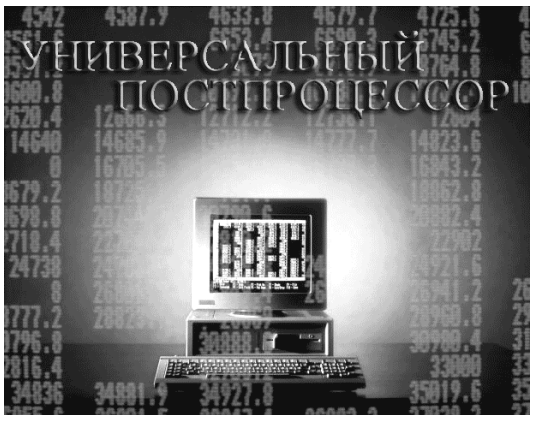


Глава 6

Исходная (APT) программа системы ГеММа-3D



Исходная (APT) программа системы ГеММа-3D

Формат исходной программы системы ГеММа-3D

Исходная программа системы ГеММа-3D представляет собой последовательность команд, называемых APT-командами, описывающими поведение рабочего органа (инструмента) станка с ЧПУ.

Исходная программа системы ГеММа-3D состоит из:

- команды PARTNO (см. далее), с параметрами - целочисленным номером программы и следующим за ним текстом комментария - заголовка APT-программы;
- подпрограмм (от 0 до 50);
- тела главной программы;
- команды завершения APT-программы END или FINI.

Таким образом, структура APT-программы имеет следующий вид:

```
заголовок APT-программы (команда PARTNO)
подпрограмма 1
подпрограмма 2
...
тело главной программы
команда END или FINI
```

Подпрограммы описывают типовые действия рабочего органа станка, которые могут быть выполнены посредством вызова подпрограммы из главной программы или других подпрограмм. Подпрограмма состоит из:

- команды PROCEDURE (см. далее) с параметром - целочисленным номером подпрограммы - заголовка подпрограммы;
- тела подпрограммы;
- команды возврата из подпрограммы RETURN.

Общая структура подпрограммы:

```
заголовок подпрограммы (команда PROCEDURE)
тело подпрограммы
команда RETURN
```

Тело главной программы или подпрограммы - это последовательность технологических и геометрических APT-команд которые транслируются постпроцессором в последовательности управляющих кодов для конкретных систем ЧПУ. Описание возможных APT-команд приведено в следующем разделе.

Команды исходной программы системы ГеММа-3D

Каждая APT-команда начинается с новой строки и представляет собой ключевое слово, обозначающее данную команду, за которой следуют параметры этой команды. Количество, тип и смысл параметров APT-команды различны. Все линейные размеры параметров задаются в мм, а угловые - в градусах, если это не оговаривается особо.

Пример структуры APT-команды:

```
SPINDL/<CLW | CCLW >,< обороты_шпинделя>
```

Здесь:

SPINDL-ключевое слово команды;

в первых угловых скобках заключены разделённые вертикальной чертой альтернативные варианты первого параметра;

< обороты_шпинделя> - фактическое значение второго параметра.

В синтаксисе ряда команд могут иметь место конструкции типа:

[, параметр_цикла_0 ... [, параметр _цикла_11]] - в квадратные скобки заключаются необязательные элементы команд. Многоточие обозначает возможность включения в команду нескольких таких элементов.

PARTNO - команда заголовка APT-программы

Синтаксис

```
PARTNO/ номер _программы текст_комментария
```

Описание

Заголовок APT-программы. Должен быть первым оператором APT-программы.

При анализе команды инвариантным постпроцессором параметр command# получает значение HEADER. Значения номер_программы и текст_комментария присваиваются параметрам partno# и partno_text$ соответственно. Соответствующие значения получают также параметры proc_body#, proc_off#, prog_body#, prog_off#. Остальным параметрам анализируемой команды инвариантный постпроцессор присваивает значения по умолчанию.

REMARK - команда комментария APT-программы

Синтаксис

```
REMARK/текст_комментария
```

Описание

Коментарий APT-программы. При анализе APT-программы значение <текст_комментария> получает переменная comment$.

END, FINI - команда конца APT-программы

Синтаксис

```
FINI или END
```

Описание

Обозначают конец APT-программы. Команды исходной программы, следующие за командами END или FINI, игнорируются.

При обработке команды инвариантный постпроцессор устанавливает значение параметра command# равным END#. Соответствующие значения получают также параметры proc_body#, proc_off#, prog_body#, prog_off.

PROCEDURE - команда заголовка подпрограммы

Синтаксис

```
PROCEDURE/ номер_подпрограммы
```

Описание

Заголовок подпрограммы. Подпрограмма вызывается из главной программы или другой подпрограммы с помощью команды CALL# с указанием в вызове значения номер подпрограммы.

При анализе команды инвариантным постпроцессором параметр command# получает значение PROCEDURE#. Целочисленное значение **<номер_подпрограммы>** при этом присваивается параметру proc_no#. Соответствующие значения получают также параметры proc_body#, proc_off#, prog_body#, prog_off#. Остальным параметрам анализируемой команды инвариантный постпроцессор присваивает значения по умолчанию.

RETURN - команда возврата из подпрограммы

Синтаксис

```
RETURN
```

Описание

Команда возврата из подпрограммы. Следующей выполняемой командой исходной программы будет команда, следующая за командой CALL#, вызвавшей данную подпрограмму.

При анализе команды инвариантным постпроцессором параметр command# получает значение RETURN#. Соответствующие значения получают также параметры proc_body#, proc_off#, prog_body#, prog_off#.

ORIGIN - команда привязки станочной системы координат к математической

Синтаксис

```
ORIGIN/ x0, y0, z0
```

Описание

Команда определяет положение нулевой точки станочной системы координат относительно математической системы координат. Используется для установления соответствия программы и модели. Значения x0, y0, z0

присваиваются параметрам xxm , yym , zzm соответственно.

BLOCK - команда определения размера заготовки

Синтаксис

```
BLOCK/ x0 , dx , y0 , dy , z0 , dz
```

где x0 - x координата угловой точки, dx - размер заготовки по x

y0 - y координата угловой точки, dy - размер заготовки по y

z0 - z координата угловой точки, dz - размер заготовки по z

Команда задает положение заготовки и ее размеры в станочной системе координат. Таким образом команда ORIGIN также влияет на параметры x0,y0,z0

TOOLNO, LOADTL - команды загрузки инструмента

Синтаксис

```
TOOLNO/ номер_инструмента
```

или

```
LOADTL/ номер_инструмента
```

Описание

Команда выбора (загрузки в шпиндель) инструмента с номером **<номер _инструмента>**.

При анализе команды инвариантным постпроцессором параметр command# получает значение LOAD_TOOL#. Целочисленное значение **<номер_инструмента>** присваивается параметру pass_cutter_no. Номер прохода pass_no# увеличивается на 1. Соответствующие значения получают также параметры pass_first#, pass_last#, prev_pass_no#, prev_cutter_no#, next_pass_no#, next_cutter_no#.

CUTTER - команда определения параметров инструмента

Синтаксис

```
CUTTER/ диаметр_наружный, радиус_скругления,
длина_инструмента.
```

Описание

Сообщает постпроцессору геометрические характеристики нового инструмента.

Инвариантный постпроцессор присваивает значения параметрам: cutter_diametr = **<диаметр_наружный>**, cutter_radius = **<диаметр_наружный/2>**, cutter_rf = **<радиус_скругления>**, cutter_length = **<длина_инструмента>** при обработке команды смены инструмента, следующей за командой CUTTER.

CUTTER5 - команда параметров инструмента

Синтаксис

```
CUTTER5/ диаметр_наружный, радиус_скругления, диаметр_торца,
конусность (в градусах), длина_инструмента.
```

Описание

Сообщает постпроцессору геометрические характеристики нового инструмента.

Инвариантный постпроцессор присваивает значения параметрам: cutter_diametr = **<диаметр_наружный>**, cutter_radius = **<диаметр_наружный/2>**, cutter_rf = **<радиус_скругления>**, cutter_length = **<длина_инструмента>**, cutter_conus = **<конусность>**, cutter_dt = **<диаметр_торца>** при обработке команды смены инструмента, следующей за командой CUTTER5.

FEDRAT - команда установки значения подачи перемещения

Синтаксис

```
FEDRAT/ подача_перемещения
```

Описание

Устанавливает новое значение подачи (скорости) движения. Отменяет действие команды RAPID.

Инвариантный постпроцессор присваивает значение **<подача_перемещения>** параметру f при обработке первого перемещения, следующего за командой FEDRAT.

RAPID - команда включения режима быстрых перемещений (позиционирования)

Синтаксис

```
RAPID
```

Описание

Объявляет, что следующие за командой RAPID перемещения должны выполняться в режиме позиционирования. Отменяет действие команды FEDRAT.

SPINDL - команда управления вращением шпинделя.

Синтаксис

Команда включения шпинделя :

```
SPINDL/ { CLW | CCLW }, обороты_шпинделя
```

Команда выключения шпинделя:

```
SPINDL/OFF
```

Описание

Устанавливает новое значение оборотов шпинделя и режим вращения шпинделя.

Инвариантный постпроцессор присваивает значение обороты_шпинделя параметру s и значение SPIN_CLW# или SPIN_CCLW# параметру sm# при обработке первого перемещения, следующего за командой включения шпинделя. В результате команды выключения шпинделя, значение параметра sm# устанавливается равным SPIN_OFF.

COOLNT - команда управления режимом охлаждения

Синтаксис

Команда включения охлаждения:

```
COOLNT/ ON
```

Команда выключения охлаждения:

```
COOLNT/ OFF
```

Описание

Изменяет режим охлаждения станка.

Инвариантный постпроцессор присваивает параметру cool# значение ON# при обработке первого перемещения, следующего за командой включения охлаждения, и OFF# после команды выключения охлаждения.

MODE - команда режима обработки.

Синтаксис

```
MODE/ { MILL | TURN | EDM | PUNCH | NIBBLING | CUTTING}
MODE/ {CORR_3D | MILL_5D}
MODE/ { NIBBLING} ,   шаг высечки
```

Описание

Устанавливает новое значение режима обработки.

Инвариантный постпроцессор присваивает ниже перечисленные значения параметру mode#.

MILL# - фрезерование;

TURN# - точение (токарная обработка);

EDM# - электроэрозионная обработка;

PUNCH# - пробивка (вырубной пресс);

NIBBLING# - высечка (вырубной пресс);

CUTTING# - резка;

CORR_3D# - проход сформирован как фрезерный и содержит информацию для 3D коррекции;

MILL_5D# - пяти-координатная фрезерная обработка;

<шаг высечки> - параметр, определяющий шаг высечки.

ROTATE - команда поворота стола

Синтаксис

```
ROTATE/ ось_поворота x, угол_поворота x, ось_поворота y,
угол_поворота y, ось_поворота z, угол_поворота z
```

Описание

При анализе команды инвариантным постпроцессором параметр command# получает значение ROTATE#. Параметр axis# устанавливается равным A_AXIS# (ось X), B_AXIS# (ось Y), C_AXIS# (ось Z). Значения **<угол_поворота>** присваивается одному из параметров aang (ось X), bang (ось Y), cang (ось Z).

ORIGNO - команда определения номера станочной системы координат

Синтаксис

```
ORIGNO/ номер станочной системы координат
```

Описание

Команда установки / смены станочной системы координат.

При анализе команды инвариантным постпроцессором параметр command# получает значение ORIGNO#. Номер системы координат заносится в параметр origno#.

FROM - команда задания координат начальной точки

Синтаксис

```
FROM/ x0, y0, z0 [,u0 [,v0 [,w0] ] ]
```

Описание

Команда установки значений координат текущей точки траектории рабочего органа станка относительно нового начала координат. Начало координат смещается таким образом, что координаты текущей точки становятся равными значениям, заданным в команде. Значения x0, y0, z0 задают декартовы координаты текущей точки в новой системе. Значения u0, v0, w0 задают новые значения для дополнительных 4-й, 5-й, 6-й координат и являются необязательными. Если эти значения не заданы, они считаются не изменившимися. При анализе команды инвариантным постпроцессором параметр command# получает значение SET_ORIGIN. Значения x0, y0, z0, u0, v0, w0 присваиваются параметрам x0, y0, z0, u0, v0, w0. Параметры x, y, z, u, v, w устанавливаются равными новым значениям x0, y0, z0, u0, v0, w0. Соответствующим образом изменяются значения параметров shift_x0, shift_y0, shift_z0, shift_u0, shift_v0, shift_w0.

GOTO - команда линейного перемещения в точку

Синтаксис

```
GOTO/ x, y, z [,u [,v [,w] ] ]
```

Описание

Команда перемещения по отрезку прямой из текущей точки траектории рабочего органа станка в точку с декартовыми координатами x, y, z. Значения u, v, w задают новые значения для дополнительных 4-й, 5-й, 6-й координат и являются необязательными. Если эти значения не заданы, они считаются не изменившимися.

При анализе команды инвариантным постпроцессором параметр command# получает значение MOVE#. Параметр primitive# устанавливается равным LINE#, если командой RAPID не был установлен режим быстрых перемещений, и LOCATE# в противном случае. Значения x, y, z, u, v, w присваиваются параметрам x, y, z, u, v, w. Параметры dx, dy, dz. du, dv, dw становятся равными разности новых и старых значений соответствующих координат. Вычисляются новые значения суммарных импульсов по каждой из координат imptot_x, imptot_y, imptot_z, imptot_u, imptot_v, imptot_w. Вычисляются касательные векторы к данному перемещению, устанавливаются признаки режимов коррекции.

ARCSLP - команда перемещения в точку по дуге окружности

Синтаксис

```
ARCSLP/ плоскость, направление, x,y,z,xc,yc,r[,u [,v[,w]]]
```

Описание

Команда перемещения по окружности радиуса r из текущей точки траектории рабочего органа станка в точку с декартовыми координатами x, y, z. Значения u, v, w задают новые значения для дополнительных 4-й, 5-й, 6-й координат и являются необязательными. Если

эти значения не заданы, они считаются не изменившимися. Смысл значений xc, yc зависит от типа плоскости:

XY - координаты x, y центра окружности, лежащей в плоскости XY;

YZ - координаты y, z центра окружности, лежащей в плоскости XZ ;

ZX - координаты z, x центра окружности, лежащей в плоскости ZX.

При анализе команды инвариантным постпроцессором параметр command# получает значение MOVE#. Параметр primitive# устанавливается равным ARC_CCLW#, если значение **<направление>** равно CCLW (движение против часовой стрелки) или ARC_CLW#, если значение **<направление>** равно CLW (движение по часовой стрелке). Если в описании постпроцессора параметр emul_q# установлен в TRUE#, постпроцессор разбивает дугу по квадрантам и анализирует перемещение в каждом квадранте по отдельности. Если радиус дуги больше, чем заданная в описании постпроцессора величина RMAX, постпроцессор заменяет дугу серией аппроксимирующих ее линейных перемещений.

Значения x, y, z, u, v, w присваиваются параметрам x, y, z, u, v, w. Параметры i, j, k и di, dj, dk устанавливаются равными координатам центра окружности и смещениям по координатным осям центра окружности относительно начальной точки дуги. Параметры dx, dy, dz. du, dv, dw становятся равными разности новых и старых значений соответствующих координат.

Вычисляются новые значения суммарных импульсов по каждой из координат imptot_x, imptot_y, imptot_z, imptot_u, imptot_v. imptot_w. Вычисляются касательные векторы к данному перемещению, устанавливаются признаки режимов коррекции.

STOP# - команда останова

Синтаксис

```
STOP
```

Описание

Объявляет, что данная точка траектории рабочего органа станка есть точка останова.

При анализе команды инвариантным постпроцессором параметр command# получает значение STOP#.

OPSTOP# - команда условного останова

Синтаксис

```
OPSTOP
```

Описание

Объявляет, что данная точка траектории рабочего органа станка есть точка условного останова.

При анализе команды инвариантным постпроцессором параметр command# получает значение OPSTOP#.

SYNCTR - команда синхронизации при движении по двум плоскостям (4D электоэрозионная обработка)

Синтаксис

```
SYNCTR/ { UP | DOWN }
```

Описание

Команда определяет к какой из двух рабочих плоскостей будет относится следующая за ней команда движения (верхней или нижней) .

CUTCOM - команда включения или выключения коррекции длины инструмента

Синтаксис

Включение коррекции длины инструмента:

```
CUTCOM/ LENGTH, { PLUS | MINUS }, номер_корректора
```

Выключение коррекции длины инструмента:

```
CUTCOM / LENGTH, OFF
```

Описание

Объявляет, что на следующем за командой перемещении инструмента должна быть включена (выключена) коррекция длины инструмента.

Инвариантный постпроцессор присваивает значение **<номер_корректора>** параметру cutter_l_crtr# при включении коррекции для обработки первого перемещения, следующего за командой включения коррекции длины инструмента. Значение параметра lcomp# устанавливается равным LCOMP_UP# или LCOMP_DOWN# при использовании модификаторов PLUS или MINUS соответственно. В результате команды выключения коррекции, значение параметра lcomp# устанавливается равным LCOMP_OFF#.

CUTCOM - команда включения или выключения коррекции радиуса инструмента

Синтаксис

Включение коррекции радиуса инструмента:

```
CUTCOM/ RADIUS, { LEFT | RIGHT }, номер_корректора
```

Выключение коррекции радиуса инструмента:

```
CUTCOM/ RADIUS, OFF
```

Описание

Объявляет, что на следующем за командой перемещении инструмента должна быть включена (выключена) коррекция радиуса инструмента.

Инвариантный постпроцессор присваивает значение **<номер_корректора>** параметру cutter_r_crtr# при включении коррекции для обработки первого перемещения, следующего за командой включения коррекции радиуса инструмента. Значение параметра rcomp# устанавливается равным RCOMP_LEFT# или RCOMP_RIGHT# при включении соответственно левой/правой коррекции. В результате команды выключения коррекции, значение параметра rcomp# устанавливается равным RCOMP_OFF#. Соответствующие значения получают также признаки обработки контура в режиме коррекции.

TOOLRIGHT - команда включения правой коррекции радиуса инструмента

Синтаксис

```
TOOLRIGHT/ номер_корректора
```

Описание

Полностью эквивалентна команде CUTCOM/ RADIUS, RIGHT, номер_корректора

TOOLLEFT - команда включения левой коррекции радиуса инструмента

Синтаксис

```
TOOLLEFT/ номер_корректора
```

Описание

Полностью эквивалентна команде CUTCOM/ RADIUS, LEFT, номер_корректора

TOOLON - команда включения коррекции радиуса инструмента

Синтаксис

```
TOOLON
```

Описание

Полностью эквивалентна команде CUTCOM/ RADIUS, OFF#

CALL - команда вызова подпрограммы

Синтаксис

```
CALL/ номер_подпрограммы
```

Описание

Обозначает обращение к подпрограмме с номером **<номер_подпрограммы>**. При анализе команды инвариантным постпроцессором параметр command# получает значение CALL#. Целочисленное значение **<номер_подпрограммы>** присваивается параметру unit_call#.

CYCLE - команда задания параметров постоянного цикла

Синтаксис

Задание параметров цикла:

```
CYCLE/ номер_цикла[,параметр_цикла_0...[,параметр_цикла_11]]
```

Отмена цикла:

```
CYCLE/OFF или CYCLE /0
```

Описание

Команда задает параметры постоянного цикла, которые будут использоваться постпроцессором при последующем анализе команды DOCYCLE. В команде может быть задано вплоть до 12-ти различных значений **<параметр_цикла>**

При анализе команды инвариантный постпроцессор присваивает параметру command# значение CYCLE#, параметру cycle# значение **<номер_цикла>**. Значения параметров cyc1e_p0,...,cyc1e_p11 устанавливается равными вещественным значениям **<параметр_цикла_0>,...,<параметр_цикла_11>** соответственно. Если эти значения не заданы, они считаются не изменившимися. В результате команды отмены цикла значение параметра cycle# устанавливается равным 0.

DOCYCLE - команда выполнения цикла

Синтаксис

```
DOCYCLE
```

Описание

Команда выполнения постоянного цикла с параметрами, заданными ранее командой CYCLE. При анализе команды инвариантным постпроцессором

параметр command# получает значение DO_CYCLE#. Значения параметров cycle#, cycle_p0, ..., cycle_p11 должны быть предварительно заданы командой CYCLE.

INSERT, PREFUN - команды вставки в текст управляющей программы

Синтаксис

```
INSERT/ последовательность_кодов
```

или

```
PREFUN/ последовательность_кодов
```

Описание

Задает последовательность символов, предназначенную для непосредственной вставки в текст управляющей программы. При анализе любой из последующих команд, параметр literal$ будет содержать строку символов < **последовательность_кодов**>. Начиная с версии 8.0, команда генерирует соответствующую команду постпроцессора .INSERT#

DELAY - команда останова на заданное время

Синтаксис

```
DELAY/ время_останова
```

Описание

Производит останов на заданное время. Инвариантный постпроцессор присваивает параметру command# значение DELAY#, параметру td# значение < **время_останова**>.

Z_PLANE - команда задания высот верхней и нижней управляющих плоскостей (4D электоэрозионная обработка)

Синтаксис

```
Z_PLANE/   Z нижней плоскости , Z верхней плоскости
```

Описание

Задает высоты верхней и нижней управляющих плоскостей при электроэрозионной обработке с наклоном проволоки.

FLUSH - команда задания параметра промывки (4D электоэрозионная обработка)

Синтаксис

```
FLUSH/ параметр промывки
```

Описание

Задает параметр промывки - flush#

GENRTR - команда задания параметра генератора (4D электоэрозионная обработка)

Синтаксис

```
GENRTR/ параметр генератора
```

Описание

Задает параметр генератора - genrtr#

EDM_ANGLE - команда задания постоянного угла наклона проволоки и номера углового корректора (2D электоэрозионная обработка)

Синтаксис

```
EDM_ANGLE/ угол наклона проволоки,  номер углового корректора
```

Описание

Задает значение угла наклона проволоки - edm_angle# и номер задающего углового корректора - edm_ang_corr#

Считается, что угловая коррекция включается вместе с коррекцией на радиус инструмента.

PUNCH - команда задания числа ударов на линейном или круговом перемещении (Пробивка)

Синтаксис

```
PUNCH/ число ударов
```

Описание

Параметр задает число ударов пуансоном на перемещении.

Действует только на следующее за собой перемещение.