициент глубины резания от радиуса инструмента 3 — задан коэффициент изменения каждого последующего прохода
FR[34]	код сохранения петель или отключения контроля коллизий 0 — контроль коллизий выключен 1 — контроль коллизий включен
FR[38]	код места формирования ПОДХОДА/ОТХОДА 0 — на каждом проходе 1 — при переходе с холостого хода на рабочую подачу и наоборот
FR[40]	код включения подачи СОЖ 0 — обработка с врезанием 1 — обработка с подъёмом
FR[41]	величина основной подачи
FR[42]	величина скорости резания или заданного количества оборотов шпинделя
FR[45]	длина касательного участка при включении корректора
FR[46]	длина перпендикулярного участка при включении коррекции
FR[47]	величина глубины резания
FR[49]	определяет оси вращения 0 — вращение вокруг оси X 1 — вращение вокруг оси Y 2 — вращение вокруг оси Z
FR[50]	определение вращения рабочих органов 0 — обработка с вращением стола 1 — обработка вращением шпинделя
FR[51]	ориентация шпинделя

	0 — ориентация инструмента нормально к оси вращения 1 — ориентация инструмента нормально к поверхности 2 — заданы угол опережения и (или) угол отклонения 3 — фиксированные углы установки инструмента
FR[52]	код скоростного фрезерования 0 — обычное фрезерование 1 — скоростное фрезерование 2 — скоростное трохоидальное фрезерование
FR[53]	шаг врезания
FR[61]	глубина врезания
FR[63]	высота гребешка
FR[64]	припуск на контрольные поверхности при обработке КЭ «Поверхность» (обработка 3Х, 4Х, 5Х)
FR[66]	величина аппроксимации
FR[69]	угол установки инструмента к оси X при фиксированном положении или угол опережения
FR[70]	угол установки инструмента к оси Y при фиксированном положении или угол опережения
FR[71]	код обкатки 0 — края отверстия не обкатывать в 3Х фрезеровании и края КЭ в 2.5Х 1 — обкатывать края отверстия
FR[72]	код аппроксимации 0 — не задана (по умолчанию величина аппроксимации 0,01) 1 — задана
FR[74]	код контроля на коллизии со шпинделем 0 — контроль выключен 1 — контроль включен
FR[75]	значение нахлёста
FR[76]	радиус / длина / приращение по оси X подхода
FR[77]	угол / шаг / приращение по оси Y подхода
FR[78]	радиус / длина / приращение по оси X отхода
FR[79]	угол / шаг / приращение по оси Y отхода
FR[80]	приращение по оси Z подхода
FR[81]	приращение по оси Z отхода
FR[82]	длина продления первого элемента траектории обработки
FR[83]	длина продления конечного элемента траектории обработки
FR[84]	глубина резания на последнем проходе
FR[85]	признак последовательной обработки при глубине врезания больше 0
FR[86]	код обработки по Z (устанавливается при наличии глубины врезания) 0 — послойная обработка 1 — зигзагом по Z 2 — спиральная обработка с зачисткой на дне 3 — спиральная без зачистки (нарезание резьбы фрезой)

FR[87]	признак движения лазера по нормали к траектории 0 — лазер движется не по нормали 1 — лазер движется по нормали
FR[89]	код задания глубины резания на последнем проходе 0 — глубина не задана 1 — глубина задана
FR[90]	код коррекции врезания при обнаружении коллизий при выполнении заданной схемы врезания 0 — игнорировать коллизии, выполняя требуемую схему врезания, не пытаясь менять угол врезания 1 — закончить обработку ТО на текущей глубине, при невозможности выполнить заданную схему врезания при других углах врезания в плане 2 — обработать с данной схемой врезания там, где возможно
FR[91]	код задания Z обработки плоскими поверхностями (активен при задании глубины резания) 0 — не учитывать плоские поверхности 1 — учитывать плоские поверхности
FR[92]	признак необходимости сопряженной обработки (для UV-обработки) 0 — нет сопряженной обработки 1 — есть сопряженная обработка
FR[93]	глубина врезания или длина врезания скорректированная
FR[94]	максимальный угол излома траектории
FR[95]	минимальный радиус скругления
FR[96]	ширина трохоиды