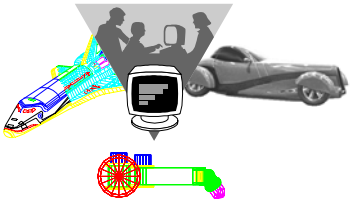
საქართველოს ტექნიკური უნივერსიტეტი

მაგისტრატურის დეპარტამენტი,
მანქანათმშენებლობის ტექნოლოგიური პროცესების
ოპტიმიზაციის ლაბორატორია,
CAD/CAM-ის ჯგუფი

ParametricCAD'97

მოხსენებათა კრებული

(სამეცნიერო კონფერენციის მოხსენებათა
კრებული, 3-4 ივლისი, 1008 წელი)



თბილისი 1008 წელი

Разработка модуля инвариантного постпроцессирования управляющих программ в САПР ТП

аспирант Н. Долидзt

Эффективность компьютеризированных производственных систем (КПС) в большой степени зависит от уровня и качества решения проблемы автоматизации програмирования систем числового програмного управления (СЧПУ) для различных видов технологического оборудования - токарных, фрезерных и других станков.

Числовое програмное управление (ЧПУ) это процесс во время которого происходит управление технологическим процессом производства посредством управлениz специальных числовых кодов,

последовательность которых образуют так называемую управляющую программу (УП). Команды УП определяют траекторию движения и режимы работы исполнительных органов технологического оборудования.

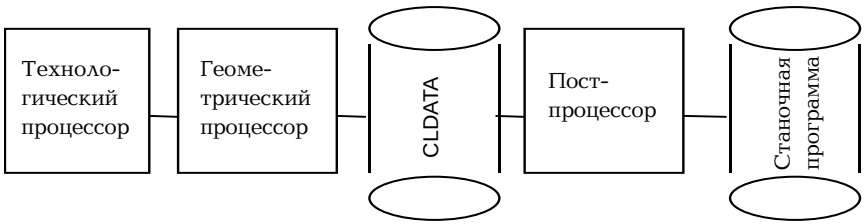
Создание УП осуществляется на этапе технологической подготовки производства при помощи специальных систем автоматизированного проэктирования (САПР УП).

В состав САПР УП входит три основных модуля:

1. *Технологический процессор*
2. *Геометрический процессор*
3. *Постпроцессор.*

Взаимосвязь описанных модулей представлена на рисунке.

Технологический процессор предназначен для



определения параметров режима работы испоолнительных органов технологического оборудования, таких как например: скорость перемещения, количество оборотов шпинделя и т. д.

CLDATA (Cutter Location Data) является промежуточным уровнем описания УП, который представляет собой массив содержащий последовательность логических записей переменной длины. Каждая запись это последовательность логических слов содержащая информацию определенного вида, например: координаты, определяющие траекотрию движения инструмента, указатели технологических режимов обработки, канонические формы геометрических элементов контура, сведения о допусках, о режущем инструменте и т. д.

Станочная управляющая программа представляет собой второй уровень описаниия УП для конкретного станка с ЧПУ.

Постпроцессор является пограмням модулем предназначенным для обработки записей CLDATA и выдающем на выходе станочную программу для каждого отдельного устройства ЧПУ, кроме того в функции постпроцессора входит осуществление т. н. обратной компиляции из станочной программы в CLDATA, что позволяет верифицировать визуальными средствами САПР УП, уже инеющуюся станочную программу.

Хотя международным стандартом ISO и регламентированны определенные правила кодирования числовой информации в станочных УП, тем не менее ISO стандарты не всегда соблюдаются.

Станочную программу записывают в виде последовательности кадров (содержание кадров станочной программы также регламентированно). Информационные слова в кадре обычно записывают в следующей последовательности:

1. Подготовительная функция - (G);

2. Размерные перемещения по различным координатам - (X,Y,Z,U,V,W,P,Q,A,B,C);

3. Параметры интерполяции или шаг резьбы - (I,J,K);

4. Функция подачи - (F);

5. Функция главного движения - (S);

6. Функция инструмента - (T);

7. Вспомогательная функция - (M);

Кроме того , в разныых устройствах ЧПУ некоторые команды имеют одинаковое функциональное назначение. Как правило, это команды подготовительной (группа G) и вспомогательной (группа M) функций, например:

G00 быстрый ход
G01 линейная интерполяция
G02 круговая интерполяция по часовой стрелке
G03 круговая интерполяция против часовой стрелке
G90 ввод абсолютного значения
G91 ввод относительного значения
G94 подача на мм/мин
G95 подача на мм/об
M03 вращение шпинделя по часовой стрелке
M04 вращение шпинделя против часовой стрелке
M05 останов шпинделя.

Остальные команды разных устройств ЧПУ могут отличатся не только форматом записи, но и назначением. Кроме того команды обозначающие определенное действие у одних УЧПУ могут совершенно отсутствовать у других устройств ЧПУ.

Внизу приведены примеры демонстрирующие различия командных слов в следующих устройствах ЧПУ:

- *Sinumerik 7T*
- *CNC - Y*
- *Hunor PNC 712*
- *Электроника НЦ 31-01.*

F - функция подачи

Sinumerik 7T	CNC-Y	Hunor PNC 712	YW31
F023	F05	F1.3	F06

Данная таблица демонстрирует разность форматов записи функции подачи у разных УЧПУ, например: у Sinumerik 7T формат F023, lfyyjt означает что во время записи функции подачи после F командного слова пишется численное значение параметра, где целое число не должно

превышать 2 еденицы, а дробная часть - 3 еденицы, кроме того 0 обозначает что в случае если целое число это целое двузначное число то на месте сотенного числа пишется 0 (напр: F01.706).

M02 - конец программы

Как видно из таблицы у Sinumerik 7T, CNC - Y и Электроника НЦ 31-01 функция M02 определяет конец программы, но эта же самая функция отсутствует у Hunor PNC 712.

Подготовительная функция G60

В таблице демонстрируется разное функциональное назначение подготовительной функции G60:

Sinumerik 7T	CNC-Y	Hunor PNC 712	YW31
nj xyjcdf yj d	jncandetn	jncandetn	jncandetn

функция G60 определяет точный останов

- *CNC - Y и Электроника НЦ 31-01 данная функция отсутствует*
- *Hunor PNC 712 - подготовительная функция G60 имеет другое функциональное назначение и определяет организацию цикла.*

M06 - смена инструмента

Функциональное назначение M06 функции это смена инструмента. Данная функциа может отсутствовать у некоторых УЧПУ как например она отсутствует у CNC - Y, Hunor PNC 712 и Электроника НЦ 31-01.

M46 - работа с инструментальным магазином

•

Как видно из таблицы М46 вспомогательная функция отсутствует у Sinumerik 7T, Hunor PNC 712 и Электроника 31-01.

Так как, основной задачей постпроцессора является создание станочной программы, то различие команд в разных устройствах ЧПУ порождает проблему постпроцессоров для разных САПР УП, состоящую в том, что каждое новое технологическое оборудование требует разработки соответствующего постпроцессорного модуля.

Решение данной проблемы возможно двумя способами:

Создание новых постпроцессорных модулей разработчиками САПР УП. Такое решение характерно для САПР УП с жесткой архитектурой, при этом адаптация програмно-математического обеспечения (ПМО) на конкретный парк УЧПУ пользователя не может беть осуществленна самим пользователем.

Настройка постпроцессорного модуля на конкретное технологическое обборудование самим пользователем, что соответствует концепции САПР УП с открытой архитектурой.

Создание САПР УП с открытой архитектурой приводит к необходимости разработки инвариантного постпроцессора, настраиваемиго на конкретную модель УЧПУ. Однако применение такого инвариантного постпроцессора снимает проблему в рамках определенных САПР УП. Как обычно САПР УП отличаются правилами описания промежуточного языка CLDATA. Несмотря на определенные унификации промежуточные данные CLDATA различных САПР УП (например: АРТ, MODAPT, ЕХАРТ, 2CL, ТЕХТРАН, АПТ ЕС, АПТ СМ, САПС - 9 И ДР.) имеют свои особенности: количество слов в записи, порядок описания, разность форматов, разность кодов; в результате возникает необходимость разработки собственного инвариантного постпроцессора для конкретной САПР УП.

Исходя из вышесказанного, возникла идея создания такого инвариантного постпроцессора, который представлял бы из себя независимый модуль адаптируемый к различным САПР УП в базе данных которой будут заложены модифицируемые данные о различных устройствах ЧПУ и о CLDATA различных САПР УП.

Как видно из следующего рисунка модуль инвариантного постпроцессора выделен из САПР УП и служит связующим звеном между САПР УП и группой УЧПУ.

Процесс разработки инвариантного постпроцессора можно разделить на следующие основные этапы:

1. *Разработка методики компиляции станочной пограммы.*
2. *Разработка методики компиляции CLDATA из станочной пограммы.*
3. *Разработка базы данных инвариантного постпроцессора.*

Расмотрим вышесказанные этапы подробно.

Методика *компиляции станочной пограммы может быть представлена в виде следующих последовательных шагов, которые позволяют считывать и идентифицировать запис CLDATA и в итоге формировать станочную программу:*

1. *Считывание записи CLDATA*
2. *Идентификация формата CLDATA*
3. *Определение функционального назначения записи CLDATA*
4. *Идентификация параметров и вспомогательных*

слов CLDATA

5. Идентификация продолжения записи CLDATA

6. Нахождение соответственной станочной функции

7. Определение формата записи слов станочной программы

8. Определение последовательности слов в станочной программе

9. Формирование параметров адресных слов станочной программы

10. Запись кадра станочной пограммы

11. Идентификация наличия следующей записи CLDATA .

12. True: переход на пункт 1.

Методика *компиляции CLDATA из станочной пограммы состоит из следующих шагов:*

1. Считывание командного слова в кадре станочной пограммы.

2. Идентификация формата командных слов станочной пограммы.

3. Определение функционального назначения командного слова станочной пограммы.

4. Нахождение соответственного функционального назначения записи CLDATA.

5. Определение формата записи CLDATA.

6. Идентификация параметров и вспомогательных слов CLDATA.

7. Формирование записи CLDATA.

8. Идентификация формирования многострочной записи

9. True: переход на пункт 1.

Разработку базы данных инвариантного постпроцессора упрощенно можно разделить на два основных этапа:

- *Разработка концептуальной модели базы данных.*
- *Разработка и создание физической модели базы данных.*

Во время разработки концептуальной модели базы

данных было расмотренно и апробированно несколько подходов.

На ранней стадии *разработки концептуальной модели возникла идея создания информационного массива для разных групп командных слов всех моделей УЧПУ, например: информационный массив "Подготовительная функция" должен иметь следующие поля:*
{Функция} {код УЧПУ} {код CLDATA} {под-код CLDATA}
Где:

- *Функция - Означает командное слово*
- *Код УЧПУ - Код присвоенный конкретному станку*
- *Код CLDATA - Главный код CLDATA*
- *Под-код CLDATA - Код вспомогательного слова CLDATA.*

Кроме того существует еще один информационный массив связывающий имя УЧПУ с кодом станка, со следующими полями:

{Станок} {Код станка}

Где :

- *Станок - Означает имя УЧПУ*
- *Код станка - имя конкретного УЧПУ употребляемый в массивах командных слов*

При данном подходе, для того чтобы в результате компиляции получить станочную программу в постпроцессоре следует жестко определить последовательность чтения информационных массивов, например:

1. *Номер кадра;*
2. *Массив подготовительной функции;*
3. *Массив размерных перемещений;*

и т. д.

Кроме того, количество информационных массивов небольшое, но сами массивы довольно большого объема так как они содержат информацию о всех моделях УЧПУ.

Данный метод также потиворечит разработанной методике компиляции станочной пограммы из CLDATA, например:

1. Не учитывается модальность командных слов (модальное командное слово деиствует и на последующий кадр), так как отсутствуют соответственные информационные поля, которые учитывают модальность функций.
2. Не возможно идентификация формата CLDATA (отсутствует информационное поле учитывающее формат CLDATA).
3. Отсутствует идентификация параметров и вспомогательных слов (отсутствуют информационные поля описывающие параметры и вспомогательные слова).
4. Отсутствует идентификация продолжения записи (отсутствуют соответственные информационные поля).

Другая отрицательная сторона разработанной концептуальной модели это отсутствие возможности компиляции CLDATA из станочной УП, т. е. Не учтена разработанная методика компиляции, так как отсутствуют информационные поля полностью описывающие CLDATA.

На втором этапе *разработки в отличие от предыдущей концептуальной модели было решено создать отдельный информационный массив для каждого отдельного УЧПУ, содержащий описания каждого информационного слова данного оборудования. Такой информационный массив должен иметь следующие поля:*
{Код информационного слова} {Псевдо-код} {Информационное слово}
Где:

- *Код информационного слова - определяет принадлежность последующих командных слов к определенной группе;*

yfghbvth: G - принадлежность к подготовительным функциям

- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ;*
- *Информационное слово - командное слово УЧПУ, напр. G00*

Кроме того существует информационный массив связывающий псевдо-коды с кодами CLDATA, со

следующими полеми:

{Псевдо-код}{код CLDATA}

Где :

- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ;*
- *код CLDATA - код главного слова CLDATA.*

Третий информационный массив связывает псевдо-коды со словами АПТ и имеет следующие поля:

{Псевдо-код} {слово АПТ}

Где:

- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ;*
- *Слово АПТ - информационное слово определяющее функциональное назначение командного слова или записи CLDATA;*

При данном подходе опять не учитывается модальность функций и разработанные меодики компиляции CLDATA из станочной УП и станочной УП из CLDATA, так как отсутствуют соответственные информационные поля а также увеличивается количество информационных массивов.

<u>На третьем этапе</u> *разработки концептуальной модели базы данных в информационный массив УЧПУ было добавленно несколько информационных полей. В результате информационный массив имеет следующие поля:*

{Код информационного слова}

{Псевдо-код} {Модальность} {Формат} {Информац. слово с форматом}

Где:

- *Код информационного слова - определяет принадлежность последующих командных слов к определенной группе*
- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ*
- *Модальность - определяет модальность данного информационного слова, напр. 0-не*

модальный, 1-модальный

- *Формат - определяет существование параметров у данного командного слова*
- *Информационное слово с форматом - определяет само командное слово и его формат если таковой существует;*

Информационный массив для CLDATA имеет следующие поля:

{Код CLDATA} {Под код CLDATA} {Псевдо-код}

Где:

- *Код CLDATA - код главного слова CLDATA*
- *Под код CLDATA - код вспомогательного слова CLDATA*
- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ.*

Данный метод не учитывает разработанную методику компиляции CLDATA из станочной УП, слова УЧПУ.

Данный метод не учитывает разработанную методику компиляции CLDATA из станочной УП, так как:

1. *Не определяется формат записи CLDATA*
2. *Не идентифицируются параметры слов CLDATA.*

В итоге невозможно сформировать запись CLDATA, поэтому возникла потребность сохранить дополнительную информацию о CLDATA и как результат модифицировать массив CLDATA.

Исходя из сказанного на следующем этапе разработки поля массивов каждого УЧПУ остаются без изменений, но добавляеься новый информационный массив связывающий псевдо-код со словами АПТ. Этот массив имеет следующие поля:

{Псевдо-код} {слово АПТ}

Где:

- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ*
- *Слово АПТ - Командное слово АПТ.*

Как сказано выше информационный массив CLDATA модифицируется и имеет следующие поля:

Псевдо-код = {(код инфор. слова, место А, формат А, номер строки, код CLDATA, место В, формат В, параметр 1, место 1, формат 1,параметр 2, место 2, формат 2,. . . , параметр n, место n, формат n)}

Где:

- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ*
- *Код инфор. слова - код информационного слова CLDATA*
- *Место А - местоположение информационного слова CLDATA*
- *Формат А - формат информационного слова CLDATA*
- *Номер строки - номер записи CLDATA*
- *Код CLDATA - Код командного слова ного слова CLDATA*
- *Место В - местоположение командного слова CLDATA*
- *Формат В - формат командного слова CLDATA*
- *Параметр n - параметр командного слова CLDATA*
- *Место n - местоположение параметра командного слова CLDATA*
- *Формат n - формат параметра командного слова CLDATA.*

Данный метод имеет существенный недостаток - дублирование записей в информационном массиве CLDATA, поэтому возникла необходимость изменения существующего информационного массива CLDATA на более совершенный массив. В итоге реконструированный информационный массив CLDATA содержит следующие поля:

Псевдо-код ={Главная часть}[Описание]{Параметры};

Где:

- *Псевдо-код - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ*
- *Главная часть - описывает код информационного*

слова CLDATA, местоположение информационного слова CLDATA, формат информационного слова CLDATA, код командного слова CLDATA, местоположение командного слова CLDATA, формат командного слова CLDATA;

- *Описание - анализирует вспомогательные слова главного командного слова CLDATA;*
- *Параметры - описывает параметры командного слова CLDATA, местоположение параметра командного слова CLDATA, формат параметра командного слова CLDATA, номер записи CLDATA;*

Следующим этапом разработки базы данных является разработка физической модели базы данных инвариантного постпроцессора

Физическая модель базы данных инвариантного постпроцессора включает следующие фаилы:

1. Фаил информационных слов АПТ разработанный в стандартной среде dBaseIV с расширением dbf. Структура данного фаила показана на следующем рисунке:

Где:

- *Field name - имя поля*
- *Type - тип поля*
- *Size - размер поля*
- *C - обозначает символьный тип;*

- *Word - информационное слово АПТ;*
- *Old_new - поле определяюшее принадлежность к*

немодифицируемым словамруемым словам;

- *MM_word - поле определяюшее принадлежность к информационным словам*
- *M_word - поле определяюшее принадлежность к главным словам*
- *Score - поле определяюшее псевдо-код*
- *Word_id - индексное поле;*

2. Стандартный текстовый фаил УЧПУ с расширением txt со следующими полями:

** Codd*
PC_codd Modal Format Fword

Где :

- ** - символ отделяющий различные группы командных слов*
- *Codd - определяет принадлежность последующих командных слов к определенной группе*
- *PC_codd - код назначенный постпроцессором для связывания кода CLDATA и определенного командного слова УЧПУ;*
- *Modal - определяет модальность данного информационного слова, напр. 0-не модальный1-модальный.*
- *Format - определяет существование параметров у данного командного слова*
- *Fword - определяет само командное слово и его формат если таковой существует;*

3. Стандартный текстовый фаил CLDATA с расширением txt со следующими полями;

Code={Main} [Ques] {Desc}

Где:

- *Code - Псевдо-код*
- *Main - описывает код информационного слова CLDATA, местоположение информационного слова CLDATA, формат информационного слова CLDATA, код командного слова CLDATA, местоположение командного слова*

CLDATA, формат командного слова CLDATA ;

- *Ques - анализирует вспомогательные слова главного командного слова CLDATA;*
- *Desc - описывает параметры командного слова CLDATA, местоположение параметра командного слова CLDATA, формат параметра командного слова CLDATA, номер записи CLDATA;*

В свою очередь поля Main, Qeus и Desc содержат собственные под-поля.
Main содержит:

- *P1=P1>T:Cn:F;*
- *P2=P2>T:Cn:F;*

Где:

P1 - основное слово CLDATA,
P2 - главное слово CLDATA,
T - тип слова,
Cn - местоположение слова
F - формат слова.

Qeus содержит следующую строку:

- *Cn=sbcode1>code, Cn= sbcode2>code+1;*

Где:

Cn - местоположение слова
sbcode1 - код вспомогательного слова
sbcode2 - код вспомогательного слова;
code - псевдо-код
code+1 - измененный псевдо-код;

Desc содержит следующую строку:

- *D1/D2/.../Dn;*

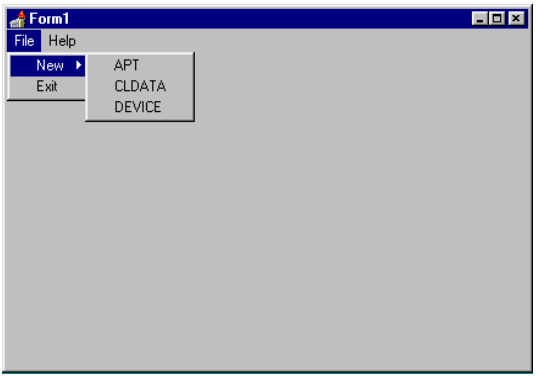
Где:
Dn в свою очередь содержит информацию о параметрах и номере строки

Dn=N(n)-{Pole};
N(n) - определяет номер строки;
Pole - содержит информацию о параметрах;
Pole= {a():Cn:F / x():Cn:F /y ():Cn:F / z():Cn:F /
ztool():Cn:F / I():Cn:F / J():Cn:F / K():Cn:F /

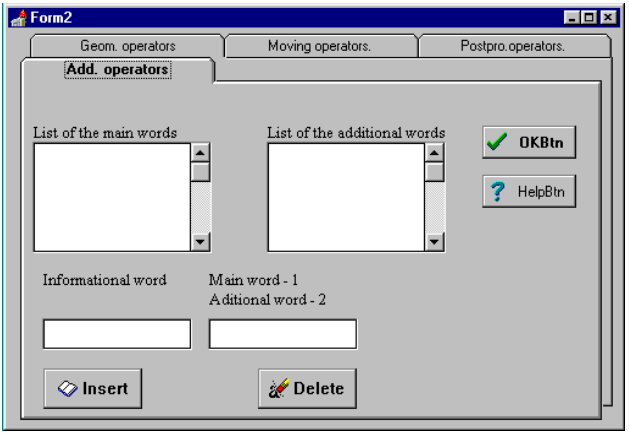
R():Cn:F / nst():Cn:F / nm():Cn:F}

- *a - любой параметр*
- *x,y,z - x,y,z координаты;*
- *I,J,K - I,J,K координаты;*
- *R - координаты;*
- *nn - номер строки CLDATA;;*
- *nst - количество слов в записи CLDATA;*

Для заполнения базы данных была создана интерактивная система в визуальной среде разработки прикладных програм Windows - Delphi.
Главное меню в интерактивном режиме имеет следующий вид:



Выбирая опцию "New" пользователь получает возможность редактировать базу данных слов АПТ, CLDATA и УЧПУ соответственно выбирая опции APT,CLDATA и DEVICE.
Если выбрана опция АПТ, то форма главного меню закрывается и появляется форма позволяющая модифицировать существующий набор слов АПТ, например добавлять, изменять и стирать слова. Данная форма показанна на следующем рисунке:



Ds,bhfz pfrkflre jghtltkztncz ghbyflkt;yjcnm ckjdf r rjyrhtnyjq uheggt jgthfnjhjd (yfgh= ghbyflkt;yjcnm r uheggt utjvtnhbxtcrbv jgthfnjhfv)=
List of the main words - выходит список (перечень) главных слов;
List of additional words - выходит список (перечень) вспомогательных слов
Informational word - ввод желаемого слова;
Main word - 1; additional word - 2 - ввод кода слова (если это главное слово - 1, если вспомогательнное - 2);
Insert - клавиша ввода в базу данных выбранного слова;
Delete - клавиша унистожения записи из базы данных;
Okbtn - клавиша возврата в главное меню;
Helpbtn - клавиша вызывающая информацию о данной форме;

Если в опции FILE главного меню выбирется опция CLDATA, то torga появляется форма позволяющая модифицировать фаил CLDATA. Данная форма показана на следующем рисунке: