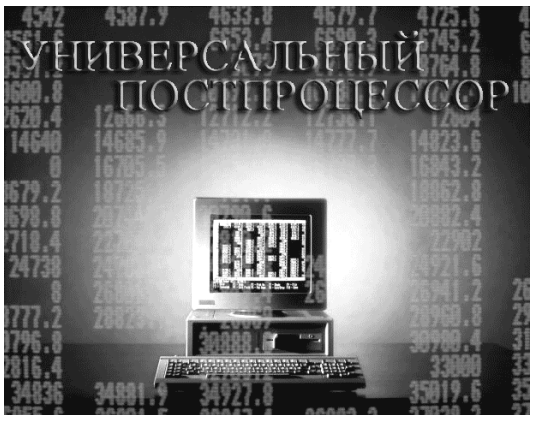


Глава 2

Описание постпроцессора



Описание постпроцессора

Общий формат описания постпроцессора

Описание постпроцессора должно начинаться с заголовка, за которым могут следовать два раздела описания - раздел инициализации (начальных установок) и главный раздел.

Заголовок состоит из служебного слова **REM**, за которым следует текстовая строка (строковый литерал). Эта строка служит для идентификации данного постпроцессора в системе ГеММа-3D. Она высвечивается в меню постпроцессоров при работе с библиотекой постпроцессоров. . В данную строку рекомендуется помещать маркировки системы ЧПУ и станка, для которого данный постпроцессор предназначен, например:

```
REM 2С42-61 модель 2Е450АМФ; абсолютный
```

Раздел инициализации представляет собой последовательность описаний необходимых разработчику констант и переменных, а также операторов, выполняемых интерпретатором перед началом анализа исходной программы. В разделе инициализации с помощью операторов присваивания устанавливаются значения параметров, которые сохраняются неизменными (таких как дискретность стойки УЧПУ), а также начальных значений параметров, которые могут изменяться в зависимости от содержания исходной программы (например, координаты начальной точки движения инструмента):

```
' Пример части раздела инициализации.
RDISCR = 0.001    'Дискретность по перемещениям 0.001
NSTART# = 10      'Идентификатор NSTART# в дальнейшем может
                   'использоваться для обозначения целой
                   'константы 10 и будет применяться в
качестве
                   'начального значения счетчика кадров.
' . . . Другая инициализация . . .
```

Главный раздел служит для описания генерации управляющей программы на основе значений параметров команд исходной программы. Именно в главном разделе описывается, что, в каком порядке и при каких условиях записывается в управляющую программу данным постпроцессором. Главный раздел представляет собой замкнутый цикл, тело которого выполняется интерпретатором постпроцессоров при обработке каждой команды исходной программы. Для извлечения очередной команды из исходной программы, в начале цикла должен стоять вызов процедуры-функции **NEXT_STEP**. Общая структура главного раздела выглядит следующим образом:

```
DO ' Начало цикла.
  CALL NEXT_STEP () ' Извлечение очередной команды.
  '                  . . .
  ' . . . Тело цикла - обработка команды . . .
  '                  . . .
LOOP ' Переход к началу цикла.
```

Имя выходного файла для записи машинной программы находится в предопределенной переменной fileout$. Перед первой записью в выходной файл, его необходимо открыть. Обычно это делается непосредственно перед началом цикла главного раздела:

```
Open fileout$ For Output As id#
```

Теперь мы можем построить простейший постпроцессор. Задача этого постпроцессора создать файл машинной программы со следующими требованиями:

- заголовок, состоящий из символа '%';
- кадры, соответствующие перемещениям в абсолютном режиме, с нумерацией кадров с шагом 1, содержащие:
- G-функции, соответствующие виду перемещения (60, 61, 62, 63);
- координаты **x, y, z** конечной точки перемещения для линейной интерполяции и позиционирования по адресам **X, Y, Z** или координаты **x** и **y** конечной точки дуги и координаты i и j центра дуги по адресам **X, Y, I, J** для круговых перемещений;
- значения рабочей подачи по адресу F
- Завершающий управляющую программу кадр с функцией M02.

Координаты должны быть выражены в мм и выводиться в формате с обязательной плавающей точкой и знаком, дискретность должна быть равна 0.001 мм. Пример описания такого постпроцессора:

```
REM Простейший постпроцессор - только перемещения
' Файл SMPL1.PSS содержит описание простейшего постпроцессора
'
' Дискретность :              0.001 мм. - декартовы
перемещения.
' Основной режим:             в абсолютах.
' Функции :                    Линейная и круговая
интерполяция.
' описания
' Раздел инициализации содержит
' присвоение начальных значений основным параметрам
' и определение имен констант, вводимых пользователем

RDISCR = 0.001          ' Параметром RDISCR значение
                        ' дискретность по перемещениям
задаётся,
                        ' равной 0.001

Open fileout$ For Output As id1#
DO
CALL NEXT_STEP()

' Главный раздел содержит описание действий
' постпроцессора по формированию управляющей программы
' для каждой анализируемой команды исходной программы.
' В зависимости от вида анализируемой команды
' т.е. значения параметра command#
' описываем действия постпроцессора по
' формированию управляющей программы:
```

```
if command# = HEADER# then    ' Информация в начале программы.
write id1# using "%%%s" , NL$)   ' Вывод заголовка (знак %)
n# = 1                             ' Счетчик кадров
устанавливается в 1
elseif command# = END# then    ' Информация в конце программы.
write id1# using "N%d", n#     ' Вывод номера текущего кадра
n# = n# + 1                     ' Увеличение счетчика кадров
write id1# using "MO 2"         ' Вывод функции MO 2
write id1# using "%s", NL$     ' Завершение кадра - символ LF
elseif command# = MOVE# then   ' Информация, связанная с
                                ' перемещением инструмента,
write id1# using "N%d", n#     ' Вывод номера текущего кадра
n# = n# + 1                     ' Увеличение счетчика кадров
' Вывод соответствующей G-функции линейной
' или круговой интерполяции:
if primitive# = LOCATE# then
write id1# using "GOO"
elseif primitive# = LINE# then
write id1# using "G01"
elseif primitive# = ARC_CLW# then
write id1# using "GO 2"
elseif primitive# = ARC_CCLW# then
write id1# using "G03"
endif
if primitive# = ARC_CLW# OR primitive# = ARC_CCLW# then

' Для круговой интерполяции вывод X, Y, I, J
write id1# using "X%+.f", x * RDISCR
write id1# using "Y%+.f", y * RDISCR
write id1# using "I%+.f", i * RDISCR
write id1# using "J%+.f", j * RDISCR

elseif primitive# = LINE# OR primitive# = LOCATE# then

' Иначе, для позиционирования или линейной интерполяции
' вывод X, Y, Z

write id1# using "X%+.f", x * RDISCR
write id1# using "Y%+.f", y * RDISCR
write id1# using "Z%+.f", z * RDISCR

endif

if primitive# <> LOCATE# then

' Вывод подачи, если перемещение не является движением с G0
write id1# using "F%.15g", f
endif

write id1# using "%s", NL$      ' Завершение кадра - символ LF

endif

LOOP
' конец описания и конец файла SMPL1.PSS
```

Модифицируем описание этого постпроцессора с тем, чтобы:

- G-функции, описывающие вид перемещения, записывались в кадр только при смене режима интерполяции
- запись координат x, y, z конечной точки перемещения для линейной интерполяции и позиционирования по адресам X, Y, Z производилась только при изменении соответствующей координаты
- запись величины рабочей подачи по адресу F производилась только при изменении величины подачи
- запись кадров, содержащих функции смены инструмента - трехзначный номер инструмента по адресу T и MO6 - производилась в соответствии с требованиями исходной программы. Внося в описание SMPL1.PSS необходимые изменения, получим следующее описание постпроцессора:

```
REM Модифицированный постпроцессор со сменой инструмента
'************************************************************
'Файл SMPL2.PSS содержит описание модифицированного
' постпроцессора,
'см. также файл SMPL1.PSS
'
'Дискретность:          0.001 мм. - декартовы перемещения.
'Основной режим:        в абсолютах.
'Функции:               Линейная и круговая интерполяция.
'                       смена инструмента
'
'************************************************************

' Раздел инициализации содержит
' присвоение начальных значений основным параметрам

RDISCR = 0.001    ' присвоение параметру RDISCR значения
                  ' дискретности по перемещениям 0.001 мм
Open fileout$ For Output As id1#
DO
CALL NEXT_STEP()
' Главный раздел содержит описание действий
' постпроцессора по формированию управляющей программы
' для каждой анализируемой команды исходной программы.
if command# = HEADER# then     ' Вывод информации в начале
                               ' программы.
write id1# using "%%%s" , NL$)' Вывод заголовка (знак %
n# = 1                        ' Счетчик кадров
                               ' устанавливается равным 1
elseif command# = END# then    ' Вывод информации в конце
                               ' программы.
write id1# using "N%d", n#     ' Вывод номера текущего кадра
n# = n# + 1                    ' Увеличение счетчика кадров
write id1# using "MO 2"        ' Вывод функции MO 2
write id1# using "%s", NL$     ' Завершение кадра - символ LF
elseif command# = LOAD_TOOL# then
                               ' Вывод информации,
                               ' связанной со сменой
                               ' инструмента,
write id1# using "N%d", n#     ' Вывод номера текущего кадра
```

```
n# = n# + 1                        ' Увеличение счетчика кадров
write id1# using "T%.3dM06", cutter_no#
                                   ' Вывод номера инструмента
                                   ' и M06
write id1# using "%s", NL$         ' Завершение кадра - символ LF
elseif command# = MOVE# then      ' Вывод информации,
                                   ' связанной с перемещением
                                   ' инструмента,
write id1# using "N%d", n#         ' Вывод номера текущего кадра
n# = n# + 1                        ' Увеличиваем счетчик кадров
' если на анализируемом перемещении
' изменился режим движения
' (т.е. параметр primitive# анализируемой команды
' не равен аналогичному параметру предшествующей
' команды) :
if primitive# <> primitive_# then
' Вывод соответствующей G-функции линейной
' или круговой интерполяции :
if primitive# = LOCATE# then
write id1# using "GOO"
elseif primitive# = LINE# then
write id1# using "G01"
elseif primitive# = ARC_CLW# then
write id1# using "GO 2"
elseif primitive# = ARC_CCLW# then
write id1# using "G03"
endif
endif

if (primitive# = ARC_CLW#) OR (primitive# = ARC_CCLW#) then
' Для круговой интерполяции вывод X, Y, I, J:
write id1# using "X%+.f", x * RDISCR
write id1# using "Y%+.f", y * RDISCR
write id1# using "I%+.f", i * RDISCR
write id1# using "J%+.f", j * RDISCR

elseif (primitive# = LINE#) OR (primitive# = LOCATE#) then
' Иначе, для позиционирования или линейной интерполяции
' вывод X, Y, Z, причем
' Запись по адресу X, если приращение этой координаты,
' dx, не равно 0:
if dx<>0.0 then
 write id1# using "X%+.f", x * RDISCR
endif
' Запись по адресу Y, если приращение этой координаты,
' dy, не равно 0:
if dy <> 0.0 then
 write id1# using "Y%+.f", y * RDISCR
endif
' Запись по адресу Z, если приращение этой координаты,
' dz, не равно 0:
if dz<>0.0 then
 write id1# using "Z%+.f", z * RDISCR
endif
endif

if (primitive#<>LOCATE#) AND (f <> f_) then
```

```
' Вывод подачи, если перемещение не является движением с GO O
' и на данном перемещении задано новое значение подачи.
' Обратите внимание на расстановку скобок в условии.
write id1# using "F%.15g", f
endif

write id1# using "%s", NL$  ' Завершение кадра - символ LF
endif

LOOP
' конец описания и конец файла SMPL2.PSS
```

В двух предыдущих примерах увеличение счетчика кадров на единицу описывалось оператором

```
n# = n# + 1
```

При необходимости изменить шаг приращения счетчика кадров, к примеру, сделать его равным 10, все вхождения оператора

```
n# = n# + 1
```

необходимо заменить на

```
n# = n# + 10
```

Чтобы избежать в дальнейшем необходимости вносить изменения сразу в нескольких местах текста, в раздел инициализации внесем описания двух поименованных констант, например, NSTART# и NSTEP#, которые будут обозначать номер первого кадра, и шаг, с которым номера кадров нумеруются. Теперь оператор, изменяющий номер кадра, будет выглядеть так:

```
n# = n# + NSTEP#
```

и чтобы изменить шаг нумерации выводимых кадров, достаточно будет изменить лишь значение константы NSTEP# в ее определении.

Кроме того, введем в описание постпроцессора операторы, которые должны:

- Отдельным кадром выводить скорость вращения шпинделя по адресу S и функции M03 или M04 в зависимости от направления вращения при включении шпиделя и M05 при выключении.
- Выводить функции M08 или M09 для включения или выключения охлаждения.
- При подходе к контуру с включением правой или левой коррекции в кадр с перемещением вслед за номером кадра будем выводить G41/42 и двузначный номер корректора по адресу D.

  При отходе от контура, с выключением коррекции на радиус инструмента, вслед за номером кадра будем выводить G40.

Ниже представляется полный текст описания такого гипотетического постпроцессора, которое может служить основой для описаний реальных постпроцессоров:

```
REM Гипотетический постпроцессор с коррекцией
'************************************************************
' Файл SMPL3.PSS содержит описание гипотетического
' постпроцессора,
```

```
' см. также файлы SMPL1.PSS и SMPL2.PSS
'
'SMPL3.PSS - файл спецификаций постпроцессора
'
'
'Дискретность:          0.001 мм. - декартовы перемещения.
'Основной режим:        в абсолютах.
'Функции:               Линейная и круговая интерполяция
'                       смена инструмента
'                       вывод коррекции на радиус инструмента
'*********************************************************
' Раздел инициализации содержит
' присвоение начальных значений основным параметрам
' и определение имен констант, вводимых пользователем

RDISCR = 0.001   'присвоение параметру дискретности по
перемещениям RDISCR значения 0.001
NSTART# = 10      'Идентификатор NSTART# в дальнейшем может
                  'использоваться для обозначения целой
                  'константы 10 и будет применяться в качестве
                  'начального значения счетчика кадров.
 NSTEP# = 5       'Идентификатор NSTEP# в дальнейшем может
                  'использоваться для обозначения целой
                  'константы 5 и будет применяться в качестве
                  'шага нумерации кадров.
Open fileout$ For Output As id1#
DO
CALL NEXT_STEP()
' Главный раздел содержит описание действий
' постпроцессора по формированию управляющей программы
' для каждой анализируемой команды исходной программы.
' Если режим вращения шпинделя изменился, т.е.
' значение параметра sm# отличается от значения
' аналогичного параметра предшествующей команды sm_#:

if sm# <> sm_# then
write id1# using "N%d", n#     ' Вывод номера текущего кадра
n# = n# + NSTEP#               ' Увеличение счетчика кадров

' Вывод функции направления вращения шпинделя
' в зависимости от значения параметра sm#:
if sm# = SPIN_CLW# then

' Вывод оборотов шпинделя по адресу S
' и функции M03:
write id1# using "S%.15gM03", s

elseif sm# = SPIN_CCLW# then

' Вывод оборотов шпинделя по адресу S
' и функции MO 4:
write id1# using "S%.15gM04", s

elseif sm# = SPIN_OFF# then

' Вывод функции MO 5:
write id1# using "MO 5"
```

```
endif

write id1# using "%s", NL$      ' Завершение кадра - символ LF

endif

' Если режим охлаждения изменился, т.е.
' значение параметра cool# отличается от значения
' аналогичного параметра предшествующей команды cool_#:
if cool#<>cool_# then

write id1# using "N%d", n#     ' Вывод номера текущего кадра
n# = n# + NSTEP#               ' Увеличение счетчика кадров

' Вывод функции направления вращения шпинделя
' в зависимости от значения параметра cool#:
if cool# = ON# then
write id1# using "MO 8"
elseif cool# = OFF# then
write id1# using "M09"
endif
write id1# using "%s", NL$     ' Завершение кадра - символ LF
endif

' В зависимости от вида анализируемой команды
' т.е. значения параметра command#
' описываются действия постпроцессора по
' формированию управляющей программы:

if command# = HEADER# then     ' Вывод информации в начале
                               ' программы,

write id1# using "%%%s" , NL$ ' Вывод заголовка знак %
n# = NSTART#                   ' Счетчик кадров
                               ' устанавливается в 1

elseif command# = END# then    ' Вывод информации в конце
                               ' программы.

write id1# using "N%d", n#     ' Вывод номера текущего кадра
n# = n# + NSTEP#               ' Увеличение счетчика кадров
write id1# using "MO 2"        ' Вывод функции MO 2
write id1# using "%s", NL$     ' Завершение кадра - символ LF

elseif command# = LOAD_TOOL# then
                               'Вывод информации,
                               'связанной со сменой
инструмента
write id1# using "N%d", n#     'Вывод номера текущего кадра
n# = n# + NSTEP#               'Увеличение счетчика кадров
write id1# using "T%.3dM06", cutter_no#
                               'Вывод номера инструмента и M06
write id1# using "%s", NL$     'Завершение кадра - символ LF.

elseif command# = MOVE# then   ' Вывод информации,
                               ' связанной с перемещением
                               ' инструмента,
write id1# using "N%d", n#     ' Вывод номера текущего кадра
```

```
n# = n# + NSTEP#                ' Увеличение счетчика кадров

' если на анализируемом перемещении
' изменился режим движения
' (т.е. параметр primitive# анализируемой команды
' не равен аналогичному параметру предшествующей
' команды primitive_#):
if primitive# <> primitive_# then

' Вывод соответствующей G-функции линейной
' или круговой интерполяции:
if primitive# = LOCATE# then
write id1# using "G00"
elseif primitive# = LINE# then
write id1# using "G01"
elseif primitive# = ARC_CLW# then
write id1# using "G02"
elseif primitive# = ARC_CCLW# then
write id1# using "G03"
endif
endif

' Если анализируемая команда
' является "подходом" к контуру с включением левой
' коррекции радиуса инструмента, выводится функция
' G41 с корректором по адресу D:

if contour_to_l# then
write id1# using "G41D%.2d", cutter_r_crtr#

' Иначе, если анализируемая команда
' является "подходом" к контуру с включением правой
' коррекции радиуса инструмента, выводится функция
' G42 с корректором по адресу D:

elseif contour_to_r# then
write id1# using "G42D%.2d", cutter_r_crtr#

' Иначе если анализируемая команда
' является "отходом" от контура (на нем происходит
' выключение коррекции радиуса инструмента)
' то выводится функция G40

elseif contour_from# then
write id1# using "G40"

endif

if (primitive# = ARC_CLW#) OR (primitive# = ARC_CCLW#) then

' Для круговой интерполяции выводятся X, Y, I, J:
write id1# using "X%+.f", x * RDISCR
write id1# using "Y%+.f", y * RDISCR
write id1# using "I%+.f", i * RDISCR
write id1# using "J%+.f", j * RDISCR

elseif (primitive# = LINE#) OR (primitive# = LOCATE#) then
```

```
' Иначе, для позиционирования или линейной интерполяции
' выводятся X, Y, Z, причем
' Запись по адресу X, если приращение координаты
' dx не равно 0:

if dx<>0.0 then
write id1# using "X%+.f", x * RDISCR
endif

' Запись по адресу У, если приращение координаты
' dy не равно 0:

if dy<>0.0 then
write id1# using "Y%+.f", y * RDISCR
endif

' Запись по адресу Z, если приращение координаты
' dz не равно 0:

if dz<>0.0 then
write id1# using "Z%+.f", z * RDISCR
endif
endif

if (primitive# <> LOCATE#) AND (f<>f_) then

' Вывод подачи, если перемещение не является движением с G00
' и на данном перемещении задано новое значение подачи.
' Обратите внимание на расстановку скобок в условии.

write id1# using "F%.15g", f

endif

write id1# using "%s", NL$    ' Завершение кадра - символ LF
endif

LOOP
' конец описания и конец файла SMPL3.PSS
'*****************************************************
```

Далее приводится пример описания реального постпроцессора для станка горизонтального копировально-фрезерного модели 6В443Ф3, работающего под управлением УЧПУ 2С42.

Этот постпроцессор выводит в управляющую программу также:

- функции включения коррекции на длину инструмента;
- кадры выхода в ноль станка при смене инструмента;
- данные о дальнейшем поведении траектории инструмента (на основе значений параметров семейства tx) при включении коррекции на радиус инструмента.

Кроме того, это описание содержит примеры использования стандартных функций HYPOT и ROUND.

Подробные комментарии см. в тексте описания:

```
REM 2C42 модель 6B443ФЗ
'*******************************************************
' 6V443F3.PSS - файл спецификаций постпроцессора
'
' УЧПУ 2C42
' станок горизонтальный копировально-фрезерный модель 6B443ФЗ
'
' Дискретность :      0.001 мм. - декартовы перемещения.
' Основной режим :    в абсолютах.
' Функции :            1. контурная обработка с коррекцией
'                          на длину и радиус инструмента
'                      2. программируемая смена инструмента
'*******************************************************
' Раздел инициализации содержит
' присвоение начальных значений основным параметрам
' и определение имен констант, вводимых пользователем
RDISCR = 0.001      ' дискретность по перемещениям 0.001 мм
NSTART# = 1         ' Идентификатор NSTART# в дальнейшем может
                    ' использоваться для обозначения целой
                    ' константы и будет применяться в качестве
                    ' начального значения счетчика кадров.

 NSTEP# = 1 :       ' Идентификатор NSTEP# в дальнейшем может
                    ' использоваться для обозначения целой
                    ' константы и будет применяться в качестве
                    ' шага нумерации кадров.
Open fileout$ For Output As id1#
DO
CALL NEXT_STEP()
' Главный раздел содержит описание действий
' постпроцессора по формированию управляющей программы
' для каждой анализируемой команды исходной программы.
' Если режим вращения шпинделя изменяется, т.е.
' значение параметра sm# отличается от значения
' аналогичного параметра предшествующей команды sm_#:

if sm# <> sm_# then

write id1# using "N%d", n#      ' Вывод номера текущего кадра
n# = n# + NSTEP#                ' Увеличение счетчика кадров

' Вывод функции направления вращения шпинделя
' в зависимости от значения параметра sm# :

if sm# = SPIN_CLW#              ' Вывод функции МОЗ:
write id1# using "MOЗ"

elseif sm# = SPIN_CCLW#         ' Вывод функции MO 4:

write id1# using "MO 4"

elseif sm# = SPIN_OFF#          ' Вывод функции MO 5:

write id1# using "MO 5"

endif

write id1# using "%s", NL$     ' Завершение кадра - символ LF
```

```
endif

' Если режим охлаждения изменился, т.е.
' значение параметра cool# отличается от значения
' аналогичного параметра предшествующей команды cool_:

if cool#<>cool_# then

write id1# using "N%d", n#        ' Вывод номера текущего
кадра
n# = n# + NSTEP#                  ' Увеличение счетчика кадров

' Вывод функции охлаждения
' в зависимости от значения параметра cool#:

if cool# = ON# then
write id1# using "MO 8"

elseif cool# = OFF# then
write id1# using "MO 9"
endif

write id1# using "%s", NL$       'Завершение кадра - символ LF

endif

' В зависимости от вида анализируемой команды
' т.е. значения параметра command#
' описывются действия постпроцессора по
' формированию управляющей программы:

if command# = HEADER# then    ' Информация в начале программы.
write id1# using "@%s%s", partno_text$, NL$
                             'Вывод строки PARTNO
                             '(текст на ленте),
                             'предваренной символом "@"
n# = NSTART#                 'Установка начального значения
                             'счетчика кадров
write id1# using "%%"         'Вывод знака процента
write id1# using "%s", NL$    'Вывод знака LF

elseif command# = END# then  'Информация в конце программы

write id1# using "N%d", n#    'Вывод номера текущего кадра
n# = n# + NSTEP#             'Увеличение счетчика кадров
write id1# using "MO 2"       'Вывод функции MO 2
write id1# using "%s", NL$    'Завершение кадра - символ LF

elseif command# = LOAD_TOOL# then
                             ' Вывод информации,
                             ' связанной со сменой
                             ' инструмента.

write id1# using "N%d", n#    'Вывод номера текущего кадра
n# = n# + NSTEP#             'Увеличение счетчика кадров
write id1# using "G90G17G54XOYOZO"
```

```
                                    'Вывод выхода в 0 программы
write id1# using "%s", NL$          'Завершение кадра - символ LF

write id1# using "N%d", n#          'Вывод номера текущего кадра
n# = n# + NSTEP#                    'Увеличение счетчика кадров
write id1# using "T%.3d", pass_cutter_no#
                                    'Вывод номера инструмента
write id1# using "%s", NL$          'Завершение кадра - символ LF

elseif command# = MOVE# then   ' Вывод информации,
                                    ' связанной с перемещением
инструмента.
write id1# using "N%d", n#          ' Вывод номера текущего кадра
n# = n# + NSTEP#                    ' Увеличение счетчика кадров

'если на анализируемом перемещении
'изменился режим движения
'(т.е. параметр primitive# анализируемой команды
'не равен аналогичному параметру предшествующей
'команды primitive_#)
'или включается режим коррекции длины,
'(т.е. параметр lcomp анализируемой команды
'не равен аналогичному параметру предшествующей
'команды lcomp_# и текущее значение параметра lcomp
'не равно LCOMP_OFF#)
'или анализируемая команда
'является "подходом" к контуру с включением
'коррекции радиуса инструмента

if (primitive#<>primitive_#)
OR (lcomp<>lcomp_#) AND (lcomp<>LCOMP_OFF#)
OR contour_to# then

' Вывод соответствующей G-функции линейной
' или круговой интерполяции :

if primitive# = LOCATE# then
write id1# using "GOO"
elseif primitive# = LINE# then
write id1# using "GO1"
elseif primitive# = ARC_CLW# then
write id1# using "G02"
elseif primitive# = ARC_CCLW# then
write id1# using "G03"
endif

' Для включения левой коррекции
' вывод G41 с корректором
' по адресу D:

if contour_to_l# then
write id1# using "G41D%.2d", cutter_r_crtr#

' Иначе, для включения правой коррекции
' вывод G42 с корректором
' по адресу D:

elseif contour_to_r# then
```

```
write id1# using "G42D%.2d", cutter_r_crtr#

' Иначе, для включения коррекции на длину инструмента
' вывод G43 с корректором по адресу D:

elseif (lcomp<>lcomp_#) AND (lcomp<>LCOMP_OFF#) then
write id1# using "G43D%.2d", cutter_l_crtr#

endif
endif

' Для выключения коррекции на радиус инструмента (отход)
' вывод G45 :

if contour_from# then

write id1# using "G45"

' Иначе, для выключения коррекции на длину инструмента
' вывод G46 с предыдущим корректором:

elseif (lcomp<>lcomp_#) AND (lcomp = LCOMP_OFF#) then
write id1# using "G46D%.2d", cutter_l_crtr_#

endif

' Для круговой интерполяции вывод по адресам X, У, I, J:

if ((primitive# = ARC_CLW#) OR (primitive# = ARC_CCLW#)) then
if dx <> 0.0 then
write id1# using "X%+f", x
endif
if dy <> 0.0 then
write id1# using "Y%+f", y
endif
write id1# using "I%+f", i
write id1# using "J%+f", j

' Иначе, для позиционирования или линейной интерполяции
' вывод по адресам X, Y, Z:

elseif (primitive# = LINE#) OR (primitive# = LOCATE#) then
if dx<>0.0 then
write id1# using "X%+f", x
endif
if dy<>0.0 then
write id1# using "Y%+f", y
endif
if dz<>0.0 then
write id1# using "Z%+f", z
endif

endif

' При включении коррекции на радиус инструмента (подход):

if contour_to# then
```

```
' Вывод по адресам I и J соответствующих проекций вектора,
' направление которого совпадает с касательной к
' траектории движения в начале следующего кадра (перемещения)
' а длина равна длине анализируемого перемещения
' (Hypot (dx, dy))

if _tx0<>0.0 then

qqq = Round(_tx0*Hypot(dx,dy)/Hypot(_tx0,ty0))
write id1# using "I%+f", qqq
endif
if _ty0<>0.0 then
write id1# using "J%+f",
Round(_ty0*Hypot(dx,dy)/Hypot(_1x0,ty0))
endif

endif

'При изменении величины подачи
'(т.е. когда параметр f анализируемой команды
'не равен аналогичному параметру предшествующей
'команды f _)
'вывод подачи по адресу F:

if f<>f_ then
write id1# using "F%.15g", f
endif
write id1# using "%s", NL$ ' Завершение кадра - символ LF

endif

LOOP
' конец описания и конец файла 6V443F3.PSS
'******************************************************
```

Еще один пример - описание постпроцессора для станка BOSCH CC100M.

На этом примере показано, как описать работу постпроцессора при наличии подпрограмм в исходной программе. Для этого необходимо:

- обрабатывать коды параметра command# со значениями PROCEDURE# и RETURN# (начало подпрограммы и возврат из подпрограммы);
- обрабатывать код параметра command# со значением CALL# (вызов подпрограммы);
- при анализе команды перемещения проверять, принадлежит ли данная команда главной программе или подпрограмме, и в зависимости от этого задавать вывод координат инструмента в абсолютах или в приращениях.

  При этом необходимо выводить в управляющую программу функции, переключающие УЧПУ из абсолютного режима в инкрементальный и обратно.

Подробные комментарии см. в тексте описания:

```
REM BOSCH CC100M;                    абсолютный
'******************************************************
'CC100M.PSS - файл спецификаций постпроцессора
'Дискретность:     0.001 мм - декартовы перемещения.
```

```
'Основной режим    абсолютный.
'Функции:           1. контурная обработка с коррекцией
'                      на длину и радиус инструмента;
'                   2. программируемая смена инструмента.
'                   3. использование подпрограмм
'
' Раздел инициализации содержит
' присвоение начальных значений основным параметрам
' и определение имен констант, вводимых пользователем

RDISCR = 0.001  'присвоение параметру дискретности по
                 'перемещениям RDISCR значения 0.001

NSTART# = 1     'Идентификатор NSTART# в дальнейшем может
                 'использоваться для обозначения целой
                 'константы и будет применяться в качестве
                 'начального значения счетчика кадров.

NSTEP# = 1      'Идентификатор NSTEP# в дальнейшем может
                 ' использоваться для обозначения целой
                 ' константы
                 ' и будет применяться в качестве
                 ' шага нумерации кадров.
Open fileout$ For Output As id1#
DO
CALL NEXT_STEP()

' Главный раздел содержит описание действий
' постпроцессора по формированию управляющей программы
' для каждой анализируемой команды исходной программы.

' Если режим вращения шпинделя изменился, т.е.
' значения параметров s или sm# отличаются от значений
' аналогичных параметров предшествующей команды s_ или sm_#

if (s <> s_) OR (sm# <> sm_#) then

write id1# using "N%d", n#        ' Вывод номера текущего
кадра
n# = n# + NSTEP#                  ' Увеличение счетчика кадров

' Вывод функции направления вращения шпинделя
' в зависимости от значения параметра sm#:

if sm# = SPIN_CLW# then

' Вывод оборотов шпинделя по адресу S и функции M3:
write id1# using "S%.15gM3", s

elseif sm# = SPIN_CCLW# then

' Вывод оборотов шпинделя по адресу S и функции M4:
write id1# using "S%.15gM4", s

elseif sm# = SPIN_OFF# then

' Вывод функции M5:
write id1# using "M5"
```

```
endif

write id1# using "%s", NL$ ' Завершение кадра - символ LF

endif

' Если режим охлаждения изменился, т.е.
' значение параметра cool# отличается от значения
' аналогичного параметра предшествующей команды cool_:

if cool#<>cool_# then

write id1# using "N%d",n#      ' Вывод номера текущего кадра
n# = n# + NSTEP#               ' Увеличение счетчика кадров

' Вывод функции охлаждения
' в зависимости от значения параметра cool#:

if cool# = ON# then
 write id1# using "M8"
elseif cool# = OFF# then
 write id1# using "M9"
endif

write id1# using "%s", NL$     ' Завершение кадра - символ LF

endif

' В зависимости от вида анализируемой команды
' т.е. значения параметра command#
' описываются действия постпроцессора по
' формированию управляющей программы:

if command# = HEADER# then    ' Вывод информации в начале
программы.

' Вывод заголовка программы :
write id1# using "PROGRAMME %.2d %-18sM RWED 4%s",  part-
no#,"CC100M", NL$
elseif command# = END# then   ' Вывод информации в конце
программы.

' Вывод кадра, содержащего
' номер кадра n# по адресу N,
' функцию M30,
' ограничитель кадра LF:

write id1# using "N%dM30%s", n# , NL$
n# = n# + NSTEP#

elseif command# = PROGRAM# then
'Вывод информации,связанной с началом главной программы

' Установка начального значения счетчика кадров:

n# = NSTART#
```

```
' Вывод кадра, содержащего
' номер кадра n# по адресу N,
' функции G90 и G17,
' ограничитель кадра LF:

write id1# using "N%dG90G17%s", n# , NL$
n# = n# + NSTEP#

elseif command# = PROCEDURE# then    ' Вывод информации,
                                     ' связанной с
                                     ' началом подпрограммы.

' Вывод кадра, содержащего
' номер кадра n# по адресу N,
' номер подпрограммы по адресу S,
' ограничитель кадра LF :

write id1# using "N%d$%d%s", n#, proc_no# , NL$
n# = n# + NSTEP#

elseif command# = RETURN# then      ' Вывод информации,
                                    ' связанной с возвратом из
                                    ' подпрограммы.

' Вывод кадра, содержащего
' номер кадра n# по адресу N,
' функцию G99,
' ограничитель кадра LF :

write id1# using "N%dG99%s", n# , NL$
n# = n# + NSTEP#

elseif command# = CALL# then      ' Вывод информации,
                                  ' связанной с вызовом
                                  ' подпрограммы

'Вывод кадра, содержащего
'номер кадра n# по адресу N,
'функцию G22,
'номер вызываемой подпрограммы по адресу P,
'ограничитель кадра LF:

write id1# using "N%dG22P%d%s", n#, unit_call# , NL$
n# = n# + NSTEP#

elseif command# = LOAD_TOOL# then ' Вывод информации,
                                  ' связанной со сменой
                                  ' инструмента,

' Вывод кадра, содержащего
' номер кадра n# по адресу N,
' двузначный номер инструмента по адресу T,
' функцию M6,
' ограничитель кадра LF :

write id1# using "N%dT%.2d%.2dM6%s", n#, pass_cutter_no#,
pass_cutter_no# , NL$
n# = n# + NSTEP#
```

```
elseif command# = MOVE# then     ' Вывод информации,
                                  ' связанной с перемещением
инструмента.
write id1# using "N%d", n#        ' Вывод номера текущего кадра
n# = n# + NSTEP#                  ' Увеличение счетчика кадров
'Если анализируемое перемещение является первым перемещением в
'подпрограмме выводится G91 для перехода в инкрементальный
'режим:

if proc_begin# then
 write id1# using "G91"
' Иначе, если анализируемое перемещение принадлежит
' главной программе
' и ему предшествовала команда вызова подпрограммы,
' то выводится G90 для возврата в абсолютный режим:

elseif prog_body# AND (command_# = CALL#) then
write id1# using "G90"
endif

' Если на анализируемом перемещении
' сменилась плоскость интерполяции
' (т.е. параметр plane# анализируемой команды
' не равен аналогичному параметру предшествующей
' команды plane# _):

if plane# <> plane_# then

' Вывод G-функции соответствующей
' плоскости интерполяции:

if plane# = PLANE_XY# then
write id1# using "G17"
elseif plane# = PLANE_YZ# then
write id1# using "G19"
elseif plane# = PLANE_ZX# then
write id1# using "G18"
endif

endif

' если на анализируемом перемещении
' изменился режим движения
' (т.е. параметр primitive# анализируемой команды
' не равен аналогичному параметру предшествующей
' команды primitive_#):

if primitive# <> primitive_# then

' Вывод соответствующей G-функции линейной
' или круговой интерполяции:

if primitive# = LOCATE# then
write id1# using "GO"
elseif primitive# = LINE# then
write id1# using "Gl"
elseif primitive# = ARC_CCLW# then
```

```
write id1# using "G3"
elseif primitive# = ARC_CLW# then
write id1# using "G2"
endif

endif
' При подходе к контуру с включением левой коррекции
' выводится G41 и двузначный номер корректора:

if contour_to_l# then
write id1# using "G41T%.2d", cutter_r_crtr#

' Иначе, при подходе к контуру с включением правой коррекции
' выводится G42 и двузначный номер корректора:

elseif contour_to_r# then
write id1# using "G42T%.2d", cutter_r_crtr#

' Иначе, при отходе от контура с выключением правой коррекции
' выводится G40:

elseif contour_from# then
write id1# using "G40"
endif

' Если анализируемая команда перемещения выполняется
' в главной программе то текущие координаты выводятся в
' абсолютах:

if prog_body# then

' при ненулевом перемещении по оси X:
if dx<>0.0 then
' Выводится новое значение x:
write id1# using "X%.15g", x*RDISCR
endif

' при ненулевом перемещении по оси Y:
if dy<>0.0 then
' Выводится новое значение y:
write id1# using "Y%.15g", y*RDISCR
endif

' при ненулевом перемещении по оси Z:
if dz<>0.0 then
' Выводится новое значение z:
write id1# using "Z%.15g", z*RDISCR
endif

' Для круговой интерполяции вывод по адресам I, J, K:

if (primitive# = ARC_CLW#) OR (primitive# = ARC_CCLW#) then

' Если дуга лежит в плоскости XY выводятся I и J:

if plane# = PLANE_XY# then
write id1# using "I%.15g", i*RDISCR
write id1# using "J%.15g", j*RDISCR
```

```
' Иначе, если дуга лежит в плоскости YZ выводятся J и K:

elseif plane# = PLANE_YZ# then

write id1# using "J%.15g", j*RDISCR
write id1# using "K%.15g", k*RDISCR

' Иначе, если дуга лежит в плоскости ZX выводятся I и K:

elseif plane# = PLANE_ZX# then

write id1# using "I%.15g", i*RDISCR
write id1# using "K%.15g", k*RDISCR
endif

endif

' Иначе, если анализируемая команда перемещения выполняется в
' подпрограмме, то текущие координаты выводятся в приращениях

elseif proc_body# then

' при ненулевом перемещении по оси X:
if dx<>0.0 then

' Выводится новое значение dx:
write id1# using "X%.15g", dx*RDISCR
endif

' при ненулевом перемещении по оси Y:
if dy<>0.0 then
' Выводится новое значение dy:
write id1# using "Y%.15g", dy*RDISCR
endif

' при ненулевом перемещении по оси Z:
if dz<>0.0 then
' Выводится новое значение z:
write id1# using "Z%.15g", dz*RDISCR
endif

' Для круговой интерполяции вывод по адресам I, J, K:
if (primitive# = ARC_CLW#) OR (primitive# = ARC_CCLW#) then

' Если дуга лежит в плоскости XY выводятся I и J:
if plane# = PLANE_XY# then
write id1# using "I%.15g", di*RDISCR
write id1# using "J%.15g", dj*RDISCR

' Иначе, если дуга лежит в плоскости YZ выводятся J и K

elseif plane# = PLANE_YZ# then
write id1# using "J%.15g", dj*RDISCR
write id1# using "K%.15g", dk*RDISCR

' Иначе, если дуга лежит в плоскости ZX выводятся I и K
elseif plane# = PLANE_ZX# then
```

```
write id1# using "I%.15g", di*RDISCR
write id1# using "K%.15g", dk*RDISCR
endif

endif

endif

' если на анализируемом перемещении
' изменился режим коррекции длины инструмента
' (т.е. параметр Icomp анализируемой команды
' не равен аналогичному параметру предшествующей
' команды lcomp_#):

if (Icomp <> lcomp_#) then

' Для включения коррекции
' вывод корректора
' по адресу T:

if Icomp<>LCOMP_OFF# then
write id1# using "T%.2d", cutter_l_crtr#

' Иначе, для выключения коррекции на длину инструмента
' вывод T00:

elseif Icomp = LCOMP_OFF# then
 write id1# using "T00"

endif

endif
```

```
'При изменении величины подачи
'(т.е. параметр f анализируемой команды
'не равен аналогичному параметру предшествующей
'команды f_)
'вывод подачи по адресу F:

if f<>f_ then
write id1# using "F%.15g", f
endif

write id1# using "%s", NL$ ' Завершение кадра - символ LF

endif

LOOP

' конец описания и конец файла SMPL3.PSS
'*******************************************************
```