Содержание

Введение

Техническое задание

1. Анализ мехатронной системы станка с ЧПУ

1.1 Общая структура и классификация систем ЧПУ

1.2 Обоснование применения программного управления оборудованием

1.3 Назначение постпроцессоров

2. Разработка системы подготовки обработки детали станка с ЧПУ

2.1 Функциональная модель системы подготовки обработки детали станка с ЧПУ

2.2 Алгоритм работы программного модуля

2.3 Выбор программных средств реализации системы

3. Технологическая реализация системы подготовки обработки детали станка с ЧПУ

3.1 Описание кодов программного модуля

3.2 Правила установки и настройки программного модуля

3.3 Тестирование и отладка программного модуля

3.4 Инструкция пользователя

4. Расчет экономической эффективности

4.1 Расчет затрат на разработку системы

4.2 Методология расчета общей стоимости владения программным продуктом

4.3 Экономический эффект

5. Обеспечение безопасности жизнедеятельности

5.1 Общие сведения о безопасности жизнедеятельности

5.2 Требования к рабочему месту инженера–программиста

5.3 Расчет естественного освещения рабочего места

5.4 Расчет искусственного освещения

Заключение

Список литературы

Приложение А Исходный текст программного модуля

Приложение Б Параметры проекта Delphi8

Приложение B Текст настроечного файла программного модуля4

Приложение Г Текст файлов связи с пакетом 3D моделирования0

Приложение Д Текст управляющей программы ЧПУ3

Введение

Велико разнообразие машин и устройств, которые создаются и используются человеком. Современные машины – это сложные технические системы, состоящие из большого числа технических аппаратов, приводов различного типа, приспособлений, измерительных и решающих устройств. Все эти машины и устройства представляют собой оборудование, являющееся основой функционирования самых различных систем: машиностроительных заводов, транспорта, электростанций и т.д.

Таким образом, под оборудованием можно понимать всю сумму технологий, на которую опирается человечество в своем развитии. Появление развитой вычислительной техники, персональных компьютеров, информационных сетей открывает реальные возможности для дальнейшей автоматизации оборудования – от автоматизации его проектирования до создания высокоадаптивных машин и систем различного назначения.

В машиностроительной и металлургической промышленности более половины всех станков работают в условиях массового, серийного и мелкосерийного производства, при этом удельный вес основного времени относительно не велик.

Применение систем числового программного управления в станках является наиболее эффективным средством повышения машинного времени и автоматизации мелкосерийного производства, что обеспечивает высокую технико-экономическую эффективность и позволяет организовать централизованную подготовку программ обработки даже вне предприятия, которые легко могут быть размножены.

При использовании станков с ЧПУ, наряду с повышением производительности, сроки подготовки производства сокращаются почти вдвое. Вместе с тем резко сокращаются слесарно-доводочные и другие работы, требующие больших трудозатрат и допускающие ошибки. Также можно получить значительную экономию средств на проектирование и изготовление технологической оснастки.

Появление промышленных роботов и других средств автоматизации производства значительно снижает объем ручного труда и выставляет высокие требования к уровню квалификации персонала. В значительной степени увеличивается доля инженерного труда, появляются потребности в новых знаниях и развитии новых технологий.

Автоматизация технологических процессов требует внесения серьезных изменений в технологию механической обработки, технологию создания технологических процессов, заставляет разрабатывать и вводить новые правила и стандарты для более эффективной работы всей системы.

Разработка программного обеспечения для автоматизации производства выходит на совершенно новый уровень с появлением современных средств проектирования, разработки и ведения документации. Разработка специального программного обеспечения особенно важна в условиях инструментального производства, где изготавливаются уникальные по сложности и точности изделия, большинство из которых просто невозможно изготовить универсальным путем.

Техническое задание

1. Основания для разработки

Систему подготовки обработки детали станка с числовым программным управлением (ЧПУ) Walter CIP6 разработать на основании заявки бюро разработки управляющих программ технологического отдела Инструментального производства ОАО "АВТОВАЗ".

Тема: Система подготовки обработки детали станка с ЧПУ

2. Назначение разработки

Система выполняется в виде программного модуля и создается для автоматического расчета всех необходимых параметров обработки по введенным данным и вывода текста программы, готового для ввода в память станка.

Система подготовки обработки детали станка с ЧПУ должна быть предназначена для:

* ввода исходных данных с чертежа и технологии;
* создания наладки режущего инструмента;
* расчета всех необходимых геометрических свойств;
* проверки по ограничениям станка;
* визуализации обработки;
* корректировки введенных данных;
* расчета траектории обработки;
* диагностики ошибок;
* формирования кода управляющей программы;
* выдачи в файл текста управляющей программы обработки.

Внедрение системы подготовки обработки детали шлифовального станка с ЧПУ Walter CIP6, позволит создавать программы для обработки сложного по конструкции осевого инструмента, проводить глубокий анализ конструкционных и режущих свойств реального изделия на этапе разработки управляющей программы, повысить качество выпускаемой продукции и избежать выпуск брака.

3. Требования к программному изделию

3.1 Требования к функциональным характеристикам

Программный модуль должен обеспечивать выполнение следующих функций:

* Ввод исходных данных с чертежа и технологии;
* Создание наладки режущего инструмента;
* Расчет всех необходимых геометрических свойств;
* Проверку по ограничениям станка;
* Визуализацию обработки;
* Корректировку введенных данных;
* Расчет траектории обработки;
* Назначение подачи с учётом ограничений, связанных с характером движения, допустимым диапазоном подач, особенностями реализации режимов разгона – торможения в ЧПУ;
* Формирование циклов обработки;
* Назначение припусков обработки;
* Расчет траектории обработки;
* Диагностика ошибок;
* Формирование кода управляющей программы;
* Выдачу в файл текста управляющей программы обработки.

Результатом работы будет файл управляющей программы, готовый к запуску на станке в ЧПУ Walter CIP6.

3.2 Требования к надежности

* ограничение ввода некорректной информации;
* контроль достоверности информации;
* обеспечение наглядности создаваемых процессов;
* защита от случайных нажатий клавиш;
* дружественность интерфейса пользователя.

3.3 Условия эксплуатации

* корректная работа пользователей;
* соблюдение инструкций работы с программным модулем;
* соответствие программного обеспечения, установленного на компьютере пользователя.

3.4 Требования к составу и параметрам технических средств

Система предназначена для работы на IBM – совместимых компьютерах с минимальной конфигурацией: процессор – 2200 Mh и выше, объем оперативной памяти 512 Mb и более, размер свободного места на жестком диски не менее 50Mb, монитор с разрешением 1024х768 и выше, профессиональная видеокарта, мышь с колесом прокрутки, клавиатура.

3.5 Требования к информационной и программной зависимости

Система должна работать под управлением операционной системы Windows 2000 или Windows XP, пакет твердотельного моделирования Designer Modeling 2005.

4. Требования к программной документации

Разрабатываемые программные модули должны быть документированы, т.е. тексты программ должны содержать все необходимые комментарии.

Информационная система должна включать справочную систему о работе и подсказки пользователю.

В состав сопровождающей документации должны входить:

* описание программных модулей;
* описание функций программных модулей.

Инструкция пользователю по работе с программными модулями.

5. Технико-экономические показатели

Разработать систему подготовки обработки детали станка с ЧПУ предназначенную для создания управляющей программы операций шлифования осевого инструмента, автоматизации расчетов технологических переходов, а также визуализации обработки при помощи пакета твердотельного моделирования Designer Modeling 2005.

Использование данного программного модуля должно обеспечить корректное преобразование траектории движения инструмента и технологические команды в коды управления соответствующей комбинацией "станок - система ЧПУ", без использования другого специализированного программного обеспечения. А также позволит создавать программы для обработки сложного по конструкции осевого инструмента, проводить глубокий анализ конструкционных и режущих свойств реального изделия на этапе разработки управляющей программы, повысить качество выпускаемой продукции.

Разработать систему подготовки обработки детали станка с ЧПУ, которая будет иметь простой интерфейс, понятный конечному пользователю, высокую скорость работы, возможность анализа ряда технологических и конструкционных параметров.

6. Стадии и этапы разработки

Разработка основного алгоритма работы модуля дней

Отладка и тестирование на реальных изделиях дней

Разработка пользовательского интерфейса дней

Тестирование специалистами бюро разработки управляющих программ для анализа работы и выдачи замечаний дней

Доработка замечаний дней

Окончательное тестирование работы программного модуля

постпроцессора дней

7. Запуск и отладка программного модуля

Запуск и отладка программного модуля включает следующие этапы:

* Разработка, согласование и утверждение программы и методики испытаний;
* Проведение предварительных государственных, межведомственных, приёмо-сдаточных и других видов испытаний;
* Корректировка программы и программной документации по результатам испытаний;
* Подготовка и передача программы и программной документации для сопровождения и изготовления;
* Оформление и утверждение акта о передаче программы на сопровождение и изготовление.

8. Порядок контроля и приемки программного модуля

Первоначальные испытания работоспособности разработанного программного модуля мехатронной системы станка с ЧПУ Walter CIP6 проводит разработчик на стадии его разработки. Контроль работоспособности готового программного модуля постпроцессора проводит заказчик. Для этого он может привлекать экспертов своей организации или экспертов со стороны.

# 1. Анализ мехатронной системы станка с ЧПУ

## 1.1 Общая структура и классификация систем ЧПУ

На основе достижений кибернетики, электроники, вычислительной техники и приборостроения были разработаны принципиально новые системы программного управления – системы ЧПУ, широко используемые в промышленности. Эти системы называют числовыми потому, что величина каждого хода исполнительного органа станка задается с помощью числа. Каждой единице информации соответствует дискретное перемещение исполнительного органа на определенную величину, называемой разрешающей способностью системы ЧПУ или ценой импульса.

В определенных пределах исполнительный орган можно переместить на любую величину, кратную разрешающей способности. Число импульсов, которое можно подать на вход привода, чтобы осуществить требуемое перемещение, определяется отношением длины перемещения к цене импульса системы ЧПУ. Число импульсов, записанное в определенной системе кодирования на носителе информации (перфоленте, магнитной ленте и других), является программой, определяющей величину размерной информации.

Станки с программным управлением по виду управления подразделяют на станки и системами циклового программного управления и станки с системами числового программного управления. Системы циклового программного управления более просты, так как в них программируется только цикл работы станка, а величины рабочих перемещений, т.е. геометрическая информация, задаются упрощенно, например, с помощью упоров. В станках с ЧПУ управление осуществляется от программы, в которой в числовом виде занесена и геометрическая, и технологическая информация.

Система ЧПУ – это совокупность специализированных устройств, методов и средств, необходимых для реализации ЧПУ станком, предназначенная для выдачи управляющих воздействий исполнительным органам станка в соответствии с управляющей программой [1].

Устройство программного управления станками – это часть системы ЧПУ, выполненная как единое целое с ней и осуществляющая выдачу управляющих воздействий по заданной программе.

Числовое программное управление – это управление, при котором программу задают в виде записанного на каком-либо носителе массива информации. Управляющая информация для систем ЧПУ является дискретной и ее обработка в процессе управления осуществляется цифровыми методами. Управление технологическими циклами практически повсеместно осуществляется с помощью программируемых логических контроллеров, реализуемых на основе принципов цифровых электронных вычислительных устройств.

Структурная схема системы ЧПУ представлена на рисунке 1, а. Чертеж детали (ЧД), подлежащий обработке на станке с ЧПУ, одновременно поступает в систему подготовки программы (СПП) и систему технологической подготовки (СТП). Последняя обеспечивает систему подготовки программы данными о разрабатываемом технологическом процессе, режимах резания и так далее. На основании этих данных разрабатывается управляющая программа (УП). Наладчики устанавливают на станок приспособления, режущие инструменты согласно документации, разработанной в системе технологической подготовки. Установку заготовки и снятие готовой детали осуществляет оператор или автоматический загрузчик. Считывающее устройство (СУ) считывает информацию с носителя программы. Информация поступает в устройство ЧПУ, которое выдает управляющие команды на целевые механизмы (ЦМ) станка, осуществляющие основные и вспомогательные движения цикла обработки. Операционная система на основе информации (фактическое положение, скорость перемещения исполнительных узлов, фактический размер обрабатываемой поверхности, тепловые и силовые параметры технологической системы и др.) контролируют величину перемещения целевого механизма. Станок содержит несколько целевых механизмов, каждый из которых включает в себя (рисунок 1, б): двигатель (ДВ), являющийся источником энергии; передачу (П), служащую для преобразования энергии и ее передачи от двигателя к исполнительному органу (ИО); собственно исполнительный орган (стол, салазки, суппорт, шпиндель и т.д.), выполняющие координатные перемещения цикла.

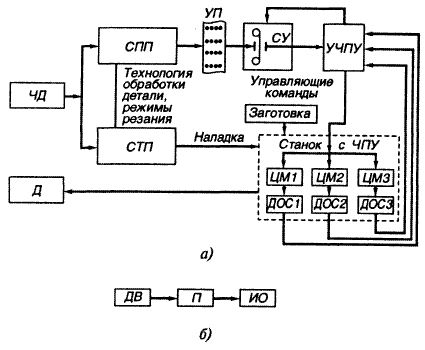


Рисунок 1 - Структурная схема системы ЧПУ и целевого механизма

Система ЧПУ может видоизменяться в зависимости от вида программоносителя, способа кодирования информации в управляющей программе и метода ее передачи в систему ЧПУ. Устройство ЧПУ размещают рядом со станком (в одном или двух шкафах) или непосредственно на станке (в подвесных или стационарных пультах управления). Двигатели приводов подач станков с ЧПУ, имеющие специальную конструкцию и работающие с конкретным устройством ЧПУ, являются составной частью системы ЧПУ.

Все данные, необходимые для обработки заготовки на станке с ЧПУ, получает от управляющей программы, которая содержит два вида информации: геометрическую и технологическую. Геометрическая информация – координаты опорных точек траектории движения инструмента, а технологическая – данные о скорости, подаче, номере инструмента и т. д. Управляющую программу записывают на программоносителе. В оперативных системах ЧПУ программа может вводиться (с помощью клавиш) непосредственно на станке.

Важнейшей технической характеристикой систем ЧПУ является ее разрешающая способность или дискретность, т. е. минимально возможная величина линейного и углового хода исполнительного органа станка, соответствующая одному управляющему импульсу. Большинство современных систем ЧПУ имеют дискретность 0,001 мм/импульс, реже 0,0001 мм/импульс.

Системы ЧПУ классифицируют по следующим признакам:

* по уровню технических возможностей;
* по технологическому назначению;
* по числу потоков информации (незамкнутые, замкнутые, самоприспосабливающиеся или адаптивные);
* по принципу задания программы (в декорированном виде, т.е. в абсолютных координатах или в приращениях от ЭВМ);
* по принципу привода (ступенчатый, регулируемый, следящий, шаговый);
* по числу одновременно управляемых координат;
* по способу подготовки и ввода управляющей программы.

По уровню технологических возможностей международной классификации системы ЧПУ делятся на следующие классы:

* NC – системы с покадровым чтением перфоленты на протяжении цикла обработки каждой заготовки;
* SNC – системы с однократным чтением всей перфоленты перед обработкой партии одинаковых заготовок;
* CNC – системы со встроенной малой ЭВМ (компьютером, микрокомпьютером);
* DNC – системы прямого числового управления группами станков от одной ЭВМ;
* HNC – оперативные системы с ручным набором программ на пульте управления.

По технологическому назначению системы ЧПУ подразделяются на четыре вида: позиционные; обеспечивающие прямоугольное формообразование; обеспечивающие прямолинейное формообразование; обеспечивающие криволинейное формообразование [22].

Позиционные системы ЧПУ обеспечивают высокоточное перемещение (координатную установку) исполнительного органа станка в заданную программой позицию за минимальное время. По каждой координатной оси программируется только величина перемещения, а траектория перемещения может быть произвольной. Перемещение исполнительного органа из позиции в позицию осуществляется с максимальной скоростью, а переход к заданной позиции – минимальной скоростью. Точность позиционирования повышается в результате подхода исполнительного органа к заданной позиции всегда с одной стороны (например, слева направо). Позиционными системами ЧПУ оснащают сверлильные и координатно-расточные станки.

Системы ЧПУ, обеспечивающие прямоугольное формообразование, в отличие от позиционных систем позволяют управлять перемещением исполнительного органа станка в процессе обработки. В процессе формообразования исполнительный орган станка перемещается по координатным осям поочередно, поэтому траектория инструмента имеет ступенчатый вид, а каждый элемент этой траектории параллелен координатным осям. Чтобы сократить время перемещения исполнительного органа из одной позиции в другую, в ряде случаев используют одновременное движение по двум координатам. При грубом позиционировании подход исполнительного органа к заданной позиции осуществляется с разных сторон, а при точном позиционировании – всегда с одной стороны. Число управляемых координат в таких системах достигает 5, а число одновременно управляемых координат – 4. Указанными системами оснащают токарные, фрезерные, расточные станки.

Системы ЧПУ, обеспечивающие прямолинейное (под любым углом к координатным осям станка) формообразование и позиционирование, управляют движением инструмента при резании одновременно по двум координатным осям (X и Y). В данных системах используют двухкоординатный интерполятор, выдающий управляющие импульсы сразу на два привода подач. Общее число управляемых координат в таких системах 2 – 5. Указанные системы обладают большими технологическими возможностями (по сравнению с прямоугольными) и применяются для оснащения токарных, фрезерных, расточных и других видов станков.

Системы ЧПУ, обеспечивающие криволинейное формообразование, позволяют управлять обработкой плоских и объемных деталей, содержащих участки со сложными криволинейными контурами.

Системы ЧПУ, обеспечивающие прямоугольное и криволинейное формообразование, относятся к контурным (непрерывным системам), так как они позволяют обрабатывать заготовку по контуру. Контурные системы ЧПУ имеют, как правило, шаговый двигатель.

Многоцелевые (сверлильно-фрезерно-расточные) станки для расширения их технологических возможностей оснащают контурно-позиционными системами ЧПУ.

По числу потоков информации системы ЧПУ делятся на замкнутые, разомкнутые и адаптивные.

Разомкнутые системы ЧПУ характеризуются наличием одного потока информации, поступающего со считывающего устройства к исполнительному органу станка. В механизмах подач таких систем используют шаговые двигатели. Крутящий момент, развиваемый шаговым двигателем, недостаточен для привода механизма подачи. Поэтому указанный двигатель применяют в качестве задающего устройства, сигналы которого усиливаются различными способами, например, с помощью гидроусилителя моментов (аксиально-поршневого гидродвигателя), вал которого связан с ходовым винтом привода подач. В разомкнутой системе нет датчика обратной связи, и поэтому отсутствует информация о действительном положении исполнительных органов станка.

Замкнутые системы ЧПУ характеризуются двумя потоками информации: от считывающего устройства и от датчика обратной связи. В этих системах рассогласование между заданными и действительными величинами перемещения исполнительных органов устраняется благодаря наличию обратной связи.

Адаптивные системы ЧПУ характеризуются тремя потоками информации:

* от считывающего устройства;
* от датчика обратной связи по пути;
* от датчиков, установленных на станке и контролирующих процесс обработки по таким параметрам, как износ режущего инструмента, изменение сил резания и трения, колебание припуска и твердости материала обрабатываемой заготовки и т.д. Такие системы позволяют корректировать программу обработки с учетом реальных условий резания.

По способу подготовки и ввода управляющей программы различают так называемые оперативные системы ЧПУ (в этом случае управляющую программу готовят и редактируют непосредственно на станке, в процессе обработки первой детали из партии или имитации ее обработки) и системы, для которых управляющая программа готовится независимо от места обработки детали. Причем независимая подготовка управляющей программы может выполняться либо с помощью средств вычислительной техники, входящих в состав систем ЧПУ данного станка, либо вне ее (вручную или с помощью системы автоматизации программирования).

## 1.2 Обоснование применения программного управления оборудованием

По мере развития общества потребности человека неуклонно растут. Добиваясь определенного уровня благосостояния, человек стремится к новым вершинам – такова его природа. В то же время, находя пути удовлетворения своих потребностей, человек вынужден становиться производителем материальных благ. А став производителем, вынужден постоянно работать над совершенствованием технологии производства, повышая его эффективность и улучшая качество продукции [2]. На сегодняшний день никто из нас не мыслит себе жизнь без домашней и офисной техники, посуды, упаковки и т.д., имеющих современный дизайн. А это – пластмассовые и металлические корпуса, состоящие из массы сложных поверхностей, которые подчас невозможно даже отобразить на чертеже. Уровень современной технологии позволяет создавать в памяти компьютера трехмерные модели деталей сложной формообразующей оснастки штампов и пресс-форм и изготавливать их на станках с числовым программным управлением в условиях единичного производства. При этом процесс подготовки управляющих программ для станка с числовым программным управлением выполняется автоматически на основе трехмерной модели изделия. Сложная деталь изготавливается "с листа", без создания опытных образцов. Естественно, что при наличии на предприятии потока индивидуальных заказов остро встает проблема организации и планирования производства, решаемая также при помощи современных автоматизированных систем. Идея гибкого автоматизированного производства не является новой, но при современном уровне развития компьютерных технологий и сетевых решений приобретает особое значение. В то же время задача создания такого производства достаточно сложна. Она выдвигает повышенные требования, как к программному обеспечению инженерного корпуса, так и к системам управления оборудованием.

Наш взгляд на систему числового программного управления металлообрабатывающего оборудования за последние 10 - 15 лет претерпел кардинальные изменения. Ранее единственной тенденцией развития станков с числовым программным управлением была тенденция превращения их в обрабатывающие центры. При этом станки оснащались магазинами, рассчитанными на большое количество инструментов. Для встраивания станка в автоматическую линию он комплектовался быстросменными столами-спутниками и их накопителями. Программирование осуществлялось вручную. С появлением мощных (и особенно персональных) компьютеров стало возможным создавать управляющие программы для станков с числовым программным управлением автоматически, сократив до минимума количество ручного труда. Это позволило разрабатывать управляющие программы для обработки сложных трехмерных поверхностей, используемых чаще всего в формообразующих деталях оснастки штампов и пресс-форм. При этом требования, предъявляемые к станкам с числовым программным управлением, изменились. Произошло это в основном благодаря увеличению размера и структурным изменениям управляющей программы, а также существенному увеличению доли основного времени обработки при соответствующем уменьшении доли вспомогательного времени. Управляющая программа (УП) обработки одной поверхности теперь достигает нескольких мегабайтов (а иногда и десятков мегабайтов) и состоит из множества "коротких кадров" – программных блоков, описывающих короткие перемещения инструмента (чем выше точность обработки, тем короче эти перемещения, так как их длина определяет точность аппроксимации). Обработка при этом ведётся преимущественно одним инструментом много часов подряд, а станок теряет уже ставшее привычным оснащение: магазины, столы-спутники и т.д. Разумеется, речь здесь идет о станке, предназначенном для изготовления сложной формообразующей оснастки. Количество управляемых координат при этом возросло до пяти. Но вместе с этим значительно выросли требования к системе ЧПУ.

Учитывая имеющийся на данный момент практический опыт, можно сказать, что современная система ЧПУ, предназначенная для сложной трехмерной обработки, должна обеспечивать следующее:

1. Возможность ввода (приема с внешнего носителя или через сеть) управляющих программ неограниченного размера, их редактирования и исполнения как единого целого.
2. С целью уменьшения основного времени обработки – опережающая (по отношению к исполнению) расшифровка кадров управляющей программы:

* возможность отработки движения без снижения до нуля скорости в конце перемещения, описанного в каждом отдельном кадре (при соблюдении условия отсутствия превышения максимальных ускорений по осям);
* это позволяет отрабатывать сложную траекторию, описываемую в управляющей программе и состоящую из множества "коротких кадров", на скорости, близкой к заданной скорости подачи;
* определение предельных ускорений по управляемым осям с учетом динамических характеристик станка;
* возможность перехода с одной траектории обработки на другую на рабочей подаче без торможений и разгонов по трехмерной петле, рассчитанной системой CAM (computer-aided manufacturing);
* для этого требуется интеграция с CAM-системой или с CAM-сервером через сеть;
* возможность работы на повышенных скоростях рабочих подач (до 60 м/мин);
* с этой целью помимо достаточной для перекрытия требуемого диапазона регулирования разрядности цифро-аналогового преобразователя необходимо, чтобы время гарантированной реакции системы управления движением было относительно малым (около 200 мкс).

1. С целью снижения времени переналадки – доступ к файлам и ресурсам конструкторского и технологического бюро через стандартную сеть, включая поддержку стандартных (в том числе распределенных) баз данных:

* встроенная функция трехмерной коррекции траектории движения инструмента на величину его радиуса;
* возможность интеграции с CAM-сервером для выполнения полноценной коррекции УП по результатам предыдущих операций (в том числе трёхмерной коррекции траектории движения инструмента на величину его радиуса);
* привязка набора управляющих программ, подпрограмм, корректоров, параметров системы и служебной информации к конкретному изделию (проекту);
* возможность параллельно с процессом обработки выполнять редактирование или эмуляцию работы другой управляющей программы, ввод управляющей программы с дискеты (в том числе с использованием многотомных архивов), доступ к сети, включая обращение к CAM-серверу;
* возможность автоматизированного измерения (или поиска) баз заготовки (детали), контроля размеров детали и инструмента.

1. С целью увеличения коэффициента загрузки оборудования в условиях единичного и мелкосерийного производства – возможность работы совместно с системой управления верхнего уровня на основе стандартных сетевых технологий, возможность информационной поддержки систем планирования и диспетчеризации на уровне цеха или участка.
2. С целью повышения надежности системы – повышение ресурса узлов ЧПУ за счет применения отлаженных серийных модулей (плат), применение вместо традиционных реле высоконадежных твердотельных силовых модулей (с гальванической изоляцией прочностью не менее 1500 В) и оптоэлектронных датчиков положения.
3. С целью увеличения ремонтопригодности и уменьшения времени поиска неисправностей – наглядное представление сигналов электроавтоматики в соответствии с электрической схемой станка и приведенными в техническом описании алгоритмами работы:

* наличие подсистемы диагностики и выдачи сообщений оператору;
* возможность непосредственного управления исполнительными устройствами;
* конфигурирование ЧПУ из стандартных узлов, из которых также могут быть построены системы управления другим оборудованием завода, и с использованием по возможности стандартного базового программного обеспечения.

1. С целью обеспечения гибкости системы – возможность постоянной доработки системы ЧПУ в соответствии с непрерывно растущими требованиями современного производства:

* возможность быстрой адаптации к любому технологическому оборудованию (в том числе не металлорежущему);
* возможность интеграции со сложными автономными системами (например, с системами технического зрения);
* возможность выполнения необходимых измерений и обмера детали-прототипа с целью создания трехмерной математической модели или построения управляющей программы для копирования;
* наличие гибкой архитектуры системы, реализуемой в зависимости от поставленной задачи;
* надежная поддержка фирмой-разработчиком и обновление версий базового программного обеспечения.

Технология работы на оборудовании с четырьмя и более управляемыми осями на данный момент не достаточно хорошо отработана. Это связано с особенностями систем ЧПУ у каждого производителя, собственным набором и направлением дополнительных осей координат, типом станка и его назначением.

Системы ЧПУ у такого оборудования чаще всего обладают развитой системой программирования, которая может быть реализована на базе языка программирования высокого уровня. Примером может служить система ЧПУ фирмы Siemens со встроенным языком программирования Sinumerik.

Применение станков с ЧПУ в сравнении с обычным оборудованием создает ряд технико-экономических преимуществ.

Производительность этих станков выше производительности станков того же типа, но без программного управления, в три раза, потребность же в производственных площадях в три раза меньше. Значительно вырастает производительность труда у рабочих.

Большой эффект дают станки с ЧПУ при выполнении особо сложных операций, поэтому с их использованием высвобождаются высококвалифицированные рабочие, а также резко сокращаются затраты на технологическую подготовку производства, эксплуатацию инструмента, содержание контролеров отдела технического контроля.

Главный эффект программного оборудования заключается в увеличении до 80-90% работы оборудования (15-20% у обычных станков). Обусловлено это тем, что резко сокращается вспомогательное время, время на смену инструмента и переналадку оборудования.

Переналадка станков в этом случае заключается в замене программы, записанной на магнитной ленте или другом программном носителе, а в ряде случаев в замене инструментов. Широкий диапазон работ, выполняемых станками с ЧПУ, делает их особенно ценными в единичном и мелкосерийном производстве, а также на предприятиях, выпускающих сложную продукцию. Имеется опыт включения станков с программным управлением в поточные линии на предприятиях серийного и массового производства.

В современных условиях широко распространяется такой вид программного оборудования, как обрабатывающие центры. Они представляют собой многооперационные станки с автоматической сменой инструмента. По мнению специалистов, обрабатывающие центры по своей производительности эквивалентны 3-4 станкам с ЧПУ и 8-12 обычным станкам. При условии правильного выбора и рациональной эксплуатации затраты на приобретение обрабатывающих центров окупаются за 3-4 года.

Для эффективного использования станков с ЧПУ необходимо создать систему организованного обеспечения. Она должна представлять собой комплекс взаимодействующих мероприятий, подчиненных основной задаче - изготовлению деталей высокого качества в намеченные сроки при минимальных затратах труда и себестоимости. Система организации работ должна включать технико-экономическое обоснование применения станков с ЧПУ, номенклатуру деталей для обработки на станках, специальную структуру системы, надлежащее обслуживание станков, автоматизированную разработку управляющих программ. Одним из условий достижения высоких экономических показателей эксплуатации станков с ЧПУ является формирование целесообразной номенклатуры обрабатываемых деталей. Практика эксплуатации станков с ЧПУ показала, что эффективная их работа возможна при построении специальной организационной структуры, сориентированной на изменения, которые вносит появление в парке оборудования станков с ЧПУ. Такая структура должна включать производственные цехи и участки, подразделения экономического обслуживания, специализированную технологическую службу. Опыт отечественных и зарубежных предприятий свидетельствует о целесообразности установки станков с ЧПУ в одном производственном помещении, создание специализированных цехов и участков. Расположение станков в одном помещении создает условия для более качественного их обслуживания, многостаночного обслуживания, улучшения планирования и контроля за работой оборудования и т. п.

1.3 Назначение постпроцессоров

Постпроцессор – это модуль, преобразующий файл траектории движения инструмента и технологических команд, рассчитанный процессором CAM или CAD/CAM-системы, в файл управляющей программы в строгом соответствии с требованиями методики ручного программирования конкретного комплекса "станок – система с ЧПУ". Постпроцессор выполняет немалое количество функций, например:

* кодирует линейные перемещения сообразно цене импульса;
* выполняет линейную или круговую интерполяцию перемещений по дуге окружности, а также кодирует их в импульсах;
* рассчитывает динамику перемещений, отслеживая и, если нужно, уменьшая слишком большую подачу на малом перемещении (станок не успеет разогнаться);
* автоматически выдает в кадр вектора или функции коррекции на радиус инструмента;
* строит текущий кадр по шаблону, автоматически нумеруя кадры под адресом "N";
* превращает подачи, назначенные технологом, в конкретный набор символов с адресом "F" и выдает в нужное место кадра;
* оформляет как начало, так и конец управляющей программы, а также структуру кадра.

На самом деле число функций, выполняемых среднестатистическим постпроцессором значительно больше. Например постпроцессор должен выдавать в кадр перемещения только по тем координатам, движения по которым имело место, а также правильно определять выпуклость или вогнутость контура детали для правильного расчета вектора коррекции и многое другое [3].

Разработка полноценного постпроцессора с нуля (т.е. без средств автоматизации постпроцессирования) может занять у программиста средней квалификации более 6 месяцев.

Первые постпроцессоры специально обученный программист разрабатывал для каждого комплекса "станок – система с ЧПУ" индивидуально. Далее происходил длительный процесс доводки постпроцессора, путем активных консультаций с технологом-расчетчиком управляющих программ, а также опытными прогонами управляющих программ (рассчитанных при помощи постпроцессора) на станке с ЧПУ. Постпроцессор сдавался в опытную эксплуатацию заказчику только после успешного прохождения испытаний, затем наступал процесс исправления ошибок и неучтенных при разработке особенностей программирования стойки и даже технологии изготовления деталей, принятых на данном предприятии. Такой постпроцессор был индивидуальный для данного станка, стойки ЧПУ и, нередко, технологии обработки. Стоимость разработки индивидуального постпроцессора была очень высока, и могла достигать нескольких тысяч долларов. Исправить ошибки и сделать нововведения в постпроцессоре мог только программист, разработавший данный постпроцессор.

Созданием индивидуальных постпроцессоров занимались все фирмы, как отечественные, так и зарубежные, примерно в 1960-1970 годах прошлого столетия.

Постпроцессирование прошло несколько ступеней развития. С одной стороны в 80-х годах прошлого века наблюдался всемирный бум автоматизации машиностроения, с другой – очень быстро увеличивалось количество новых станков с непременно новой системой ЧПУ, с третьей стороны возник небывалый спрос на средства автоматизированного проектирования для таких станков со стороны заводов и компаний. В этих условиях метод индивидуального постпроцессирования оказался не эффективным, в виду очень больших временных затрат на создание и отладку каждого постпроцессора. Эти объективные причины подтолкнули разработчиков постпроцессоров к идее автоматизации собственного труда – то есть средств автоматизации разработки постпроцессоров.

Одним из первых методов автоматизации разработки постпроцессоров стало обобщение информации об использовании одной и той же системы с ЧПУ вместе со станками различных производителей, по принципу обработки (например, токарная, фрезерная и т.д.). Это оказалось возможным, в связи с тем, что управляющие программы для таких станков, "вооруженных" однотипной системой с ЧПУ, различались в лучшем случае незначительными вариациями в оформлении структуры кадра, значностью перемещений, оформлением начала и конца программы. Поэтому вскоре начали обобщать алгоритмы разработки постпроцессоров на однотипное оборудование разных фирм, но имеющее одну и ту же систему с ЧПУ.

Идея использования обобщенных постпроцессоров дала существенные преимущества, ведь разработка постпроцессора для новой модификации станка с системой ЧПУ, для которой уже имелся обобщенный постпроцессор, требовала от программиста всего лишь небольшой модификации узкого набора программ для учета особенностей нового оборудования.

В результате, в разы сократились сроки, стоимость и трудоемкость разработки нового постпроцессора, существенно снизились издержки фирм-разработчиков постпроцессоров и их заказчиков. Этот метод оказал сильное воздействие на конкурентную борьбу между производителями CAM-систем в мире. Кроме того, некоторые фирмы продавали именно "обобщенные постпроцессоры" на 5-10 станков с одной системой с ЧПУ по цене одного индивидуального, что было выгодно их клиентам и чрезвычайно невыгодно фирмам-конкурентам, еще не освоившим эту технологию.

Небольшое число современных CAM-систем до сих пор используют в своем составе обобщенные постпроцессоры.

Следующим шагом развития систем разработки постпроцессоров стало применение автоматических корректоров кадров управляющих программ. Суть этого нововведения – дать возможность разработчику или пользователю описать на специальном макроязыке изменения, которые затем автоматически и последовательно выполняются постпроцессором над каждым кадром во время формирования управляющей программы. Этот метод был впервые примененный в системе PEPS. Макроязык коррекции кадров управляющей программы применяется теперь весьма широко и не только в обобщенных постпроцессорах.

Почти одновременно с появлением методологии обобщенного постпроцессирования, начал разрабатываться метод создания универсальных постпроцессоров. Такие постпроцессоры последовательно читают записи из файла траектории движения инструмента и техкоманд (CLDATA-файл) и выполняют преобразование этих записей в один или несколько кадров управляющей программы по некоторым правилам, отличным для разных станков и систем ЧПУ.

Постпроцессор соотносит каждой записи CLDATA-файла алгоритм ее превращения в кадр управляющей программы, и сохранить эти правила отдельно для каждого станка-системы ЧПУ в виде файла. Именно это дало возможность создать один универсальный постпроцессор как машину, транслирующую каждую запись CLDATA-файла в кадры управляющей программы по правилам, которые можно подгружать из внешних файлов.

Такой метод получил название "универсальный постпроцессор". Программист описывал алгоритмы обработки каждой записи CLDATA-файла применительно к методике ручного программирования конкретного комплекса "станок – система с ЧПУ" и сохранял эти правила (алгоритмы) в виде текстовых файлов-постпроцессоров. Технолог, в свою очередь, лишь выбирал – при помощи какого файла-описателя алгоритмов, преобразовать свой CLDATA-файл в файл управляющей программы.

Эта идея, заимствованная из методов построения трансляторов с настраиваемой лексикой и семантикой, получила шилокое развитие на рубеже 90-х годов прошлого века. Подавляющее большинство CAD/CAM-систем используют сегодня именно такой метод для решения проблем постпроцессирования.

Особняком от всех этих способов обработки стоит разработка программного обеспечения для оборудования, с узкой сферой применения. К таким относятся всевозможные шлифовальные станки для изготовления осевого режущего инструмента. Основной проблемой разработки подобного программного обеспечения является практически полное отсутствие универсальных средств их разработки и привязанность к технологии изготовления. Также большую проблему создает существенная зависимость полученной геометрии на изделии от геометрических параметром инструмента.

Также, подобное оборудование накладывает огромное количество ограничений на обработку, которые необходимо постоянно учитывать.

В связи с этим, наиболее целесообразно создавать специальные программные модули, которые могут взять на себя всю математическую нагрузку по расчету траекторий движений, учета ограничений и создания управляющей программы.

Современные постпроцессоры для подобного оборудования требуют только введения данных из чертежа или иной документации на изделие, а затем путем внутренних расчетов выдают готовую программу обработки, с учетом всех технологических особенностей. Подобные программные модули являются узкоспециализированными и разрабатываются по специальному заказу предприятия. Последней тенденцией стало встраивание таких модулей в систему ЧПУ, что позволило квалифицированному рабочему самостоятельно программировать сложные виды обработки.

# 2. Разработка системы подготовки обработки детали станка с ЧПУ

## 2.1 Функциональная модель системы подготовки обработки детали станка с ЧПУ

Для создания функциональной модели удобно пользоваться CALS системами. CALS – класс информационных технологий, направлен на обеспечение безбумажной поддержки жизненного цикла продукта. Предметом CALS является безбумажная технология взаимодействия между организациями заказывающими, производящими и эксплуатирующими тот или иной продукт, а также формат представления соответствующих данных. Доказав свою эффективность, CALS технологии начали активно применяться в промышленности, строительстве, транспорте и других отраслях экономики, расширяясь и охватывая все этапы жизненного цикла продукта – от маркетинга до утилизации. В данном случае необходима только та часть функций системы BpWin, работающей по CALS технологии, которая отвечает за построение функциональной модели [23].

Функциональная модель представляет собой структурированное изображение функций производственной системы или среды, информации и объектов, связывающих эти функции. Модель строится методом декомпозиции: от крупных составных структур к более мелким и простым.

На нулевом уровне (рисунок 2) вся система представляется в виде черного ящика, целью которого является создание управляющей программы для системы ЧПУ. Для этого на вход системы должен поступить ряд данных. Данные из чертежа изделия включают весь комплекс информации о геометрических свойствах изделия, его размерах, наборе поверхностей, подлежащих обработке, их точности и шероховатости.

Технологический процесс содержит рекомендации по режимам резания, технологической оснастке, применяемой при данной обработке. Также в технологическом процессе могут содержаться технологические нюансы обработки, которые обязательно надо учесть при создании управляющей программы системы ЧПУ.

Также на вход должна поступить информация об инструменте и заготовке, которые будут использоваться в процессе обработки данного изделия.

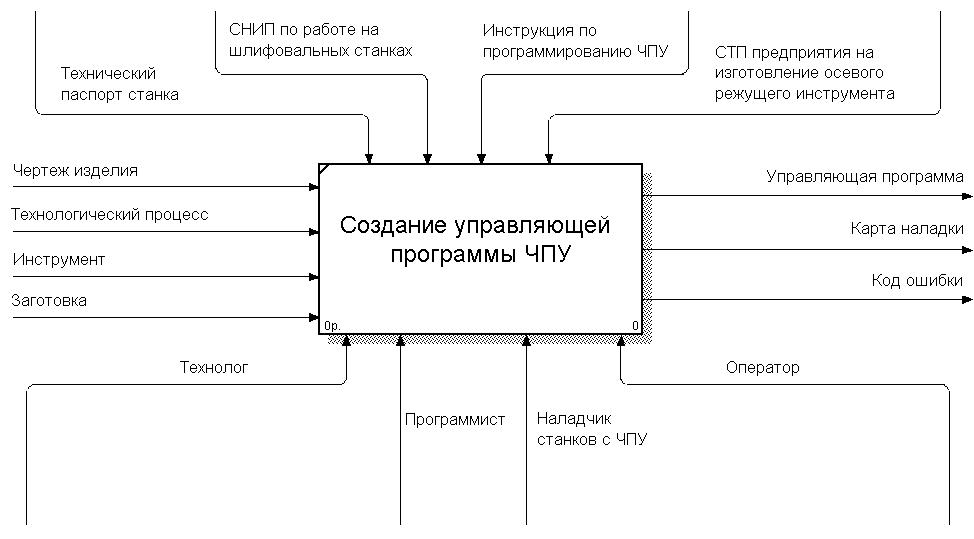


Рисунок 2 - Нулевой уровень функциональной модели

Работать с данной системой будут технологи, отвечающие за технологический процесс, программисты, отвечающие за корректную работу всех программных модулей системы и процесс постпроцессирования в код управляющей программы системы ЧПУ. Совместно с наладчиком станков с ЧПУ будет осуществляться ввод данных об инструменте и заготовке, в частности об их взаимном расположении друг относительно друга. Оператор будет вводить данные коррекции после правки кругов. Иногда функции оператора и наладчика станков с ЧПУ могут объединяться в одном специалисте.

Работа системы должна осуществляться на основании следующих документов:

* технического паспорта станка, в котором содержится информация об особенностях его эксплуатации, ограничениях перемещений по осям, режимах работы и габаритных ограничениях заготовки и инструмента;
* инструкции по программированию ЧПУ, в которой содержится справочная информация о языке программирования, вспомогательных и служебных функциях, ограничениях системы ЧПУ;
* Санитарные нормы и правила (СНИП) по работе на шлифовальных станках содержит методологию по способам безопасного ведения обработки;
* Стандарт предприятия (СТП) предприятия на изготовление осевого режущего инструмента содержит всю технологическую базу по обработке данного инструмента, накопленную за все время его работы, а также возможные варианты решения спорных вопросов по конструкции изделия и технологии его обработки.

Результатом работы будет выдача управляющей программы для системы ЧПУ, карты наладки для данного оборудования или кода ошибки, если введенных данных не достаточно, либо они ошибочны и выполнить расчеты и преобразования по ним не возможно.

На первом уровне система разбивается два модуля (рисунок 3).

Первый модуль занимается расчетом перемещений инструмента, учитывая технологические и технические ограничения. Все входные данные поступают именно на этот модуль и соответственно все их преобразования также происходят в этом модуле. В связи с этим, этот модуль должен работать со всеми выше перечисленными документами. Работать с этим модулем должны также все выше перечисленные люди.

Модуль будет выдавать два вида файлов. Одна группа файлов будет являться управляющей программой системы ЧПУ, две другие группы файлов системные, обеспечивающие связь первого модуля со вторым. Первая группа системных файлов должна передавать исходные данные, содержащие информацию об оснащении станка (тип оснастки, инструмент, заготовка, их взаимное расположение). Вторая группа файлов должна содержать рассчитанные данные (траектории всех перемещений инструмента).

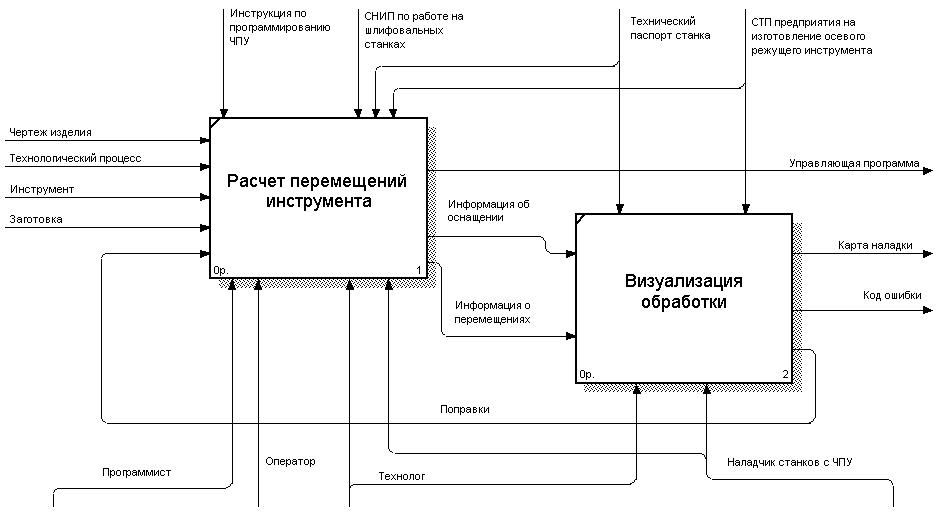


Рисунок 3 - Первый уровень функциональной модели

Второй модуль обеспечивает визуализацию обработки в анимированном, фотореалистичном виде. Данный модуль обеспечивает возможность наглядно проследить весь процесс обработки и введя необходимые поправки избежать получения бракованной продукции. Также этот модуль позволит провести процесс отладки и оптимизации программы на этапе ее создания.

Работать с этим модулем должны технолог и наладчик станков с ЧПУ. Работа модуля основывается на техническом паспорте станка и СТП предприятия на изготовление осевого режущего инструмента. Результатом работы данного модуля является создание карты наладки и проверка результата обработки на аномалии (отклонение размеров от заданных, геометрия поверхностей, наличие зарезов, как на изделии, так и на оснастке).

На втором уровне, модуль расчета перемещений инструмента можно разделить на четыре этапа (рисунок 4).

Первый этап – это моделирование заготовки [4, 5]. На нем задаются параметры заготовки (размеры, геометрические особенности, вылет из патрона). Все это производится на основании технического паспорта станка и СТП предприятия на изготовление осевого режущего инструмента.

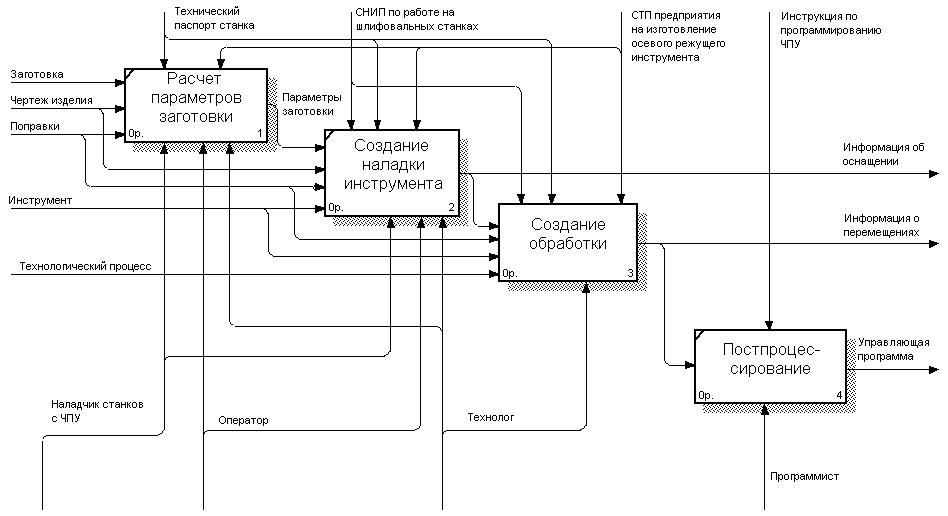


Рисунок 4 - Второй уровень функциональной модели

Заданием параметров занимается технолог, но в процессе работы параметры могут менять в определенных пределах наладчик станков с ЧПУ и оператор. После окончания ввода данных параметры заготовки в оцифрованном виде передаются на следующий этап. Следующий этап – моделирование наладки инструмента. В этом этапе задаются типы используемых кругов, их размеры и координатные привязки к системе координат станка. Профиль каждого круга должен быть поставлен в соответствие поверхности (или группе поверхностей) которую он обрабатывает, поэтому при подборе кругов и создании их наладки необходимо использовать данные чертежа изделия. При этом необходимо руководствоваться техническим паспортом станка, СТП предприятия на изготовление осевого режущего инструмента и СНИП по работе на шлифовальных станках. Вводом данных должны заниматься те же люди, что и на первом этапе. По результату этого этапа создается первая группа системных файлов, содержащих информацию об оснащении. Те же данные, что и в файле передаются на следующий этап. Третий этап – моделирование обработки. Самый важный этап, на котором создается траектория обработки. Траектория должна учитывать особенности технологического процесса, и применяемого инструмента. Ввод данных на этом этапе осуществляет только технолог. На этот этап оказывают влияние те же документы, что и на втором этапе. На этом этапе формируется группа системных файлов, содержащих информацию обо всех перемещениях инструмента и заготовки, та же информация передается на следующий этап. На трех выше перечисленных этапах при определенных условиях необходимо вводить поправки по результатам визуализации. Это сделано для того, чтобы система могла работать в условиях реального производства, где невозможно учесть всех случайностей, как то: отсутствие заказанного инструмента, отклонений заготовок, наличия оснастки и т.д. Последний этап – постпроцессирование. На этом этапе вся набранная системой информация преобразуется в управляющую программу системы ЧПУ. Преобразование осуществляется программистом на основании инструкции по программированию ЧПУ станка. Этап моделирования обработки в свою очередь можно разделить еще на два этапа (рисунок 5). Первый этап – анализ геометрии изделия. На этом этапе технолог производит настройку приложения для максимально объективного преобразования данных в графическую информацию. Сюда необходимо подать информацию об оснащении и об инструменте. Здесь технолог вводит все уточнения по конструкции изделия, которых не было в чертеже. Все уточнения вводятся на основании СТП предприятия на изготовление осевого режущего инструмента.

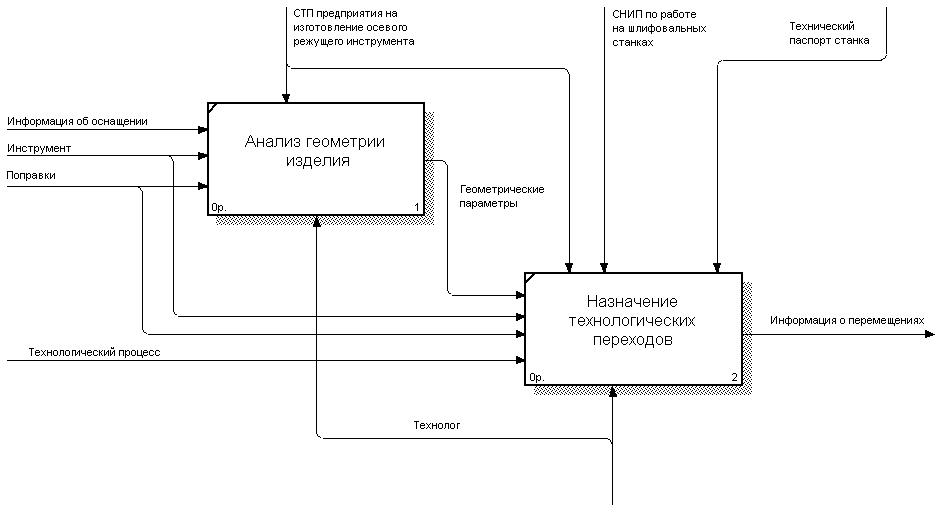


Рисунок 5 - Третий уровень функциональной модели

В результате проведенного анализа на следующий этап передаются все геометрические параметры в оцифрованном виде [24]. Второй этап – назначение технологических переходов. На этом этапе технолог выделяет отдельные элементарные операции в соответствии с технологическим процессом и применяемым инструментом. Определяет все траектории движений на каждой операции в соответствии с СТП предприятия на изготовление осевого режущего инструмента, СНИП по работе на шлифовальных станках и техническим паспортом станка. По итогам этого этапа выдается информация о перемещениях в группу системных файлов и на этап постпроцессирования.

## 2.2 Алгоритм работы программного модуля

Обобщенный алгоритм работы программного модуля мехатронной системы станка с ЧПУ показан на рисунке 6.

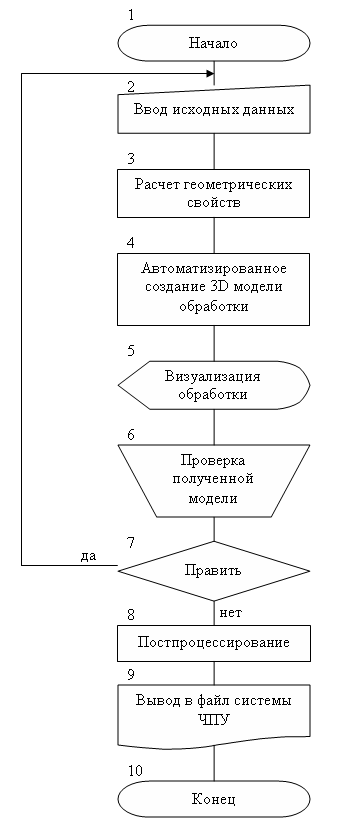


Рисунок 6 - Алгоритм работы программы

Первым этапом работы с программным модулем является ввод исходных данных. Исходные данные должны включать геометрические свойства изделия, заготовки, инструмента, особенности технологии обработки, конструктивные особенности оснастки и так далее [4, 5, 6, 7, 8, 9, 10]. Все эти параметры должны вводиться в удобной для пользователя форме. Каждый из этих параметров должен вводиться отдельно друг от друга, но иметь определенные связи между собой. Связи обеспечивают создание ограничений на вводимые данные.

Сразу после ввода данных программа запускается на расчет траекторий движения инструмента. Этот процесс должен быть максимально автоматизирован, и иметь определенные настраиваемые параметры, чтобы быть достаточно универсальным.

Траектории движений инструмента сохраняются в виде понятном только программному модулю. Чтобы заставить пакет 3D моделирования читать эти данные, их необходимо преобразовать в вид, понятный этому пакету. Затем эти данные запускаются на выполнение пакетом 3D моделирования.

В этом пакете вначале отрисовываются все основные элементы станка, находящиеся в зоне обработки, вся оснастка, рекомендованная к применению, а также инструмент и заготовка. Далее пакет, имитируя все перемещения инструмента относительно заготовки, осуществляет вычитание одного тела из другого. Таким образом, формируется наиболее приближенная к действительности 3D модель детали. С этой моделью можно производить целый ряд действий: промерить все основные размеры, оценить топологию построения поверхностей, произвести контроль зарезов оснастки, элементов станка и заготовки во время всех технологических переходов.

По результатам этой проверки принимается решение по правке исходных данных (величины подходов, отходов, врезаний, вылета заготовки, замена оснастки и инструмента). Этот процесс повторяется до тех пор, пока не достигается оптимальный результат, который удовлетворяет всем требованиям к данному изделию.

Когда пользователь данного программного продукта принимает решение о завершении правки исходных данных, он приступает в процессу постпроцессирования. Он заключается в преобразовании всей накопленной информации в программный код системы ЧПУ.

Этот программный код записывается в файл, который без дополнительных доработок может быть передан на станок.

Такой подход позволяет достаточно хорошо прорабатывать каждую программу для системы с ЧПУ и наглядно отслеживать весь процесс обработки, не прибегая к пробным деталям.

За счет использования стороннего пакета 3D моделирования можно значительно упростить создание программного модуля, за счет исключения из программного кода акселераторов 3D графики, заменив их неявным использованием уже готовых функций пакета 3D моделирования [25].

Так же пакет 3D моделирования обладает рядом полезных функций анализа моделей, которые позволяют проводить дополнительную проверку геометрических свойств изделия, а также функций, упрощающих создание сопутствующей документации.

## 2.3 Выбор программных средств реализации системы

В соответствии с функциональной моделью программный модуль состоит из двух частей: модуля расчета и модуля визуализации.

Для создания модуля расчета лучше всего подойдет язык программирования Delphi.

Delphi – это среда быстрой разработки, в которой в качестве языка программирования используется язык Delphi. Язык Delphi – строго типизированный объектно-ориентированный язык, в основе которого лежит хорошо знакомый программистам Object Pascal [23].

Бурное развитие вычислительной техники, потребность в эффективных средствах разработки программного обеспечения привели к появлению систем программирования, ориентированных на так называемую "быструю разработку", среди которых можно выделить Borland Delphi и Microsoft Visual Basic. В основе систем быстрой разработки (RAD-систем, Rapid Application Development – среда быстрой разработки приложений) лежит технология визуального проектирования и событийного программирования, суть которой заключается в том, что среда разработки берет на себя большую часть рутинной работы, оставляя программисту работу по конструированию диалоговых окон и функций обработки событий. Производительность программиста при использовании RAD-систем возрастает в несколько раз.

Изначально среда разработки была предназначена исключительно для разработки приложений Microsoft Windows, затем был реализован также для платформ Linux (как Kylix), однако после выпуска в 2002 году Kylix 3 его разработка была прекращена, и, вскоре после этого, было объявлено о поддержке MicrosoftNET.

Delphi – результат развития языка Турбо Паскаль, который, в свою очередь, развился из языка Паскаль. Паскаль был полностью процедурным языком, Турбо Паскаль, начиная с версии "5.5", добавил в Паскаль объектно-ориентированные свойства, а в Object Pascal – динамическую идентификацию типа данных с возможностью доступа к метаданным классов (то есть к описанию классов и их членов) в компилируемом коде, также называемом интроспекцией. Данная технология получила обозначение RTTI.

Также отличительным свойством Object Pascal от "С++" является то, что объекты по умолчанию располагаются в динамической памяти. Однако можно переопределить виртуальные методы NewInstance и FreeInstance класса TObject. Таким образом, абсолютно любой класс может осуществить возможность расположения системных и исполняемых файлов в любом удобном месте. Соответственно организуется и "многокучность" [11].

Система ЧПУ станка Walter CIP6 создана фирмой Siemens. Фирма Siemens является одним из ведущих производителей систем ЧПУ, занимая по объему продаж первое место в Европе и второе в мире. Эти показатели были достигнуты благодаря не только знаменитому немецкому качеству, но и аппаратным и технологическим возможностям систем управления [26].

Сегодня фирма Siemens предлагает две группы УЧПУ:

* семейство SINUMERIK 802C, 802S, 802D, которое ориентировано на применение в простых токарных и фрезерных станках (эти системы ЧПУ ограничены по количеству осей и имеют оптимальные функциональные возможности, соответствующие их назначению);
* семейство SINUMERIK 810D, 840D (SINUMERIK 810D – для станков с небольшими рабочими усилиями, SINUMERIK 840D – наиболее распространенное базовое модульное ЧПУ для широкого круга станков и технологических задач);

ЧПУ SINUMERIK 810D, 840D представляют собой интегрированные мультипроцессорные системы, в которых нельзя выделить ЧПУ и приводы в самостоятельные устройства. Пульт оператора состоит из следующих частей:

* MМС-процессора (Human Machine Communication), который представляет собой персональный компьютер ММС100.2 (Intel 486, MS-DOS) или ММС103 (Intel Pentium, Windows 95);
* дисплея, который может быть цветным или монохромным (10,4" TFT плоский экран).

Кроме того, пульт может быть оснащен полноразмерной клавиатурой, дисководом, адаптером подключения к сети Ethernet. Весь интерфейс полностью русифицирован. На ММС-процессоры можно установить дополнительное программное обеспечение. Это позволяет произвести объединение рабочих мест проектировщика (конструктора-технолога) и станочника-оператора.

Имеющиеся сегодня возможности позволяют говорить о действительном программировании, а не о кодировании перемещений, как это было раньше. Базовый набор технологических функций ЧПУ позволяет использовать его с широкой гаммой станков (токарные, фрезерные, шлифовальные и т.п. станки и обрабатывающие центры). Целый ряд высокоуровневых функций обеспечивает такие возможности как:

* функции позволяющие создавать плавные непрерывные кривые (в системе возможно использование трех видов сплайнов и кривых, заданных с помощью полиномов третьего порядка);
* изменение величины подачи по заданному закону в пределах одного кадра;
* автоматическое предотвращение зарезов;
* преобразование рабочей системы координат в пространстве;
* ориентация инструмента относительно плоскости обработки (при наличии на станке поворотной инструментальной головы при условии 2,5D-обработки значительно упрощается обработка наклонных поверхностей);
* ориентация инструмента относительно криволинейных поверхностей;
* задание запрещенных для перемещения зон на станке;
* создание программных конструкций, аналогичных языкам высокого уровня;
* вызов внешних подпрограмм при возникновении аварийных ситуаций (в случае возникновения нештатных ситуаций, например, при поломке инструмента, автоматически может быть вызвана специальная подпрограмма, внутри которой организуются все необходимые действия по устранению ситуации).

Язык программирования для ЧПУ содержит много элементов из языков программирования высокого уровня (больше всего он напоминает Паскаль). Программист может создавать переменные различных типов, использовать команды условных и безусловных переходов, арифметические и логические операции, циклы, выполняемые по условиям (WHILE – ENDWHILE; REPEAT – UNTIL; FOR – ENDFOR). Система предоставляет доступ к внутренним переменным (например, к текущей памяти положения, к значениям остатков пути внутри кадра и т.д.).

Имея эти возможности, можно создавать программы, построенные по принципу групповой технологии, для ввода новой детали необходимо просто заполнить таблицу внутри программы [12]. Это существенно облегчает работу оператора (простой запуск с промежуточного инструмента или технологического перехода).

Модуль визуализации слишком сложен для самостоятельного создания поэтому целесообразно воспользоваться стандартными средствами какой-нибудь CAD-системы. Самой удобной для этих целей является система Designer Modeling компании CoCreate. Отличительной особенностью данной системы является развитый набор функций для автоматизации процесса разработки моделей с использованием внешних макро команд, записанных во внешний файл.

В качестве языка макро команд система Designer Modeling использует язык LISP.

Лисп (LISP, от английского Processing language – "обработка списков") – семейство языков программирования, программы и данные в которых представляются системами линейных списков символов. Лисп является вторым в истории (после Фортрана) высокоуровневым языком программирования, который используется по сей день. Создатель Лиспа Джон Маккарти занимался исследованиями в области искусственного интеллекта и созданный им язык по сию пору является одним из основных средств моделирования различных аспектов искусственного интеллекта [23].

Традиционный Лисп имеет динамическую систему типов. Язык является функциональным, но многие поздние версии обладают также чертами императивности, к тому же, имея полноценные средства символьной обработки, становится возможным реализовать объектную ориентированность, примером такой реализации является платформа CLOS.

Язык Лисп, наряду с языком Ada, прошел процесс фундаментальной стандартизации для использования в военном деле и промышленности, в результате чего появился стандарт Common Lisp. Его реализации существуют для большинства платформ.

Одной из базовых идей языка Lisp является представление каждого символа как узла многокоординатной символьной сети; при этом координаты, свойства, уровни сети записаны в так называемых слотах символа.

Основной механизм Лиспа – инкапсулированная в список определяющая голова списка и подключённый к ней хвост списка, который рекурсивно также может быть списком. Лисп-машина способна воспринимать каждый поступающий на неё список на самом абстрактном уровне, например как мета-Лисп-машину, модифицирующую воспринимающую машину. В такой динамичной, высокоабстрактной среде можно реализовать как строго-научные системы, так и неисчислимое множество программистских трюков и генераторов всевозможных машин.

Любая программа на Лиспе состоит из последовательности выражений (форм). Результат работы программы состоит в вычислении этих выражений. Все выражения записываются в виде списков – одной из основных структур Лиспа, поэтому они могут легко быть созданы посредством самого языка. Это позволяет создавать программы, изменяющие другие программы или макросы, позволяющие существенно расширить возможности языка.

Внешне исходный код программы на Лиспе отличается обилием круглых скобок; редактирование программ значительно упрощается использованием текстового редактора, поддерживающего автоматическое выравнивание кода, подсветку соответствующих пар скобок и такие специальные команды, как "закрыть все открытые скобки", "перейти через список вправо" и т. д.

Символьная природа языка (то есть отсутствие в символьном пространстве традиционной метрической геометрии расстояний, последовательностей и т.д.) позволяет легко и продуктивно распараллеливать Лисп-процессы. Что нашло использование в сверхмощных телекоммуникационных, сетевых Лисп-системах.

Первые области применения Лиспа были связаны с символьной обработкой данных и процессами принятия решений.

Наиболее популярный сегодня диалект Коммон Лисп является универсальным языком программирования. Он широко используется в самых разных проектах: интернет-серверы и службы, серверы приложений и клиенты, взаимодействующие с реляционными и объектными базами данных, научные расчёты и игровые программы.

Одно из направлений использования языка Lisp – его использование в качестве скриптового языка, автоматизирующего работу в ряде прикладных программ.

Сферы применения Лиспа многообразны: наука и промышленность, образование и медицина, от декодирования генома человека до системы проектирования авиалайнеров.

# 3. Технологическая реализация системы подготовки обработки детали станка с ЧПУ

## 3.1 Описание кодов программного модуля

Любой проект в Delphi состоит из нескольких частей (набора файлов, каждый из которых отвечает за конкретную часть проекта). Это файлы, содержащие тексты кода, динамические библиотеки, файлы ресурсов и файл проекта, который содержит команды для объединения всех частей в единое целое. Все файлы создаются неявным образом, т.е. программист не должен задумываться, какие файлы ему надо создать.

При запуске Delphi автоматически создается новый проект, содержащий одну пустую форму (окно Windows, готовое для запуска, со стандартным набором функций) со всеми сопутствующими файлами. Это окно уже имеет функции закрытия, расширения до полного окна и сворачивания (рисунок 7).

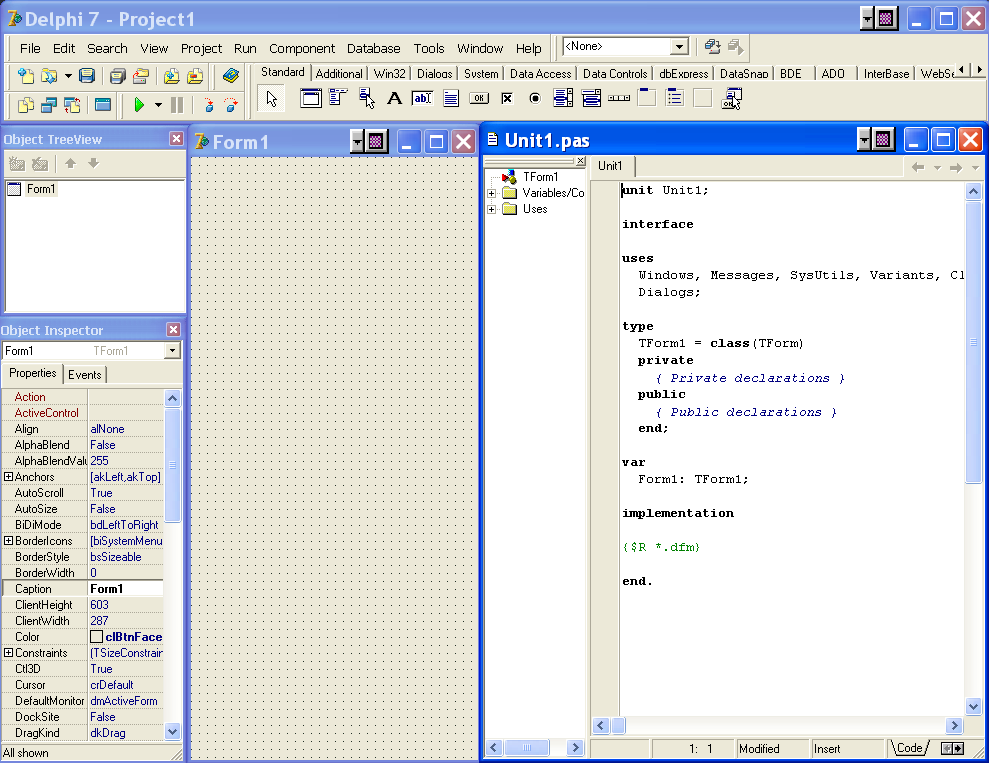


Рисунок 7- Новый проект Delphi

В простейшем случае проект состоит из файла описания проекта, файла главного модуля, файла ресурсов, файла описания формы, файла модуля формы, в котором находятся основной код приложения, в том числе функции обработки событий на компонентах формы, файл конфигурации.

Начинается главный модуль словом "program", за которым следует имя программы, совпадающее с именем проекта. Имя проекта задается в момент сохранения проекта, и оно определяет имя создаваемого компилятором исполняемого файла программы. Далее за словом "uses" следуют имена используемых модулей: библиотечного модуля "Forms" и модуля формы "vrunl.pas" [13].

Строка "{$R \*.RES}" — это директива компилятору подключить файл ресурсов. Файл ресурсов содержит ресурсы приложения: пиктограммы, курсоры, битовые образы и др. Звездочка показывает, что имя файла ресурсов такое же, как и у файла проекта, но с расширением "res".

Файл ресурсов не является текстовым файлом, поэтому просмотреть его с помощью редактора текста нельзя. Для работы с файлами ресурсов используют специальные программы (например "Resource Workshop").

Помимо главного модуля, каждая программа включает в себя еще как минимум один модуль формы, который содержит описание стартовой формы приложения и поддерживающих ее работу процедур [14].

Для разрабатываемого программного модуля необходимо создать 17 форм. При запуске программного модуля на экране будут отображаться только два из них, остальные будут вызываться при необходимости.

Основным окном проекта будет первое, появившееся при создании проекта, остальные окна будут присоединенными к проекту. Присоединение новых форм к проекту происходит по средствам команды "ShowMessage". По средствам этой же команды происходит и вывод любых нестандартных диалоговый окон (стандартные диалоговые окна выводятся с помощью команды "MessageDlg", с рядом настраиваемых параметров, таких как набор кнопок, выводящееся сообщение и маркер).

Технология программирования в Delphi основана на интуитивно понятном принципе. В начале строится интерфейс пользователя, заполнением формы различными стандартными компонентами с помощью мыши. Большинство визуальных параметров компонентов настраивается в "инспекторе объектов" (рисунок 8).

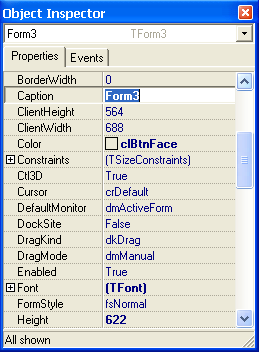


Рисунок 8 - Инспектор объектов

Он состоит из двух вкладок: "properties" и "events". На вкладке "properties" настраиваются такие параметры как заголовок, габариты, положение на форме, наполнение при запуске, видимость и множество других параметров. На вкладке "events" описываются события (результат действий, которые можно произвести над объектом). С помощью событий реализуются почти все обработчики данных.

Модуль состоит из следующих разделов: интерфейса, реализации, инициализации.

Раздел интерфейса (начинается словом "interface") сообщает компилятору, какая часть модуля является доступной для других модулей программы. В этом разделе перечислены (после слова "uses") библиотечные модули, используемые данным модулем. Также здесь находится сформированное Delphi описание формы, которое следует за словом "type".

Раздел реализации открывается словом "implementation" и содержит объявления локальных переменных, процедур и функций, поддерживающих работу формы. Начинается раздел реализации директивой "{$R \*.DFM}", указывающей компилятору, что в процессе генерации выполняемого файла надо использовать описание формы. Описание формы находится в файле с расширением "dfm", имя которого совпадает с именем модуля. Файл описания формы генерируется средой Delphi на основе внешнего вида формы. За директивой "{$R \*.DFM}" следуют процедуры обработки событий для формы и ее компонентов. Сюда же программист может поместить другие процедуры и функции.

Раздел инициализации позволяет выполнить инициализацию переменных модуля. Инструкции раздела инициализации располагаются после раздела реализации (описания всех процедур и функций) между "begin" и "end". Если раздел инициализации не содержит инструкций, то слово "begin" не указывается.

Для того чтобы в процессе набора текста программы воспользоваться шаблоном кода и вставить его в текст программы, нужно нажать комбинацию клавиш "<Ctrl>+<J>" и из появившегося списка выбрать нужный шаблон. Выбрать шаблон можно обычным образом, прокручивая список, или вводом первых букв имени шаблона (имена шаблонов в списке выделены полужирным).

В основном окне будет выводиться вся самая необходимая информация, которая может быть использована при редактировании исходных данных, для получения оптимального результата. В нем будут отображаться все координаты перемещения инструмента, а также в нем будут собраны все обработчики событий, отвечающие за обмен информацией между всеми элементами проекта. Также в этом окне будут располагаться все основные кнопки, отвечающие за создание файлов связи с программным продуктом 3D моделирования. Также есть возможность пошаговой отработки программы системы ЧПУ, которая должна быть представлена в верхней части окна (рисунок 9).

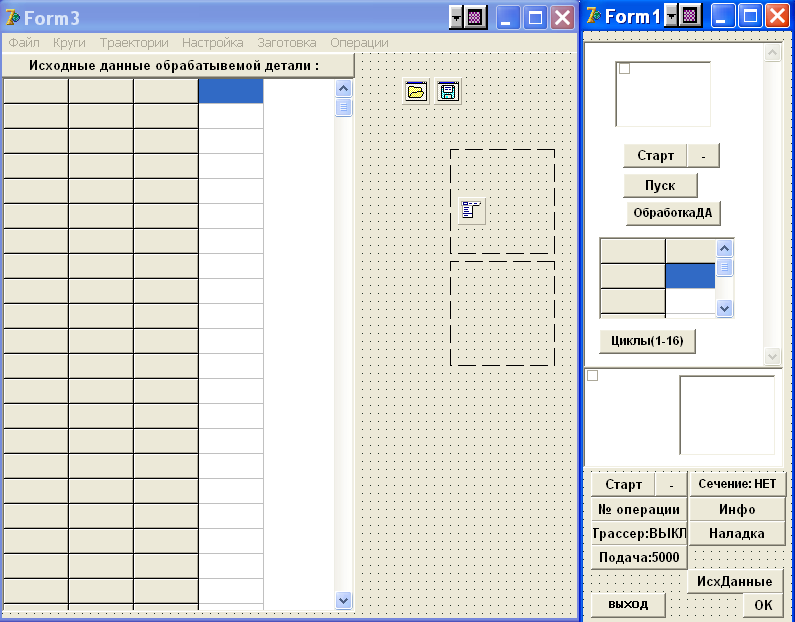


Рисунок 9 - Формы, появляющиеся при запуске программного модуля

Еще одна форма, появляющаяся при запуске программного модуля осуществляет накопление данных. Накопление данных происходит с помощью массива, который записывается в текстовый файл и читается из него по средствам элемента "Grid" (сетка таблицы). Этот элемент не обязателен для отображения, но важен для работоспособности программы в целом, поэтому ее можно сделать невидимой. Эта форма содержит также главное меню, по средствам которого происходит вызов форм для заполнения исходных данных.

Еще одной важной задачей сетки "Grid", является оперативное хранение введенных данных и позволяет осуществить реализацию постпроцессирования. Редактор кода выделяет ключевые слова языка программирования ("procedure", "var", "begin", "end", "if" и другие) полужирным шрифтом, что делает текст программы более выразительным и облегчает восприятие структуры программы (рисунок 10).

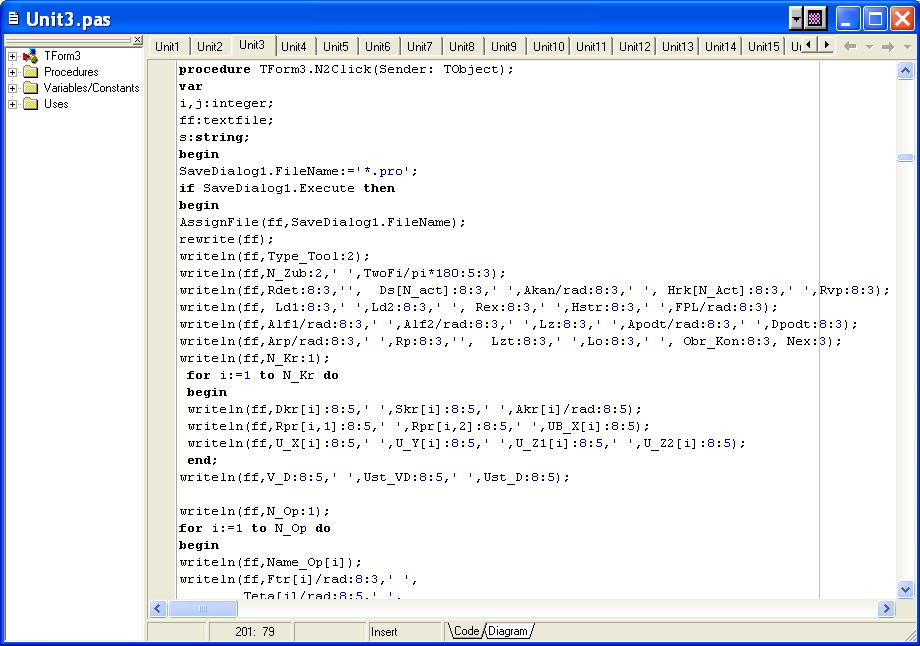


Рисунок 10 - Редактор кода программы

В процессе набора текста программы редактор кода выводит справочную информацию о параметрах процедур и функций, о свойствах и методах объектов. Один из параметров выделен полужирным. Так редактор подсказывает программисту, какой параметр он должен вводить. После набора параметра и запятой в окне подсказки будет выделен следующий параметр. И так до тех пор, пока не будут указаны все параметры.

Остальные формы отвечают за удобный ввод данных (параметры заготовки, инструмента, операций обработки, настройка технологических параметров и т.д.). Некоторые параметры требуют математических расчетов. Delphi со стандартными настройками может производить только самые простые математические действия. Чтобы вводить тригонометрические функции, а также сложные алгебраические действия необходимо подключить дополнительный модуль Delphi "Math". Сделано это для того, чтобы не загружать компилятор лишними данными и упростить программу, если она таких вычислений не требует. Модуль "Math" необходимо ввести в строку "Uses" редактора кода.

Из всех настроек можно выделить настройку самого постпроцессора для перевода в Lisp. Так как этот постпроцессор создает файл для пакета 3D моделирования, то параметры, выведенные в этот файл влияют на качество полученной модели. Модель в данном случае строится по направляющей и большому количеству сечений. При таком способе построения качество модели зависит от количества созданных сечений, и от качества каждого сечения в отдельности. Тут необходимо найти компромисс между качеством производительностью. Чем больше кривизна поверхностей модели, тем больше сечений надо, для получения качественной модели, но слишком большое количество сечений в свою очередь может перегрузить процессор даже достаточно мощного компьютера, что также не допустимо. На качество каждого сечения влияет количество элементарных отрезков и дуг, из которых оно состоит. Тут также слишком малое количество элементарных частей может привести к значительным отклонениям от действительных размеров, а слишком большое их количество может привести к математическим аномалиям (длина отрезка равна нулю или дуга образует внутреннюю петлю) [27].

Некоторые формы несут в себе только справочную информацию, и не влияют на программный модуль, другие формы не могут выводиться на экран, а несут лишь вспомогательную функцию. Они предназначены для создания пользовательского набора инструментов, наборов и типов данных, а также событий, уникальных для этого программного модуля.

Весь проект Delphi сохраняется в специально созданном для этого каталоге. В подкаталогах главного каталога сохраняются пользовательские файлы, на которые ссылается программный модуль (набор картинок, текстовые файлы, содержащие массивы данных, дополнительные библиотеки данных). Для проверки работоспособности программного модуля, в Delphi есть достаточно большой объем инструментальных средств. Это пошаговые компиляторы и отладчики с широким набором функций проверки правильности кода и т.п. При запуске программы на проверку, Delphi автоматически компилирует программный модуль в исполняемый файл (рисунок 11).

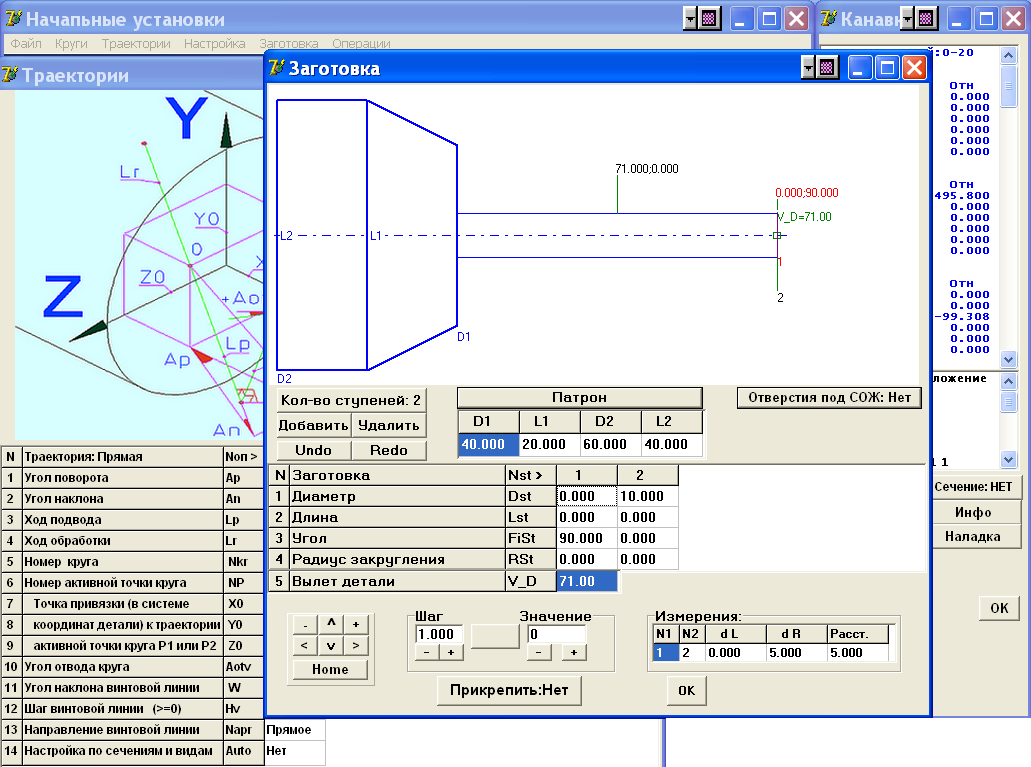


Рисунок 11 - Программный модуль

Компилятор генерирует исполняемую программу лишь в том случае, если исходный текст не содержит синтаксических ошибок. В большинстве случаев в только что набранной программе есть ошибки. Программист должен их устранить.

Процесс устранения ошибок носит итерационный характер. Обычно сначала устраняются наиболее очевидные ошибки, например, декларируются необъявленные переменные. После очередного внесения изменений в текст программы выполняется повторная компиляция.

После того как программа отлажена, необходимо выполнить ее окончательную настройку, т.е. задать название программы и выбрать значок, который будет изображать исполняемый файл приложения в папке или на рабочем столе, а также на панели задач во время работы программы [15].

## 3.2 Правила установки и настройки программного модуля

Процесс установки программы предполагает не только создание каталога и перенос в него выполняемых файлов и файлов данных с промежуточного носителя, но и настройку системы.

Для работы программного модуля, для начала необходимо установить на компьютер программный пакет One Space Designer Modeling 2005 и новее, созданный компанией CoCreate. Процесс установки этого пакета полностью описан в инструкции к дистрибутиву.

Далее в корне любого постоянного диска, необходимо создать каталог с названием "CIP6". Такое название было принято при создании программного модуля, поэтому менять его нельзя. Внутри этого каталога должен быть создан подкаталог "Emul". Сделано это для того, чтобы в каталоге "CIP6" можно было создать ряд других каталогов, в которых могла бы храниться информация, связанная со станком, но не связанная с программным модулем (требование заказчика).

В каталоге "Emul" создаются подкаталоги, содержащие служебные файлы.

Каталог "BMP" содержит файлы изображений, используемые для пояснений и графического оформления программного модуля.

Каталог "DAT" содержит файлы, с настроечными параметрами станка (значения абсолютных координат положения инструмента относительно заготовки в начальном положении). Эти параметры остаются неизменными в течение всего периода между ремонтами. В случае изменения этих значений достаточно открыть каждый из файлов как текстовый, вручную изменить значения этих параметров и сохранить изменения в том же файле. Сделано это для того, чтобы после ремонта станка, программа не нуждалась в повторной компиляции.

Каталог "LSP" содержит группу файлов, отвечающую за визуализацию процесса обработки в пакете 3D моделирования. Часть файлов являются неизменными, они содержат базовую геометрию, которая будет преобразована в частный случай. Это система координат, взаимное положение осей, параметрическая модель патрона и заготовки. Остальные файлы предназначены для изменения базовой геометрии. Эти файлы имеют переменное содержимое, и создаются каждый раз заново. Они строят круги, реальную заготовку, патрон, содержат информацию о перемещениях рабочих органов станка.

Также особой настройки требует и пакет 3D моделирования. Необходимо создать функцию загрузки базовой и дополнительной геометрии из файла (рисунок 12).

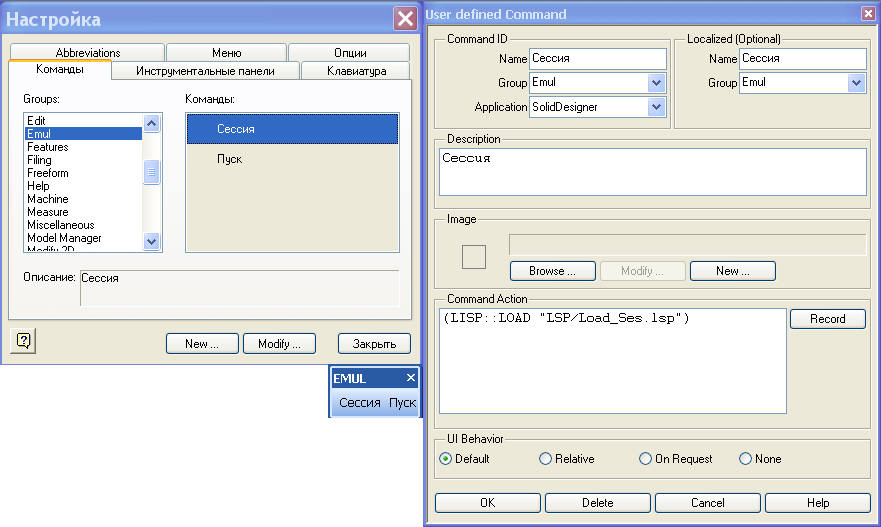


Рисунок 12 - Создание дополнительных функций

Ссылка идет на файл, в котором прописаны ссылки на группу других файлов, что упрощает настройку. После этого необходимо прописать в пакете 3D моделирования каталог "Emul", как рабочую директорию.

На этом процесс установки завершается.

## 3.3 Тестирование и отладка программного модуля

Первые программные системы разрабатывались в рамках программ научных исследований или программ для нужд министерств обороны. Тестирование таких продуктов проводилось строго формализовано с записью всех тестовых процедур, тестовых данных, полученных результатов. Тестирование выделялось в отдельный процесс, который начинался после завершения кодирования, но при этом, как правило, выполнялось тем же персоналом [23].

В тестировании программного модуля можно выделить четыре этапа:

* тестирование отдельных модулей;
* совместное тестирование модулей;
* тестирование спецификации программы;
* тестирование всего комплекса в целом (т.е. поиск несоответствия созданного программного продукта сформулированным ранее целям проектирования, отраженным в техническом задании).

Модульное тестирование (юнит-тестирование) – тестируется минимально возможный для тестирования компонент, например, отдельный класс или функция. Часто модульное тестирование осуществляется разработчиками программного обеспечения.

Интеграционное тестирование – тестируются интерфейсы между компонентами, подсистемами. При наличии резерва времени на данной стадии тестирование ведётся итерационно, с постепенным подключением последующих подсистем.

Системное тестирование – тестируется интегрированная система на её соответствие требованиям.

Альфа-тестирование – имитация реальной работы с системой штатными разработчиками, либо реальная работа с системой потенциальными пользователями и заказчиком. Чаще всего альфа-тестирование проводится на ранней стадии разработки продукта, но в некоторых случаях может применяться для законченного продукта в качестве внутреннего приёмочного тестирования. Иногда альфа-тестирование выполняется под отладчиком или с использованием окружения, которое помогает быстро выявлять найденные ошибки. Обнаруженные ошибки могут быть переданы специалистам по тестированию для дополнительного исследования в окружении, подобном тому, в котором будет использоваться программное обеспечение.

Бета-тестирование – в некоторых случаях выполняется распространение версии с ограничениями (по функциональности или времени работы) для некоторой группы лиц, с тем чтобы убедиться, что продукт содержит достаточно мало ошибок. Иногда бета-тестирование выполняется для того, чтобы получить обратную связь о продукте от его будущих пользователей.

Часто для свободного (открытого) программного обеспечения стадия альфа-тестирования характеризует функциональное наполнение кода, а бета-тестирования – стадию исправления ошибок. При этом, как правило, на каждом этапе разработки промежуточные результаты работы доступны конечным пользователям.

На первых двух этапах используются, прежде всего, методы структурного тестирования, так как на последующих этапах тестирования эти методы использовать сложнее из-за больших размеров проверяемого программного обеспечения. Последующие этапы тестирования ориентированы на обнаружение ошибок различного типа, которые не обязательно связаны с логикой программы. Известны два подхода к тестированию модулей: монолитное и пошаговое тестирование.

Существующие на сегодняшний день методы тестирования программного обеспечения не позволяют однозначно и полностью выявить все дефекты и установить корректность функционирования анализируемой программы, поэтому все существующие методы тестирования действуют в рамках формального процесса проверки исследуемого или разрабатываемого программного обеспечения. Такой процесс формальной проверки или верификации может доказать, что дефекты отсутствуют с точки зрения используемого метода.

Это означает, что нет никакой возможности точно установить или гарантировать отсутствие дефектов в программном продукте с учётом человеческого фактора, присутствующего на всех этапах жизненного цикла программного обеспечения.

Существует множество подходов к решению задачи тестирования и верификации ПО, но эффективное тестирование сложных программных продуктов – это процесс, в высшей степени, творческий, не сводящийся к следованию строгим и чётким процедурам или созданию таковых.

С точки зрения ISO9126, качество (программных средств) можно определить как совокупную характеристику исследуемого программного обеспечения с учётом следующих составляющих:

* надёжность;
* сопровождаемость;
* практичность;
* эффективность;
* мобильность;
* функциональность.

В терминологии профессионалов тестирования, фразы тестирование "белого ящика" и тестирование "чёрного ящика" относятся к тому, имеет ли разработчик тестов доступ к исходному коду тестируемого программного обеспечения, или же тестирование выполняется через пользовательский интерфейс либо прикладной программный интерфейс, предоставленный тестируемым модулем.

При тестировании белого ящика, разработчик теста имеет доступ к исходному коду программ и может писать код, который связан с библиотеками тестируемого ПО. Это типично для юнит-тестирования, при котором тестируются только отдельные части системы. Оно обеспечивает то, что компоненты конструкции работоспособны и устойчивы, до определённой степени. При тестировании белого ящика используются метрики покрытия кода.

При тестировании чёрного ящика, специалист по тестированию имеет доступ к ПО только через те же интерфейсы, что и заказчик или пользователь, либо через внешние интерфейсы, позволяющие другому компьютеру либо другому процессу подключиться к системе для тестирования. Например, тестирующий модуль может виртуально нажимать клавиши или кнопки мыши в тестируемой программе с помощью механизма взаимодействия процессов, с уверенностью в том, все ли идёт правильно, что эти события вызывают тот же отклик, что и реальные нажатия клавиш и кнопок мыши. Как правило, тестирование чёрного ящика ведётся с использованием спецификаций или иных документов, описывающих требования к системе. Как правило, в данном виде тестирования критерий покрытия складывается из покрытия структуры входных данных, покрытия требований и покрытия модели (в тестировании на основе моделей).

При тестировании серого ящика разработчик теста имеет доступ к исходному коду, но при непосредственном выполнении тестов доступ к коду, как правило, не требуется.

Покрытие кода, по своей сути, является тестированием методом белого ящика. Тестируемое ПО собирается со специальными настройками или библиотеками и запускается в особом окружении, в результате чего для каждой используемой (выполняемой) функции программы определяется местонахождение этой функции в исходном коде. Этот процесс позволяет разработчикам и специалистам по обеспечению качества определить части системы, которые, при нормальной работе, используются очень редко или никогда не используются (такие как код обработки ошибок и т.п.). Это позволяет сориентировать специалистам по тестированию на тестирование наиболее важных режимов.

Специалисты по тестированию могут использовать результаты теста покрытия кода для разработки тестов или тестовых данных, которые расширят покрытие кода на важные функции.

Как правило, инструменты и библиотеки, используемые для получения покрытия кода, требуют значительных затрат производительности и памяти, недопустимых при нормальном функционировании ПО, поэтому они могут использоваться только в лабораторных условиях.

## 

## 3.4 Инструкция пользователя

При использовании программного модуля пользователь должен обладать полным комплектом конструкторской и технологической документации. Работа начинается с моделирования заготовки, модуль для которого показан на рисунке 13.

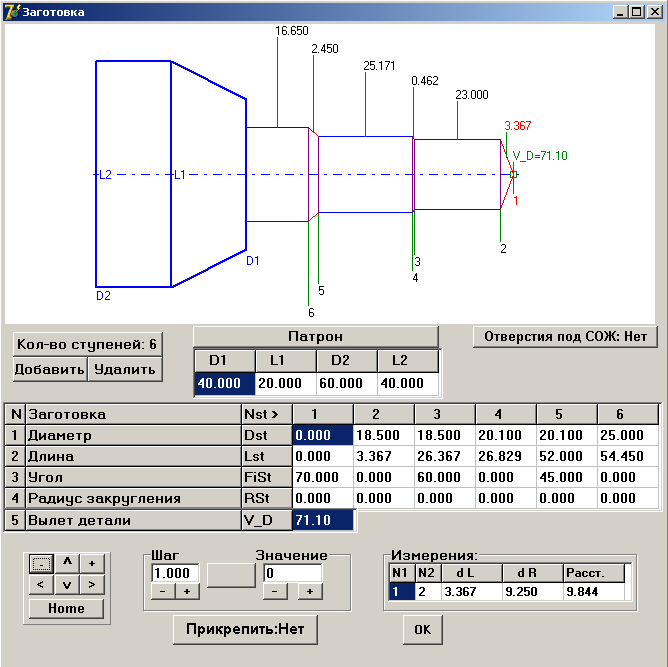


Рисунок 13 - Модуль моделирования заготовки

Форма и положение заготовки относительно патрона имеют первостепенное значение для проведения корректной обработки шлифованием, исключения брака и аварийных ситуаций. Предложенная схема введения размеров заготовки обеспечивает создание тела вращения различной конфигурации, образующими формы которого могут быть наборы прямых линий и сопряженных с ними окружностей. Ступенью в данной схеме является отдельно взятый отрезок прямой с началом в точке, обозначенной числом равным номеру ступени. Эти числа расположены вместе с вертикальными указателями на нижней части профиля заготовки. Все изменения параметров патрона и заготовки отображаются на верхнем графическом поле. Изменение значения параметра производится непосредственно в соответствующей ячейке таблицы или с помощью вспомогательных параметров "Шаг" и "Значение". В верхней части заготовки проставлены указатели с расстояниями между соседними ступенями и значениями углов указанной ступени к оси заготовки. В нижнем правом углу, в блоке "Измерения" находятся значения осевых, радиальных перепадов и расстояний между двумя произвольными точками контура заготовки. Кнопка "Прикрепить" имеет три режима: "Нет", "Справа" и "Слева". Это означает, что при изменении параметров контур справа или слева от рассматриваемой ступени может меняться по-разному. Добавление и удаление ступеней желательно не делать на первой и последней ступенях, иначе может произойти разрушение топологии контура заготовки. В случае любой непонятной ситуации можно применить откат вперед или назад (до 20 шагов) кнопками "Undo" и "Redo". Для заготовок с отверстиями под СОЖ предусмотрена упрощенная схема отображения отверстий в графическом поле. После ввода всех необходимых параметров и нажатия кнопки "ОК", генерируется программа для создания 3-d моделей патрона и заготовки. Следующим этапом является создание наладки инструмента (рисунок 14).

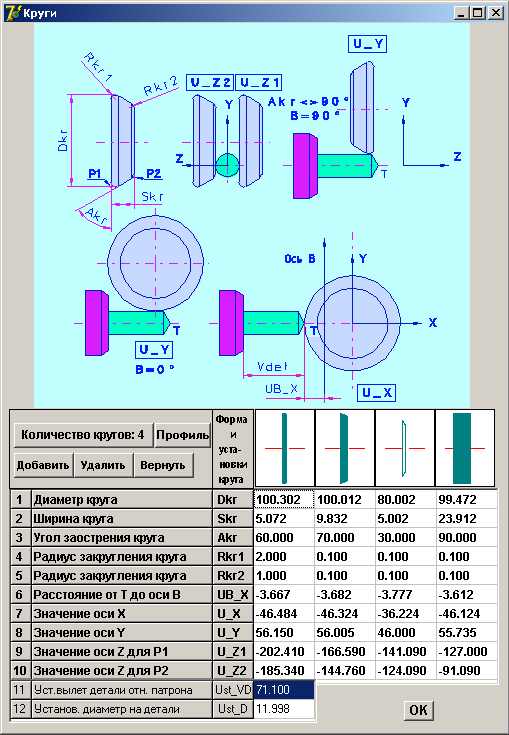


Рисунок 14 - Модуль моделирования наладки инструмента

Эта часть программного модуля использует данные, полученные после замеров кругов на станке. Количество кругов, их форма и расположение на оправке определяются шлифовщиком и технологом, исходя из их знаний и опыта. Шлифовщику необходимо замерить на каждом круге все соответствующие размеры и записать их в бланк замеров [4, 5, 6, 7, 8, 9, 10, 11, 12]. Все замеры шлифовщик должен производить вручную, связано это с индивидуальными особенностями каждого круга, которые невозможно учесть на предварительном этапе. После ввода всех необходимых параметров и нажатия кнопки "ОК", генерируется программа для создания 3D моделей кругов и оправки. Дальше наступает самая главная часть процесса разработки – создание траекторий перемещений рабочих органов станка (рисунок 15).

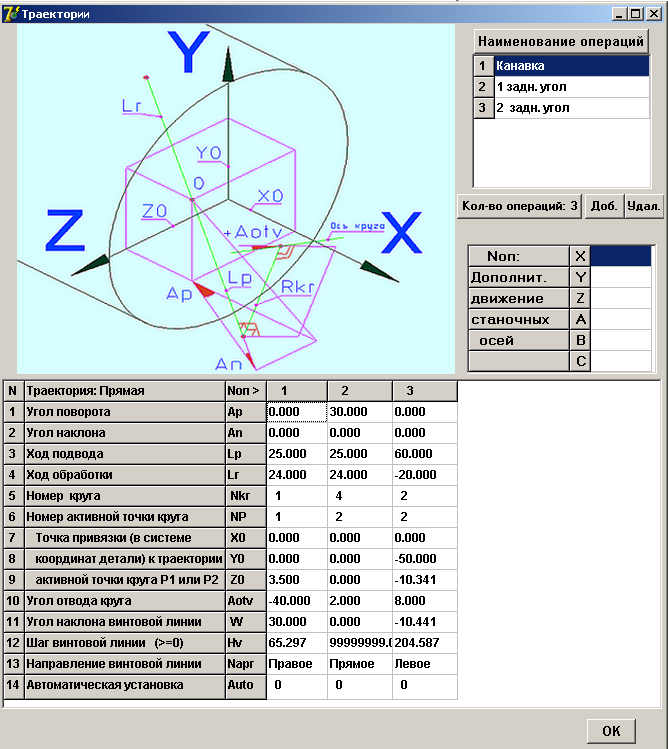


Рисунок 15 - Модуль моделирования траектории обработки

Операция – это совокупность движений одной или нескольких осей станка при обработке одним кругом поверхностей одного типа. Например, для обычного сверла необходимо 4 операции: обработка винтовой канавки, формирование ленточки, заточка задней поверхности и подточка передней поверхности. При этом деление по числу зубьев, подвод и отвод круга, распределение припуска и выхаживание могут происходить внутри одной операции. Кроме того, операция может содержать и дополнительные циклические движения осей для реализации, например, ступенчатой обработки затылка у сверла.

Любая линия образуется движением точки. На шлифовальном круге выбирается активная точка и по ней фиксируется траектория движения всего круга. Ориентация круга такова, что окружность, образованная активной точкой, касается траектории, а ось круга и касательная к траектории в активной точке расположены под углом. Такое положение круга позволяет ему двигаться вдоль траектории, а угол отвода должен обеспечить отсутствие задевания обработанной поверхности задней, по отношению к направлению движения, частью круга.

Как правило, к обработанной поверхности обращена часть профиля круга с активной точкой. Но иногда требуется обработать поверхность полным профилем или с присутствием задевания задней частью круга. В этом случае нужно значительно уменьшать скорость подачи круга в зоне резания.

Для обработки винтовых поверхностей (цилиндрических или конических) с постоянным шагом необходимо дополнительное вращение вокруг оси заготовки при рабочем ходе.

В ручном режиме заполнения предлагается сформировать достаточно большие группы операций (до 20 операций) для обработки, например, нестандартных видов осевого инструмента. Затем это может быть прототипом для дальнейшего использования. Каждая выбранная операция может быть представлена в анимированном виде в пакете 3D моделирования.

Последний этап работы с программным модулем – настройка технологических циклов (рисунок 16).

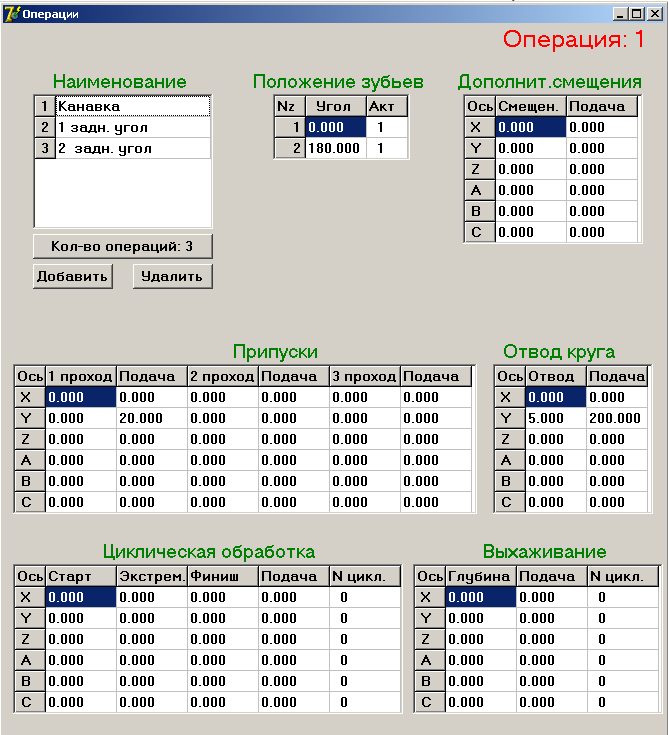


Рисунок 16 - Модуль настройки операций

Внутри каждой операции существуют два типа движения шлифовального круга: со снятием материала и без снятия материала с заготовки (главное и вспомогательное). Как правило, все движения имеют циклический характер. Данный модуль предлагает пооперационное управление технологическими циклами.

Переключение, добавление или удаление операций производится в блоке "наименование". Здесь же можно изменить и наименование операции.

"Положение зубьев" – для угловой разбивки положения зубьев и назначение обработки на каждый зуб.

"Дополнит. смещения" – зависят от расположения обрабатываемой поверхности и предназначены для смещения по одной из осей траектории движения шлифовального круга.

"Припуски" назначаются на сторону и максимум на 3 прохода вместе с подачей. Величина последнего припуска равна нулю. Если подача равна нулю, то припуск не учитывается. Величина припуска определяет расстояние, на которое отодвинут круг вдоль назначенной оси от окончательного расчетного положения.

"Отвод круга" используется для возврата круга в исходное положение относительно детали после обработки. Величина отвода и подача назначаются применительно к одной или нескольким (до 3-х) осям одновременно.

"Циклическая обработка" – комплексное перемещение шлифовального круга для обработки поверхностей методом обкатывания. Круг проходит через три положения: старт, экстремум и финиш. Экстремумом может быть любое промежуточное значение выбранной оси. Количество циклов (до 20) равномерно распределяется на все значения от старта до финиша. В результате получается ступенчатая поверхность с регулируемой шероховатостью. Подача на каждую ось задается отдельно.

"Выхаживание" – стандартное циклическое движение шлифовального круга для получения более качественной по шероховатости обработанной поверхности. Ось для выхаживания может быть выбрана только одна.

После заполнения всех таблиц и формирования готового осевого инструмента в пакете 3D моделирования можно создать управляющую программу для станка.

# 4. Расчет экономической эффективности

## 4.1 Расчет затрат на разработку системы

В таблице 4.1 представлены сведения о численности работающих, принимавших участие в разработке программного изделия.

Таблица 4.1 - Ведомость фонда оплаты труда

|  |  |  |  |
| --- | --- | --- | --- |
| Категория работающих | Количество, чел. | Должностной оклад, руб. | Фонд заработной платы, руб. |
| Программист | 1 | 10500 | 39690 |
| Конструктор | 1 | 13200 | 49896 |

Затраты на разработку программного обеспечения (Кп) определяется по формуле:

, (4.1)



где

Кпр – затраты на проектирование программного обеспечения (ПО), руб.;

Кпо – затраты на создание программных изделий, образующих ПО, руб.;

Кио – затраты на подготовку информационного обеспечения длительного пользования, создания базы данных ПО, руб.;

Ко – Затраты на отладку ПО, руб.

Укрупненный расчет на разработку ПО можно выполнить по формуле:

, (4.2)



где

Фз/п – фонд основной заработной платы разработчиков и других исполнителей работы, руб.;

βд – коэффициент дополнительной зарплаты (принимается 0,1 - 0,15);

βс – коэффициент отчислений на социальные нужды от основной и дополнительной заработной платы (принимается 0,26);

βн – коэффициент накладных расходов организации, разрабатывающей проект (принимается 0,6 - 0,8);

βпр – коэффициент прочих расходов (принимается 0,1 - 0,2);

tэвм – машинное время, затраченное для отладки программного обеспечения, ч;

Cм-ч – стоимость машино-часа работы ЭВМ, руб.



Кп=245728,91 руб.

4.1.1 Расчет фонда заработной платы

Фонд заработной платы консультанта и разработчика программного модуля с учетом отчислений на социальные нужды рассчитывается:

, (4.3)



где

Вр – время на проектирование и создание программного модуля, мес.;

О – среднемесячный оклад разработчиков, руб.;

Ксн – коэффициент расходов на социальные нужды.

Среднемесячный оклад разработчика принимается в соответствии с окладами организации, где выполняется проект.

В результате чего получим фонд заработной платы программиста:

;



руб.



Фонд заработной платы конструктора:

;



руб.



Общий фонд заработной платы:

руб.



Время, затраченное на отладку программного обеспечения на ЭВМ tэвм устанавливается экспертным путем или по фактическим затратам машинного времени. Оно составило 40% от длительности разработки программного модуля.

tэвм= 3∙0,4∙168;

tэвм= 201,6 ч.

4.1.2 Расчет себестоимости машино-часа работы персонального компьютера

Стоимость машино-часа работы персонального компьютера или комплекса средств автоматизации См-ч берется в бухгалтерии той организации, где ведется разработка программного модуля [16]. При отсутствии этих данных необходимо выполнить расчет себестоимости машино-часа работы с учетом конкретных условий.

Себестоимость машино-часа работы ЭВМ определяется по формуле:

, (4.4)



где

Зп – затраты на заработную плату обслуживающего персонала с учетом всех отчислений, руб.;

А – годовая сумма амортизации, руб.;

Зэ – затраты на силовую электроэнергию, руб.;

Зр – затраты на ремонт и обслуживание оборудования в год, руб.;

Зм – затраты на материалы в год, руб.;

Зн – накладные расходы, руб.;

Фд – действительный годовой фонд времени работы ЭВМ, ч.

Расчет затрат на заработную плату обслуживающего персонала производится по формуле:

, (4.5)



где

n – количество работников (принимается 1 работник);

li – месячный оклад работника (принимается 9800 руб);

kд – коэффициент, учитывающий дополнительную заработную плату (принимается от 1,1 до 1,2); kсн – коэффициент, учитывающий отчисления на социальные нужды (в соответствии с законодательством РФ kсн=1,26).

Зп=9800∙12∙1,1∙1,26;

Зп=162993,6 руб.

Расчет стоимости персонального компьютера для разработки программного модуля представлен в таблице 4.2.

Таблица 4.2 - Расчет стоимости ЭВМ для разработки ПО

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование оборудования | Модель, серия | Кол-во, шт. | Стоимость, т.руб. | |
| ед. | всего |
| Системный блок | Профессиональная графическая станция Compaq | 1 | 36200 | 36200 |
| Монитор | Compaq P1210 | 1 | 13100 | 13100 |
| Принтер | HP LaserJet P2015 | 1 | 11500 | 11500 |
| Итого | | | | 60800 |

Годовые амортизационные отчисления по ЭВМ считаются по формуле:

, (4.6)



где

Соб – стоимость персонального компьютера и прочего оборудования, используемого при разработке и отладке программного модуля;

На – норма амортизации, %.

Общая норма амортизации (На) основного фонда вычислительной техники составляет 35%.

При подстановке данных из таблицы 4.2 в формулу 4.6 получаем следующие значения:

;



А=21280 руб.

Годовые затраты на электроэнергию рассчитываются следующим образом:

Зэ = N ∙ Т ∙ Кисп ∙ Цэл, (4.7)

Где N – мощность оборудования, кВт;

Т – время работы оборудования, ч.;

Кисп – коэффициент использования оборудования по мощности, принимаем 0,75; Цэл – стоимость 1 кВт\*ч электроэнергии, руб.

В среднем вычислительная техника потребляет 0,4 кВт электроэнергии в час. Время потребления энергии ЭВМ составляет 1488 часов. Стоимость силовой электроэнергии равна 1,62 руб/кВт.

Зэ=0,4∙1488∙0,75∙1,62;

Зэ=723,17 руб.

Затраты на текущие ремонты Зр и на материалы Зм в год берутся по данным бухгалтерии. При укрупненном расчете их сумма может быть принята от 4,5 до 10% от стоимости ЭВМ.

Зр=60800∙0,06;

Зр=3648 руб.

В накладные расходы включаются затраты на амортизацию и содержание площадей, затраты на отопление, освещение и прочие [2].

Коэффициент накладных расходов организации, разрабатывающей проект принимаются равным 0,6 - 0,9 от затрат на основную заработную плату разработчиков.

Зн=89586∙0,8;

Зн=71668,8 руб.

Годовой фонд времени Фд устанавливается, исходя из номинального фонда времени и времени профилактики оборудования и ремонтов:

Фд = (365-Пр-В)∙q∙S∙Kисп, (4.8)

где

Пр – количество праздничных дней;

В – количество выходных дней;

S – количество смен;

q – продолжительность смены, ч.

Фд=248∙8∙1∙0,75,

Фд=1488 ч.

Подставив подсчитанные значения в формулах 4.5 - 4.8 в формулу 4.4 можно вычислить стоимость машино-часа работы ЭВМ:

См-ч=(162993,6 + 21280 + 723,17 + 3648 + 71668,8)/1488;

См-ч=174,94 руб/час.

4.1.3 Расчет цены программного обеспечения

Минимальная цена разработки программного обеспечения (Zmin) складывается из полных затрат на разработку Кп и минимально необходимой суммы прибыли Пmin и рассчитывается по формуле:

Zmin=Кп+Пmin . (4.9)

Сумма прибыли Пmin рассчитывается исходя из планируемого минимального уровня, рентабельности затрат организации-разработчика:

Пmin=Кп∙Rmin/100 (4.10)

где

Rmin – минимальный уровень рентабельности, % (20-30%);

Пmin= 245728,91∙0,25;

Пmin= 61432,23 руб.

Подставив полученные значения из формулы 4.10 в формулу 4.9 можно рассчитать минимальную цену разработки программного модуля:

Zmin= 245728,91+ 61432,23;

Zmin= 307161,14 руб.

После выполнения расчетов, можно подвести итоги по подсчету себестоимости программы:

* Время, затраченное на разработку и отладку программного модуля на персональном компьютере составляет 3 месяца;
* Затраты на разработку программного модуля составляет 245728,91 рублей;
* Фонд заработной платы составляет 89586 рублей;
* Затраты на зарплату обслуживающего персонала составляют 162993,6 рубля;
* Затраты на электроэнергию составляют 723,17 рубля;
* Затраты на текущие ремонты составляют 3648 рублей;
* Накладные расходы составляют 71668,8 рублей;
* Стоимость машино-часа работы персонального компьютера составляют 174,94 рубля/час.

Исходя из вышеперечисленных затрат, можно сказать что стоимость программного модуля, которая составляет 307161,14 рубль, является достаточно реальной суммой, за которую можно продать данный программный продукт.

## 

## 4.2 Методология расчета общей стоимости владения программным продуктом

Расчет стоимости владения программным модулем включает определение затрат на каждой стадии жизненного цикла информационного продукта их классификацию и обоснование [12]. Формулы для расчета стоимости владения программным модулем приведены ниже.

Стоимость теоретического проекта рассчитывается:

С т.пр = Zmin + С под ,

где Спод – стоимость поддержки, которая складывается из расходов на услуги консультантов. Принимается время работы консультантов поддержки равным 50% от времени работы при создании [19]. Тогда время работы консультанта равно 1,5 мес. При размере оплаты услуг консультанта 10000 руб/мес, стоимость поддержки составит 15000 руб.

С т.пр = 307161,14+15000;

С т.пр = 322161,14 руб.

Стоимость технологического проекта рассчитывается:

С тех.пр = Р ап + С под ,

где Рап – расходы на аппаратное обеспечение. Спод – стоимость поддержки, которая складывается из расходов на услуги программистов. Принимается время работы программистов равным 50% от времени работы при создании. Тогда время работы программиста равно 1,5 мес. При размере оплаты услуг программиста 10500 руб/мес, стоимость поддержки составит 15750 руб.

С тех.проекта = 60800 + 15750;

С тех.проекта = 76550 руб.

Общая стоимость владения информационным продуктом рассчитывается:

ОСВ = С т.пр + С тех.пр,

ОСВ = 322161,14 + 76550;

ОСВ = 398711,14 руб.

## 

## 4.3 Экономический эффект

Разработанный программный продукт, программный модуль станка с ЧПУ, необходим для автоматизации процесса создания управляющей программы для станка Walter CIP6, оснащенного системой ЧПУ Sinumeric 840D. При расчёте экономического эффекта от применения программного продукта, общая стоимость разработанного программного модуля станка с ЧПУ оказалась ниже стоимости аналогичного модуля, предлагаемого фирмой-изготовителем станка [18].

, (4.11)



где ОВСр – общая стоимость владения разработываемого программмного продукта, руб.; ОВСп – общая стоимость владения покупного программного продукта, руб.

ОВСп = 638540 руб.;

Э – экономический эффект, руб.

Э = 638540 - 398711,14;

Э = 239828,86 руб.

Разработанный программный продукт, программный модуль станка с ЧПУ, на 239828,86 рублей дешевле аналогичного программного модуля разработанного фирмой фирмой-изготовителем станка.

# 5. Обеспечение безопасности жизнедеятельности

## 5.1 Общие сведения о безопасности жизнедеятельности

Безопасность жизнедеятельности (БЖД) – это комплекс мероприятий, направленных на обеспечение безопасности человека в среде обитания, сохранение его здоровья, разработку методов и средств защиты путем снижения влияния вредных и опасных факторов до допустимых значений, выработку мер по ограничению ущерба в ликвидации последствий чрезвычайных ситуаций мирного и военного времени [20].

Охрана здоровья трудящихся, обеспечение безопасности условий труда, ликвидация профессиональных заболеваний и производственного травматизма составляет одну из главных забот человеческого общества. Обращается внимание на необходимость широкого применения прогрессивных форм научной организации труда, сведения к минимуму ручного, малоквалифицированного труда, создания обстановки, исключающей профессиональные заболевания и производственный травматизм.

Помещения для эксплуатации персональных компьютеров должны иметь естественное и искусственное освещение. Эксплуатация персональных компьютеров в помещениях без естественного освещения допускается только при соответствующем обосновании и наличии положительного санитарно-эпидемиологического заключения, выданного в установленном порядке.

Естественное и искусственное освещение должно соответствовать требованиям действующей нормативной документации. Окна в помещениях, где эксплуатируется вычислительная техника, преимущественно должны быть ориентированы на север и северо-восток.

Оконные проемы должны быть оборудованы регулируемыми устройствами типа: жалюзи, занавесей, внешних козырьков и так далее.

Площадь на одно рабочее место комплектовщика компьютера на базе электронно-лучевой трубки (ЭЛТ) должна составлять не менее 6 м2, на базе плоских дискретных экранов (жидкокристаллические, плазменные) – 4,5 м2.

Для внутренней отделки интерьера помещений, где расположены компьютеры, должны использоваться диффузно отражающие материалы с коэффициентом отражения для потолка 0,7-0,8; для стен 0,5-0,6; для пола 0,3-0,5. Полимерные материалы используются для внутренней отделки интерьера помещений с персональными компьютерами при наличии санитарно-эпидемиологического заключения.

Помещения, где размещаются рабочие места с персональными компьютерами, должны быть оборудованы защитным заземлением (занулением) в соответствии с техническими требованиями по эксплуатации.

Не следует размещать рабочие места с персональными компьютерами вблизи силовых кабелей и вводов, высоковольтных трансформаторов, технологического оборудования, создающего помехи в работе персональных компьютеров.

В производственных помещениях, в которых работа с использованием персональных компьютеров является вспомогательной, температура, относительная влажность и скорость движения воздуха на рабочих местах должны соответствовать действующим санитарным нормам микроклимата производственных помещений.

В производственных помещениях, в которых работа с использованием персональных компьютеров является основной (диспетчерские, операторские, расчетные, кабины и посты управления, залы вычислительной техники и др.) и связана с нервно-эмоциональным напряжением, должны обеспечиваться оптимальные параметры микроклимата для категории работ 1а и 1б в соответствии с действующими санитарно-эпидемиологическими нормативами микроклимата производственных помещений. На других рабочих местах следует поддерживать параметры микроклимата на допустимом уровне, соответствующем требованиям указанных выше нормативов.

В помещениях должна проводиться ежедневная влажная уборка и систематическое проветривание после каждого часа работы на компьютере.

Уровни положительных и отрицательных аэроионов в воздухе помещений, где расположены персональные компьютеры, должны соответствовать действующим санитарно-эпидемиологическим нормативам.

Содержание вредных химических веществ в воздухе производственных помещений, в которых работа с использованием персональных компьютеров является вспомогательной, не должно превышать предельно допустимых концентраций вредных веществ в воздухе рабочей зоны в соответствии с действующими гигиеническими нормативами.

Содержание вредных химических веществ в производственных помещениях, в которых работа с использованием персональных компьютеров является основной (диспетчерские, операторские, расчетные, кабины и посты управления, залы вычислительной техники и др.), не должно превышать предельно допустимых концентраций загрязняющих веществ в атмосферном воздухе населенных мест в соответствии с действующими гигиеническими нормативами.

В производственных помещениях при выполнении основных или вспомогательных работ с использованием персональных компьютеров уровни шума на рабочих местах не должны превышать предельно допустимых значений, установленных для данных видов работ в соответствии с действующими санитарно-эпидемиологическими нормативами.

При выполнении работ с использованием персональных компьютеров в производственных помещениях уровень вибрации не должен превышать допустимых значений вибрации для рабочих мест (категория 3, тип "в") в соответствии с действующими санитарно-эпидемиологическими нормативами.

Шумящее оборудование (печатающие устройства, серверы и т.п.), уровни шума которого превышают нормативные, должно размещаться вне помещений с персональными компьютерами.

Рабочие столы следует размещать таким образом, чтобы видеодисплейные терминалы были ориентированы боковой стороной к световым проемам, чтобы естественный свет падал преимущественно слева.

Искусственное освещение в помещениях для эксплуатации персональных компьютеров должно осуществляться системой общего равномерного освещения. В производственных и административно-общественных помещениях, в случаях преимущественной работы с документами, следует применять системы комбинированного освещения (к общему освещению дополнительно устанавливаются светильники местного освещения, предназначенные для освещения зоны расположения документов).

Освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300 - 500 лк. Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 лк. Следует ограничивать прямую блесткость от источников освещения, при этом яркость светящихся поверхностей (окна, светильники и другие), находящихся в поле зрения, должна быть не более 200 кд/м2.

Следует ограничивать отраженную блесткость на рабочих поверхностях (экран, стол, клавиатура и другие) за счет правильного выбора типов светильников и расположения рабочих мест по отношению к источникам естественного и искусственного освещения, при этом яркость бликов на экране персонального компьютера не должна превышать 40 кд/м2 и яркость потолка не должна превышать 200 кд/м2.

Показатель ослепленности для источников общего искусственного освещения в производственных помещениях должен быть не более 20. Показатель дискомфорта в административно-общественных помещениях не более 40.

Яркость светильников общего освещения в зоне углов излучения от 50 до 90 градусов с вертикалью в продольной и поперечной плоскостях должна составлять не более 200 кд/м2, защитный угол светильников должен быть не менее 40 градусов. Светильники местного освещения должны иметь не просвечивающий отражатель с защитным углом не менее 40 градусов.

Следует ограничивать неравномерность распределения яркости в поле зрения пользователя персонального компьютера, при этом соотношение яркости между рабочими поверхностями не должно превышать 3:1 - 5:1, а между рабочими поверхностями и поверхностями стен и оборудования 10:1.

Для обеспечения нормируемых значений освещенности в помещениях для использования персональных компьютеров следует проводить чистку стекол оконных рам и светильников не реже двух раз в год и проводить своевременную замену перегоревших ламп.

Режим пожарной безопасности помещений регламентируется двумя основополагающими документами: ФЗ № 69 "О пожарной безопасности" и "Правилами пожарной безопасности на ОАО "АВТОВАЗ".

Правила пожарной безопасности устанавливают основные нормы и требования к помещениям, размещению в них оборудования. Оснащение помещений средствами первичного пожаротушения, планы и схемы эвакуации при пожаре, порядок вызова пожарной службы при возникновении ЧС. В каждом помещении должно быть лицо ответственное за пожарную безопасность, назначенное приказом руководителя предприятия или лицом ответственным за пожарную безопасность предприятия в целом. Все работники должны периодически, не реже 1 раза в полугодие, проходить инструктаж по пожарной безопасности, участвовать в тренировках по эвакуации из здания при пожаре, тушению предполагаемых возгораний.

Требования Правил пожарной безопасности к эксплуатации вычислительной техники содержат лишь общие положения, в которых обозначены необходимые для соблюдения документы. Одним из таких документов являются Правила устройства электроустановок (ПУЭ). В них оговорены требования к монтажу электропроводки, освещения, порядку включения, выключения и аварийного обесточивания электросетей.

В случае загорания ПК или отдельных устройств, следует немедленно отключить электропитание, сообщить в службу пожарной охраны по телефону "11-01". Оценить ситуацию и, в случае небольшого очага возгорания, попытаться начать тушение загорания, используя средства первичного пожаротушения. Для тушения можно применять огнетушители с углекислотным или порошковым зарядом. Они предназначены для тушения электроустановок с напряжением до 1000 В. При обнаружении неисправностей, перегрева оборудования, появления запаха гари следует выключить ПК и другие включенные устройства, вынуть вилки электропитания из розеток и сообщить об этом обслуживающему персоналу, то есть лицу ответственному за пожарную безопасность помещений или ответственному за электрохозяйство.

Для предотвращения возгорания необходимо соблюдать основные меры предосторожности:

* не эксплуатировать поврежденные розетки или розетки, выполненные с нарушением норм пожарной безопасности;
* не допускать работу средств вычислительной техники с неисправными шнурами питания;
* соблюдать порядок уборки и хранения горючих материалов;
* не класть на устройства вычислительной техники посторонние предметы, бумагу, документацию;
* в помещениях не пользоваться открытым огнем и электронагревательными приборами с открытыми элементами нагрева или без терморегуляторов;
* запрещается работа на неисправном оборудовании (возникновение короткого замыкания и, как следствие, характерные искры и шум, запах плавящейся изоляции проводов);
* по окончании работы, перед закрытием помещений, следует обесточить все электропотребители.

## 5.2 Требования к рабочему месту инженера–программиста

При работе с компьютером человек подвергается воздействию ряда опасных и вредных производственных факторов:

* электромагнитных полей (диапазон высоких частот, сверх высоких частот и ультра высоких частот);
* инфракрасного и ионизирующего излучений;
* повышенный уровень шума и вибрации;
* повышенная температура внешней среды;
* отсутствие или недостаточная освещенность рабочей зоны;
* электрического тока и статического электричества;
* умственное перенапряжение;
* перенапряжение зрительных и слуховых анализаторов;
* монотонность труда;
* эмоциональные перегрузки;
* большие нагрузки на мышцы рук при работе с клавиатурой ЭВМ;
* большие нагрузки на мышцы шеи, спины и ног.

В соответствии с СанПиН 2.2.2/2.4.1340-03 рабочее место инженера-программиста должно обеспечивать возможность поворота корпуса в горизонтальной и вертикальной плоскости с фиксацией в заданном положении для обеспечения фронтального наблюдения экрана. Корпус ПК, клавиатура и другие блоки и устройства должны иметь матовую поверхность с коэффициентом отражения 0,4-0,6 и не иметь блестящих деталей, способных создавать блики. Также должно предусматриваться регулирование яркости и контрастности.

Концентрации вредных веществ, выделяемых персональными компьютерами в воздух помещений, не должны превышать предельно допустимых концентраций (ПДК), установленных для атмосферного воздуха.

Рабочие места с персональными компьютерами в помещениях с источниками вредных производственных факторов должны размещаться в изолированных кабинах с организованным воздухообменом.

Конструкция персонального компьютера должна обеспечивать возможность поворота корпуса в горизонтальной и вертикальной плоскости с фиксацией в заданном положении для обеспечения фронтального наблюдения экрана монитора. Дизайн персонального компьютера должен предусматривать окраску корпуса в спокойные мягкие тона с диффузным рассеиванием света. Корпус персонального компьютера, клавиатура и другие его блоки и устройства должны иметь матовую поверхность с коэффициентом отражения 0,4 - 0,6 и не иметь блестящих деталей, способных создавать блики.

Рабочие места с персональными компьютерами при выполнении творческой работы, требующей значительного умственного напряжения или высокой концентрации внимания, рекомендуется изолировать друг от друга перегородками высотой 1,5 - 2,0 м.

Большое значение имеет рациональная конструкция и расположение элементов рабочего места, что важно для поддержания оптимальной рабочей позы человека-оператора. При размещении рабочих мест с персональным компьютером расстояние между рабочими столами с видеомониторами (в направлении тыла поверхности одного видеомонитора и экрана другого видеомонитора), должно быть не менее 2,0 м, а расстояние между боковыми поверхностями видеомониторов – не менее 1,2 м.

Экран видеомонитора должен находиться от глаз пользователя на расстоянии 600-700 мм, но не ближе 500 мм с учетом размеров алфавитно-цифровых знаков и символов.

В процессе работы с компьютером необходимо соблюдать правильный режим труда и отдыха. В противном случае отмечаются значительное напряжение зрительного аппарата с появлением жалоб на неудовлетворенность работой, головные боли, раздражительность, нарушение сна, усталость и болезненные ощущения в глазах, в пояснице, в области шеи и руках.

Конструкция рабочего стола должна обеспечивать оптимальное размещение на рабочей поверхности используемого оборудования с учетом его количества и конструктивных особенностей, характера выполняемой работы. При этом допускается использование рабочих столов различных конструкций, отвечающих современным требованиям эргономики. Поверхность рабочего стола должна иметь коэффициент отражения 0,5-0,7.

Высота рабочей поверхности стола для взрослых пользователей должна регулироваться в пределах 680 - 800 мм, при отсутствии такой возможности высота рабочей поверхности стола должна составлять 725 мм.

Модульными размерами рабочей поверхности стола для ПЭВМ, на основании которых должны рассчитываться конструктивные размеры, следует считать: ширину 800, 1000, 1200 и 1400 мм, глубину 800 и 1000 мм при нерегулируемой его высоте, равной 725 мм.

Рабочий стол должен иметь пространство для ног высотой не менее 600 мм, шириной - не менее 500 мм, глубиной на уровне колен – не менее 450 мм и на уровне вытянутых ног - не менее 650 мм.

Конструкция рабочего стула (кресла) должна обеспечивать поддержание рациональной рабочей позы при работе на персональном компьютере, позволять изменять позу с целью снижения статического напряжения мышц шейно-плечевой области и спины для предупреждения развития утомления.

Рабочий стул (кресло) должен быть подъемно-поворотным, регулируемым по высоте и углам наклона сиденья и спинки, а также расстоянию спинки от переднего края сиденья, при этом регулировка каждого параметра должна быть независимой, легко осуществляемой и иметь надежную фиксацию.

Поверхность сиденья, спинки и других элементов стула (кресла) должна быть полумягкой, с нескользящим, слабо электризующимся и воздухопроницаемым покрытием, обеспечивающим легкую очистку от загрязнений.

Рабочее место пользователя персонального компьютера следует оборудовать подставкой для ног, имеющей ширину не менее 300 мм, глубину не менее 400 мм, регулировку по высоте в пределах до 150 мм и по углу наклона опорной поверхности подставки до 20°. Поверхность подставки должна быть рифленой и иметь по переднему краю бортик высотой 10 мм.

Клавиатуру следует располагать на поверхности стола на расстоянии 100 - 300 мм от края, обращенного к пользователю или на специальной, регулируемой по высоте рабочей поверхности, отделенной от основной столешницы.

Мощность экспозиционной дозы мягкого рентгеновского излучения в любой точке на расстоянии 0,05 м от экрана при любых положениях регулировочных устройств не должна превышать 1 мкЗв/час (100 мкР/час).

## 5.3 Расчет естественного освещения рабочего места

Расчет и нормирование естественного освещения производят по коэффициенту естественной освещенности "ε" (КЕО) в % по формуле 5.1:

, (5.1)



где

Ев – освещенность внутри помещения, лк;

Ен – одновременная освещенность наружной и горизонтальной плоскости рассеянным светом небосвода, лк.

На предприятиях наибольшее распространение получило естественное боковое освещение. При таком освещении основой расчета является требуемая площадь светового проема, определяемая по формуле 5.2:

(5.2)



где

So - площадь окон, м2;

Sп - площадь пола помещения, м2;

ε н - нормированное значение КЕО, %;

ho - световая характеристика окна (6.5-29);

Кз - коэффициент запаса;

Кзо - коэффициент, учитывающий затемнение окон противостоящими зданиями (1,0-1,7);

to - общий коэффициент светопропускания, определяемый из СанПиН 2.2.2/2.4.1340-03;

r1 - коэффициент, учитывающий повышение КЕО за счет отражения света от поверхности помещения (1,05-1,7).

Коэффициент "Кз" определяется равным 1,5. Учитываем, что длина пола помещения "l", равняется 12 м, а ширина "b" 8,4 м. Находим площадь пола по формуле 5.3:

, (5.3)



.



Нормированное значение КЕО определяется равным 1,2 %.

Значения остальных коэффициентов принимаются равными:

ho = 29;

r1 = 1,2;

Кзо = 1;

to = 0,3.

При расчете получено следующее значение требуемой площади светового проема по формуле 5.2:

.



Следовательно оконный проем должен быть не менее 15 м2.

## 5.4 Расчет искусственного освещения

Искусственное освещение применяют в темное и переходное время суток, а также при недостаточном или отсутствии естественного освещения. В помещении применяется общее равномерное искусственное освещение, расчет которого производится по методу светового потока. При расчете этим методом учитывается как прямой свет от светильника, так и свет, отраженный от потолка и стен.

Освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300 – 500 лк. Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 лк. Яркость бликов на экране персонального компьютера не должна превышать 40 кд/м2 и яркость потолка не должна превышать 200 кд/м2.

В качестве источников света при искусственном освещении следует применять преимущественно люминесцентные лампы типа ЛБ и компактные люминесцентные лампы (КЛЛ).

Для освещения помещений с персональными компьютерами следует применять светильники с зеркальными параболическими решетками, укомплектованными электронными пуско-регулирующими аппаратами (ЭПРА) [21]. Допускается использование многоламповых светильников с электромагнитными пуско-регулирующими аппаратами (ЭПРА), состоящими из равного числа опережающих и отстающих ветвей. Применение светильников без рассеивателей и экранирующих решеток не допускается.

Общее освещение при использовании люминесцентных светильников следует выполнять в виде сплошных или прерывистых линий светильников, расположенных сбоку от рабочих мест, параллельно линии зрения пользователя при рядном расположении видеодисплейных терминалов. При периметральном расположении компьютеров линии светильников должны располагаться локализовано над рабочим столом ближе к его переднему краю, обращенному к оператору.

Коэффициент пульсации не должен превышать 5%.

При отсутствии светильников с ЭПРА лампы многоламповых светильников или рядом расположенные светильники общего освещения следует включать на разные фазы трехфазной сети.

Помещения, где размещаются рабочие места с персональными компьютерами, освещается лампами типа ЛБ80, световой поток которых F = 5220 лм.

Освещенность определяется по следующей формуле 5.4:

, (5.4)



где

F - световой поток каждой из ламп, лм;

E - минимальная освещенность, лк;

k - коэффициент запаса, учитывающий запыление светильников и износ источников света;

Sп - площадь помещения, м2;

N - число источников света;

η - коэффициент использования светового потока;

z - коэффициент неравномерности освещения;

y - коэффициент затенения.

Определим данные для расчета. Коэффициент "k" для помещений освещаемых люминесцентными лампами, и при условии чистки светильников не реже двух раз в год берется равным:

k = 1,4-1,5.

При оптимальном расположении светильников коэффициент неравномерности равен:

z = 1,1-1,2.

Коэффициент затенения "y" вводится в расчет для помещений с фиксированным положением работающих, а также при наличии крупногабаритных предметов и принимается равным:

у = 0,8-0,9.

Коэффициент использования светового потока "η" зависит от типа светильника, коэффициента отражения светового потока от стен, потолка, пола, а также геометрических размеров помещения и высоты подвеса светильников, что учитывается одной комплексной характеристикой - индексом помещения. Показатель помещения определяется по формуле 5.5:

, (5.5)



где

h - высота подвеса светильников над рабочей поверхностью, м;

l - ширина помещения, м;

b - длина помещения, м.

Тогда индекс помещения по формуле (5.5) получается равным:



По найденному показателю помещения "i" и коэффициентам отражения потолка и стен, определяем коэффициент использования светового потока (под которым понимается отношение светового потока, падающего на рабочую поверхность, к световому потоку источника света). Для нашего случая "η" равняется 0,22.

Тогда освещенность по формуле (5.4) равна:

лк.



Расчет показывает, что освещенность в помещении, где размещаются рабочие места с персональными компьютеров удовлетворяет требованиям, так как нормальная минимальная освещенность должна составлять 400лк.

Заключение

Данный проект был направлен на автоматизацию проектных работ по программированию станка с ЧПУ Walter CIP6. Были проведены исследовательские работы по изучению состояния дел в этой сфере, изучены предложения других компаний, был создан программный продукт, посчитаны экономические затраты на его создание и сформулированы требования по безопасности жизнедеятельности.

В процессе проектирования был получен программный модуль, отвечающий всем заявленным требованиям. Этот модуль может проводить сложные математические расчеты неявным образом, отображать в графическом представлении введенные исходные данные. В результате использования модуля можно получить готовую управляющую программу для системы ЧПУ.

Для упрощения задачи создания модуля, в нем применен принцип внешнего управления работой пакета 3D моделирования. За счет этого была реализована возможность анимированной имитации обработки на станке, с получением твердотельной математической модели изделия, очень близкой к реальному изделию, которое могло бы получиться на станке, в результате использования этой управляющей программы.

Особое значение имеет тот факт, что происходит уход от традиционных бумажных носителей на всех этапах проектирования и внедрения управляющей программы. Достигается это за счет того, что такое же программное обеспечение может быть установлено на рабочем месте станочника, и используя его, рабочий может получить всю необходимую информацию о наладке станка в более удобном виде, чем это может быть представлено на бумажной карте наладки.

Данный программный модуль предназначен для применения в группе шлифовальных станков с ЧПУ Walter, оснащенными 4 – 6 осями. С помощью этого модуля могут быть смоделированы большинство традиционных осевых инструментов, которые применяются в производстве, а также может быть осуществлена переточка затупившегося инструмента.

Возможность имитации обработки позволяет отказаться от применения пробной детали, которая предназначена для отладки программы обработки и чаще всего оказывается испорченной.

Также был произведен анализ условий труда программиста, который должен работать в этом приложении. Были описаны параметры рабочего места и рассчитаны нормы освещенности рабочего помещения. В частности были рассчитаны такие параметры как площадь оконного проема и количество ламп искусственного освещения, необходимых для освещения данной комнаты при работе с персональным компьютером.

Был произведен экономический расчет стоимости разработки данного программного модуля. В нем были учтены стоимость компьютера и периферийного оборудования, применяемых в ходе разработки, а так же стоимость отладки и внедрения программного модуля. Анализ аналогов других фирм показал, что данный программный модуль может выполнять все те же действия, что и аналоги, но его стоимость на 239828,86 рублей ниже чем у аналогов.

Список литературы

1. Автоматизация проектно-конструкторских работ и технологической подготовки производства в машиностроении. Т. 1/Под ред. О. И. Семенкова.- Минск: Высшая школа, 2005.
2. Волков Д.И., Скляренко В.К.. Экономика предприятия. Курс лекций.- М.: Инфра-м, 2004.
3. Гольдштейн А.И., Молочник В.И. О внутренней структуре постпроцессоров. — В кн.: Повышение эффективности использования станков с ЧПУ. - Киев: Знание, 2006.
4. ГОСТ 886-77 "Свёрла спиральные с цилиндрическим хвостовиком. Длинная серия".
5. ГОСТ 2092-77 "Свёрла спиральные удлиненные с коническим хвостовиком".
6. ГОСТ 10079-71 "Развертки конические с коническим хвостовиком".
7. ГОСТ 14952-75 "Свёрла центровочные комбинированные".
8. ГОСТ 18121-72 "Развёртки котельные машинные".
9. ГОСТ 19267-73 "Развертки машинные цилиндрические с цилиндрическим хвостовиком для легких сплавов".
10. ГОСТ 21579-76 "Зенкеры с цилиндрическим хвостовиком для легких сплавов".
11. Делфи 4. Библия разработчика. Том Сван. Киев/Москва/Санкт-Петербург: Диалектика, 1998.
12. Ендовицкий Д.А., Коменденко С.Н. Организация анализа и контроля инновационной деятельности хозяйствующего субъекта. Под ред. Л.Т. Гимеровской. М.: Финансы и статистика, 2004.
13. Знакомство с Delphi 7. Архангельский А.Я. Москва: издательство Бином, 2004.
14. Инструменты программирования в Delphi 7. Фаронов В.В. СПб.: Питер, 2006.
15. Инструкция по программированию. Издание 03.96. SINUMERIK 840D/810D/FM-NC.
16. Ильенкова С.Д., Гохберг Л.М., Ягудин С.Ю. и др.Инновационный менеджмент. Учебник для вузов./ под ред. С.Д. Ильенковой. М.: Банки и биржи, Юнити, 1997.
17. Механика промышленных роботов: Учеб. Пособие для вузов: В 3 кн. / Под ред. К. В. Фролова, Е. И. Воробьева. Кн.2: расчет и проектирование механизмов / Е. И. Воробьев, О. Д. Егоров, С. А. Попов. – М: Высш. шк., 1988.
18. Инвестиции. Учебное пособие/Г.П. Подшиваленко, Н.И. Лахметкина, М.В. Макарова и др. М.:Кнорус,2004.
19. Оборудование машиностроительных предприятий / Схиртладзе А. Г Выходец., В. И., Никифоров Н. И., Отений Я. Н. − ВолгГТУ, Волгоград, 2005.
20. СанПиН 2.2.2/2.4.1340-03 "Гигиенические требования к персональным электронно-вычислительным машинам и организации работы".
21. Системная интеграция/Машиностроение СТА 3/2002 Николай Панышев, Дмитрий Ялымов "Система числового программного управления технологическим оборудованием".
22. Интернет источник. CALS-технологии. www.calscenter.com.
23. Интернет источник. Всемирная интернет энциклопедия. www.wikipedia.ru.
24. Интернет источник. Информационные системы. http://revolution./programming.
25. Интернет источник. Уфимский государственный авиационный технический университет. www.twirpx.com.
26. Интернет источник. Siemens Sinumerik 840D. www.chipmaker.ru.
27. Интернет источник. Филиппович К.В. "Идеология постпроцессирования в современных CAD/CAM-системах" Россия, ООО Евразия Лимитед, 2000. www.sapr2000.ru

Приложение А

Исходный текст программного модуля:

program Project2; //Имя программы

uses //Объявление прикрепленных к проекту файлов

Forms,

Unit1 in 'Unit1.pas' {Form1},

Unit2 in 'Unit2.pas' {Form2},

Unit3 in 'Unit3.pas' {Form3},

Unit4 in 'Unit4.pas' {Form4},

Unit5 in 'Unit5.pas' {Form5},

Unit6 in 'Unit6.pas',

Unit7 in 'Unit7.pas' {Form7},

Unit8 in 'Unit8.pas' {Form8},

Unit9 in 'Unit9.pas' {Form9},

Unit10 in 'Unit10.pas',

Unit11 in 'Unit11.pas' {Form11},

Unit12 in 'Unit12.pas' {Form12},

Unit13 in 'Unit13.pas' {Form13},

Unit14 in 'Unit14.pas' {Form14},

Unit15 in 'Unit15.pas' {Form15},

Unit16 in 'Unit16.pas' {Form16},

Unit17 in 'Unit17.pas' {Form17};

var //объявление переменных

s:string; //назначение и присвоение типа переменных

ff:textfile;

{$R \*.res} // директива компилятору подключить файл ресурсов

procedure Init\_All; //объявление процедуры, относящейся ко всему проекту

begin

N\_Ses:=1;

Init\_Form2;

Form2.Caption:='Операция N 1';

Init\_St\_Det\_Kr;

AssignFile(ff,path+'DAT\Stanok.dat'); //все параметры в проекта сохраняются в файл

reset(ff);

readln(ff,Xst[0]);

readln(ff,Yst[0]);

readln(ff,Zst[0]);

readln(ff,Ast[0]);

readln(ff,Bst[0]);

readln(ff,Cst[0]);

CloseFile(ff);

Init\_Zag; //инициализация пользовательских функций

Init\_Poly;

Init\_Chk\_Box;

Init\_Traect(1);

Traect(1);

//Profile;

Init\_Moving;

//Moving\_9484;

Load\_Op;

N\_St:=0; N\_Fin:=N\_Pos[N\_Act]; N\_Last:=0;

Init\_Data3('DAT\9484.txt'); //чтение данных из файла

Form3.Caption:='Начапьные установки';

Fill\_Str\_Grid1\_3; //вставка стандартный элементов Windows

Init\_Data4;

Fill\_Str\_Grid1\_4;

Init\_BitMap;

Draw\_All;

Init\_Data5;

Init\_Data12;

Init\_BitMap12;

Fill\_Str\_Grid3\_12(1);

Fill\_Str\_Grid1\_5;

Init\_Data7;

Fill\_Str\_Grid1\_7;

Step:=1;

str(Step:1:3,s);

Form8.Memo1.Text:=s;

Form8.Button17.Caption:='';

Nst\_Act:=0;

Init\_Zag;

Init\_Data8;

Fill\_Str\_Grid1\_8;

Fill\_Str\_Grid2\_8;

Init\_BitMap8;

Nkr\_Edit:=1;

Init\_BitMap9;

Cooling;

Fill\_Str\_Grid1\_13;

N\_Buff:=1;

Put\_Buff(N\_Buff);

Form3.Visible:=true;

Write\_Zag;

Init\_Data14;

Init\_Data\_15;

Init\_Data\_16;

end;

begin

Application.Initialize;

Application.CreateForm(TForm1, Form1); //создание вспомогательных окон

Application.CreateForm(TForm2, Form2);

Application.CreateForm(TForm3, Form3);

Application.CreateForm(TForm4, Form4);

Application.CreateForm(TForm5, Form5);

Application.CreateForm(TForm7, Form7);

Application.CreateForm(TForm8, Form8);

Application.CreateForm(TForm9, Form9);

Application.CreateForm(TForm11, Form11);

Application.CreateForm(TForm12, Form12);

Application.CreateForm(TForm13, Form13);

Application.CreateForm(TForm14, Form14);

Application.CreateForm(TForm15, Form15);

Application.CreateForm(TForm16, Form16);

Application.CreateForm(TForm17, Form17);

Init\_All;

Application.Run;

end.

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls, Buttons, CheckLst, Grids,Math;

type

TForm1 = class(TForm)

Button1: TButton;

Memo1: TMemo;

Button8: TButton;

CheckListBox1: TCheckListBox;

Button2: TButton;

Button3: TButton;

StringGrid1: TStringGrid;

Button4: TButton;

Button5: TButton;

CheckListBox2: TCheckListBox;

Button6: TButton;

Button7: TButton;

Button9: TButton;

CheckListBox3: TCheckListBox;

Button10: TButton;

Button11: TButton;

Button12: TButton;

Button13: TButton;

Button14: TButton;

Button15: TButton;

Button16: TButton;

procedure Button3Click(Sender: TObject); //функции нажатия кнопок

procedure Button2Click(Sender: TObject);

procedure Button8Click(Sender: TObject);

procedure CheckListBox1ClickCheck(Sender: TObject);

procedure StringGrid1KeyPress(Sender: TObject; var Key: Char);

procedure Button4Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

procedure Button6Click(Sender: TObject);

procedure CheckListBox2ClickCheck(Sender: TObject);

procedure Button7Click(Sender: TObject);

procedure Button9Click(Sender: TObject);

procedure CheckListBox3ClickCheck(Sender: TObject);

procedure Button10Click(Sender: TObject);

procedure Button12Click(Sender: TObject);

procedure Button13Click(Sender: TObject);

procedure Button14Click(Sender: TObject);

procedure CheckListBox1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Button15Click(Sender: TObject);

procedure Button16Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

private

{ Private declarations }

public

{ Public declarations }

end;

type

XYZ=record

x:real;

y:real;

Z:real;

end;

var

Form1: TForm1;

XYZ\_D:XYZ;

X,Y,Z,A,B,C:real;

//start

Type\_Tool:integer;

SM\_X,SM\_Y,SM\_Z,POV\_A,POV\_B,SM\_C:array[0..500] of real; //объявление массивов

Dkr,Rkr,Skr,Akr:array [1..11] of real; //для хранения вводимых параметров в сетке

Xkr,Ykr,Zkr,X1,Y1,Z1,dX,dY,dZ,XB1,YB1,ZB1,XB,YB,ZB:real;

{Ddet,}Rdet,Zst\_B,Obr\_Kon,FPL,Alf1,Alf2,Apodt,Dpodt,Wdet:real;

U\_X,U\_Y,U\_Z1,U\_Z2,UB\_X,dRdet:array [1..11] of real;

N\_Zub:integer;

Rvp,Lz,Lo,Lzt,Arp,Rp:real;

Zp\_Kr:array [1..6,1..2] of real;

Np\_Kr,Sb\_Int:array [1..11] of integer;

FiC,Ftr,Teta,Ltr\_P,Ltr\_R,LC,X0,Y0,Z0:array[0..11] of real;

XC,YC,ZC,XT,YT,ZT,XD,YD,ZD,dA,dB,dC,V\_D:real;

Xst,Yst,Zst,Ast,Bst,Cst:array [0..500] of real;

Xst\_Old,Yst\_Old,Zst\_Old,Ast\_Old,Bst\_Old,Cst\_Old:real;

N\_Act,N\_St,N\_Fin,N\_Last,Obr:integer;

FTime:real;

F,Feed:integer;

Nt,N\_Pos:array [0..500] of integer; //заполнение массивов

Com:array [0..500] of string[20];

ff1:textfile;

FName,ABS\_OTN:string;

Xtr,Ytr,Ztr,X0det,Y0det,Z0det:array [0..10] of real;

dXC1,dYC1,dZC1,dX\_PR,dY\_PR,dZ\_PR,Atr,Btr:real;

Rpr:array[1..10,1..2] of real;

Num\_Op,Nex,N\_C,N\_Op,Nop:integer;

OutMet:array [0..500] of integer;

dFiex,dYex, dZex:array [0..100] of real;

Rex,A0ex,A1ex,Yex,dRex,Hstr,Akan:real;

{D\_Auto\_Calc,}Ds,Hrk,Xrk:array [1..10] of real;

Gp,Gs:array [1..2,1..10] of real;

//N\_Cycl:array [1..10,1..2] of integer;

//Amin,Amax,AStep:array [1..7] of integer;

Num\_Check:integer;

Zel,Yel:real;

ddZ,ddY:real;

N\_Ses:integer;

Beta,X\_P1,Y\_P1,X\_P2,Y\_P2,Ld1,Ld2:real;

XXtr,YYtr,ZZtr:real;

rad:real;

N\_Kr:integer;

Nkr,N\_Razb,N\_Sech,N\_Sech\_X:array [1..11] of integer;

Z0\_Kr,Ust\_D,Ust\_VD:real;

path:string;

W,Hv,Fit\_Init:array [1..11] of real;

Xv,Yv,Zv,Xv1,Yv1,Zv1,XvT,YvT,ZvT,Y\_X,Z\_X:array [0..50,0..2,0..5,0..100] of real;

N\_Tr:integer;

Cut\_Kr:array [0..20] of integer;

N\_Cut,Vozvr:array [0..500] of integer;

Xkr4,Ykr4:array [1..4,0..4,0..100] of real;

Name\_Op:array [1..20] of string;

Viz\_Vt,Viz\_Dim,Auto\_Calc:array [1..20] of integer;

A\_Hrk:real;

dy\_Ex,dz\_Ex:array [0..100] of real;

Comp\_Y,Comp\_Z:array [0..100] of real;

x\_Out,y\_Out,z\_Out:array [0..100] of real;

Move\_Out:integer;

dfex:array [1..11] of real;

Cool:integer;

dotv\_Cool,D\_Cool,Fi\_Cool,Sm\_Kan,Fos\_Kan,Frad\_Kan,A\_Cool:real;

Opt\_Surf:array [0..3] of string;

N\_Opt:array [1..11] of integer;

NC\_Act:integer;

X\_NC,Y\_NC,Z\_NC,A\_NC,B\_NC,C\_NC,DX\_NC,DY\_NC,DZ\_NC,DA\_NC:array [1..10] of real;

N\_Dop\_Act:integer;

Adov:array [1..11] of real;

N\_Feed:integer;

Napr:array [0..11] of integer;

S\_Napr:array[0..11] of string;

Name\_Napr:array [1..3] of string;

X\_Arc,Y\_Arc,Z\_Arc: real;

XC1\_0,YC1\_0,ZC1\_0,XC2\_0,YC2\_0,ZC2\_0,XP\_0,YP\_0,ZP\_0,XR\_0,YR\_0,ZR\_0:real;

Nz\_Act:array [1..20,1..20] of integer;

Nz\_Fi:array [0..20] of real;

X\_Dop,Y\_Dop,Z\_Dop,A\_Dop,B\_Dop,C\_Dop:array [1..10] of real;

FX\_Dop,FY\_Dop,FZ\_Dop,FA\_Dop,FB\_Dop,FC\_Dop:array [1..10] of real;

X\_Pr,Y\_Pr,Z\_Pr,A\_Pr,B\_Pr,C\_Pr:array [1..10,1..3] of real;

FX\_Pr,FY\_Pr,FZ\_Pr,FA\_Pr,FB\_Pr,FC\_Pr:array [1..10,1..3] of real;

X\_Otv,Y\_Otv,Z\_Otv,A\_Otv,B\_Otv,C\_Otv:array [1..10] of real;

FX\_Otv,FY\_Otv,FZ\_Otv,FA\_Otv,FB\_Otv,FC\_Otv:array [1..10] of real;

X\_Cst,Y\_Cst,Z\_Cst,A\_Cst,B\_Cst,C\_Cst:array [1..10] of real;

X\_Cfin,Y\_Cfin,Z\_Cfin,A\_Cfin,B\_Cfin,C\_Cfin:array [1..10] of real;

X\_Cex,Y\_Cex,Z\_Cex,A\_Cex,B\_Cex,C\_Cex:array [1..10] of real;

FX\_C,FY\_C,FZ\_C,FA\_C,FB\_C,FC\_C:array [1..10] of real;

Num\_C:array [1..10] of integer;

X\_Vix,Y\_Vix,Z\_Vix,A\_Vix,B\_Vix,C\_Vix:array [1..10] of real;

FX\_Vix,FY\_Vix,FZ\_Vix,FA\_Vix,FB\_Vix,FC\_Vix:array [1..10] of real;

FX\_Kor,FY\_Kor,FZ\_Kor,FA\_Kor,FB\_Kor,FC\_Kor:real;

FX\_Ust,FY\_Ust,FZ\_Ust,FA\_Ust,FB\_Ust,FC\_Ust:real;

PosTime:array [1..10] of real;

Num\_Vix:array [1..10] of integer;

AutoCalc:boolean;

AHrk,Av,Ak,Ap,Sp:array [1..2,1..10] of real;

Avt,AHrkt:array [1..10] of real;

Xpx,Ypx:array [1..3,0..50] of real;

N\_Proection,N\_Profile,XSect:integer;

CrsAlfa:array [1..2] of real;

RminX,RminY:array [1..10] of real;

TwoFi:real;

FiRmin:array [1..2,1..10] of real;

CrsFiBeg,CrsFiEnd:array [1..10,1..2] of real;

CrsXbeg,CrsYbeg,CrsXend,CrsYend,CrsZend:array [1..10,1..2] of real;

x\_Pol,y\_Pol,z\_Pol,R\_Pol,Fit\_Pol:array [1..10,1..2,0..300] of real;

N\_Pol:integer;

NBeg,NEnd:array [1..10,1..2] of integer;

AsectK,RminS:array [1..10] of real;

NsectK,N\_PolK:array [1..10] of integer;

N\_Auto,Auto\_Tang:array[1..10] of integer;

Lzat1,Lzat2,Rzat,Hzat:array [1..10] of real;

ExtdY:array [1..50,1..10] of real;

Xzat\_c,Yzat\_c:array [1..10,1..50] of real;

CrossSect,Ntr,Ntr1:integer;

Zcont,Ycont:array [0..50,1..2,0..3] of real;

LpS,LpR:real;

N\_ZP\_PL:integer;

HkZP,HfZP,SfZP,RfZP,FiZp:array [1..10] of real;

SLent,SnLent,ALent,APovL,BPovL,XPovL,YPovL,ZPovL:array [1..10] of real;

N\_Sect:boolean;

Edit\_Op:integer;

N\_OpGr,N\_Link:array [1..10] of integer;

X\_ZP\_PL,Y\_ZP\_PL,dY\_ZP\_PL,A\_ZP\_PL:array [1..10] of real;

Z\_ZP\_PL:array [1..2,1..10] of real;

Nop\_ZP\_PL:integer;

GlDv:array [1..6,1..10] of integer;

fiAmax:real;

procedure Init\_Chk\_Box; //описание процедур построцессора

procedure Init\_St\_Det\_Kr;

procedure Init\_Moving;

procedure Zatilok;

Procedure Init\_Traect(i:integer);

procedure Traect(N\_Act:integer);

procedure Memo\_Out;

procedure W\_Str(com:string);

function w\_r(val:real;f1,f2:integer):string;

procedure G90;

procedure G91;

procedure G01;

procedure Profile;

procedure Load\_Op\_All;

procedure Load\_Op;

procedure Fix\_Det(V\_D,UB\_X,Ddet:real);

procedure Make\_Krug(N\_kr,m:integer);

procedure UpDate\_ChkBox2;

procedure Make\_Spind;

procedure Init\_All;

implementation

uses Unit2,Unit3,Unit4,Unit5,Unit6,Unit10,Unit14, Unit8,Unit13,Unit12,

Unit7,Unit9,Unit15,Unit16,Unit17;

{$R \*.dfm} //процедура реализации

var

Str\_Rel:array [0..500] of string;

ff:text;

procedure Init\_All;

var

s:string;

begin

path:=GetCurrentDir+'\';

N\_Ses:=1;

Init\_Form2;

Form2.Caption:='Операция N 1';

Init\_St\_Det\_Kr;

Init\_Zag;

Init\_Poly;

Init\_Chk\_Box;

Init\_Traect(1);

Traect(1);

//Profile;

Init\_Moving;

//Moving\_9484;

Load\_Op;

N\_St:=0; N\_Fin:=N\_Pos[N\_Act]; N\_Last:=0;

//Init\_Data3('DAT\9484.txt');

Form3.Caption:='Начальные установки';

Fill\_Str\_Grid1\_3;

Init\_Data4;

Fill\_Str\_Grid1\_4;

Init\_BitMap;

Draw\_All;

Init\_Data5;

Init\_Data12;

Init\_BitMap12;

Fill\_Str\_Grid3\_12(1);

Fill\_Str\_Grid1\_5;

Init\_Data7;

Fill\_Str\_Grid1\_7;

Step:=1;

//str(Step:1:3,s);

Form8.Memo1.Text:=s;

Form8.Button17.Caption:='';

Nst\_Act:=0;

Init\_Zag;

Init\_Data8;

Fill\_Str\_Grid1\_8;

Fill\_Str\_Grid2\_8;

Init\_BitMap8;

//Nkr\_Edit:=1;

Init\_BitMap9;

Cooling;

Fill\_Str\_Grid1\_13;

N\_Buff:=1;

Put\_Buff(N\_Buff);

Form3.Visible:=true;

Write\_Zag;

Init\_Data14;

Init\_Data\_15;

Init\_Data\_16;

end;

procedure W\_Str(com:string);

begin

Form1.Memo1.Lines.Add(com);

end;

function w\_r(val:real;f1,f2:integer):string;

var

s:string;

begin

str(val:f1:f2,s);

w\_r:=s;

end;

function w\_i(val:integer;f1:integer):string;

var

s:string;

begin

str(val:f1,s);

w\_i:=s;

end;

procedure Memo\_Out;

var

s:string;

i,j:integer;

TotTime:real;

begin

Form1.Memo1.Clear;

Form1.Memo1.Visible:=false;

str(N\_Pos[N\_Act]:2,s);

W\_Str('Кол-во позиций:0-'+s);

for i:=0 to N\_Pos[N\_Act] do

begin

W\_Str('');

if i=0 then s:='';

if i>0 then s:='G01 '+Str\_Rel[i];

W\_Str('Поз.'+w\_i(i,2)+':'+s);

W\_Str(' Абс Отн');

W\_Str('X='+w\_r(Xst[i],9,3)+' '+w\_r(SM\_X[i],9,3));

W\_Str('Y='+w\_r(Yst[i],9,3)+' '+w\_r(SM\_Y[i],9,3));

W\_Str('Z='+w\_r(Zst[i],9,3)+' '+w\_r(SM\_Z[i],9,3));

W\_Str('A='+w\_r(Ast[i],9,3)+' '+w\_r(POV\_A[i],9,3));

W\_Str('B='+w\_r(Bst[i],9,3)+' '+w\_r(POV\_B[i],9,3));

W\_Str('C='+w\_r(Cst[i],9,3)+' '+w\_r(SM\_C[i],9,3));

end;

W\_Str('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

str(N\_Act:1,s);

W\_Str('Время оп.'+s+' :'+w\_i(trunc(PosTime[N\_Act]),1)+''''+

w\_r(frac(PosTime[N\_Act])\*60,2,0)+'"');

TotTime:=0;

for j:=1 to N\_op do

TotTime:=TotTime+PosTime[j];

W\_Str('Время общее:'+w\_r(TotTime,2,3));

Form1.Memo1.Visible:=true;

end;

procedure Move(Os:char;len:real;part\_name:string); //создание структуры кадра

begin

writeln(ff1,'(position\_pa :part\_assembly :start ','"',part\_name,'" ',

':select\_done :translate :dir\_len :',Os,' :len ',len:8:6,' :done ',

':pos\_dynamic :dynamo-done)');

writeln(ff1,'(redraw\_vp "vport1")');

end;

procedure Rotate(Os:char;angle:real;part\_name:string);

begin

writeln(ff1,'(position\_pa :part\_assembly :start ','"',part\_name,'" ',

':select\_done :rotate :axis :',Os,' :rotation\_angle ',angle:8:6,' :done ',

':pos\_dynamic :dynamo-done)');

writeln(ff1,'(redraw\_vp "vport1")');

end;

procedure XX(len:real);

begin

Move('x',len,'/CIP6/N\_S');

end;

procedure YY(len:real);

begin

Move('y',len,'/CIP6/Krug');

end;

procedure ZZ(len:real);

begin

Move('z',len,'/CIP6/Krug');

end;

procedure AA(ang:real);

begin

writeln(ff1,'(current\_wp "/CIP6/N\_S/V\_S/C\_Sys/Det/w\_A")');

Rotate('w',ang,'/CIP6/N\_S/V\_S/C\_Sys/Det');

end;

procedure BB(ang:real);

begin

writeln(ff1,'(current\_wp "/CIP6/N\_S/V\_S/w\_B")');

Rotate('w',ang,'/CIP6/N\_S/V\_S');

end;

procedure CC(len:real);

begin

writeln(ff1,'(current\_wp "/CIP6/N\_S/V\_S/w\_C")');

Move('w',len,'/CIP6/N\_S/V\_S/C\_Sys');

end;

procedure AddToFile(Fn:string;N\_Pos:integer);

begin

AssignFile(ff1,Fn); //запись файла связи с 3D пакетом

Append(ff1);

Nt[N\_Pos]:=F;

writeln(ff1,'(dotimes ( n ',Nt[N\_Pos]:2,')');

if abs(SM\_X[N\_Pos])>=0.001 then

begin

XX(SM\_X[N\_Pos]/Nt[N\_Pos]);

Xst[N\_Pos]:=Xst[N\_Pos-1]+SM\_X[N\_Pos];

end;

if abs(SM\_Y[N\_Pos])>=0.001 then

begin

YY(SM\_Y[N\_Pos]/Nt[N\_Pos]);

Yst[N\_Pos]:=Yst[N\_Pos-1]+SM\_Y[N\_Pos];

end;

if abs(SM\_Z[N\_Pos])>=0.001 then

begin

ZZ(-SM\_Z[N\_Pos]/Nt[N\_Pos]);

Zst[N\_Pos]:=Zst[N\_Pos-1]+SM\_Z[N\_Pos];

end;

if abs(POV\_B[N\_Pos])>=0.001 then

begin

BB(POV\_B[N\_Pos]/Nt[N\_Pos]);

Bst[N\_Pos]:=Bst[N\_Pos-1]+POV\_B[N\_Pos];

end;

if abs(Pov\_A[N\_Pos])>=0.001 then

begin

AA(POV\_A[N\_Pos]/Nt[N\_Pos]);

Ast[N\_Pos]:=Ast[N\_Pos-1]+POV\_A[N\_Pos];

end;

if abs(SM\_C[N\_Pos])>=0.001 then

begin

CC(SM\_C[N\_Pos]/Nt[N\_Pos]);

Cst[N\_Pos]:=Cst[N\_Pos-1]+SM\_C[N\_Pos];

end;

writeln(ff1,')');

Flush(ff1);

CloseFile(ff1);

end;

procedure G90;

begin

ABS\_OTN:='G90';

X:=Xst[N\_Pos[N\_Act]];

Y:=Yst[N\_Pos[N\_Act]];

Z:=Zst[N\_Pos[N\_Act]];

A:=Ast[N\_Pos[N\_Act]];

B:=Bst[N\_Pos[N\_Act]];

C:=Cst[N\_Pos[N\_Act]];

end;

procedure G91;

begin

ABS\_OTN:='G91';

X:=0; Y:=0; Z:=0;

A:=0; B:=0; C:=0;

end;

procedure G01;

var

ss,ss1:string;

begin

inc(N\_Pos[N\_Act]);

PosTime[N\_Act]:=PosTime[N\_Act]+abs(FTime);

FTime:=0;

OutMet[N\_Pos[N\_Act]]:=Obr;

Obr:=0;

Str\_Rel[N\_Pos[N\_Act]]:=ABS\_OTN;

if pos('G91',ABS\_OTN)>0 then

begin

SM\_X[N\_Pos[N\_Act]]:=X;

SM\_Y[N\_Pos[N\_Act]]:=Y;

SM\_Z[N\_Pos[N\_Act]]:=Z;

POV\_B[N\_Pos[N\_Act]]:=B;

POV\_A[N\_Pos[N\_Act]]:=A;

SM\_C[N\_Pos[N\_Act]]:=C;

end;

if pos('G90',ABS\_OTN)>0 then

begin

SM\_X[N\_Pos[N\_Act]]:=X-Xst[N\_Pos[N\_Act]-1];

SM\_Y[N\_Pos[N\_Act]]:=Y-Yst[N\_Pos[N\_Act]-1];

SM\_Z[N\_Pos[N\_Act]]:=Z-Zst[N\_Pos[N\_Act]-1];

POV\_B[N\_Pos[N\_Act]]:=B-Bst[N\_Pos[N\_Act]-1];

POV\_A[N\_Pos[N\_Act]]:=A-Ast[N\_Pos[N\_Act]-1];

SM\_C[N\_Pos[N\_Act]]:=C-Cst[N\_Pos[N\_Act]-1];

end;

Xst[N\_Pos[N\_Act]]:=Xst[N\_Pos[N\_Act]-1];

Yst[N\_Pos[N\_Act]]:=Yst[N\_Pos[N\_Act]-1];

Zst[N\_Pos[N\_Act]]:=Zst[N\_Pos[N\_Act]-1];

Ast[N\_Pos[N\_Act]]:=Ast[N\_Pos[N\_Act]-1];

Bst[N\_Pos[N\_Act]]:=Bst[N\_Pos[N\_Act]-1];

Cst[N\_Pos[N\_Act]]:=Cst[N\_Pos[N\_Act]-1];

str(N\_Pos[N\_Act]:1,ss);

if NC\_ACt=0 then

begin

str(N\_Pos[N\_Act]:1,ss1);

Form1.CheckListBox1.Items.Add(ss+':'+Com[N\_Pos[N\_Act]]);

if N\_Pos[N\_Act]>1 then

Form1.CheckListBox1.State[N\_Pos[N\_Act]-1]:=cbUnChecked;

Form1.CheckListBox1.State[N\_Pos[N\_Act]]:=cbChecked;

AddToFile(Path+'LSP\Hod.lsp',N\_Pos[N\_Act]);

end;

end;

procedure Profile;

var

RF:real;

ff:textfile;

X3,Y3,Z3,X31,Y31,Z31,X32,Y32,Z32,Mu\_YZ,Mu\_YZ1,Mu\_YX,Mu\_XZ:real;

Xin,Yin,Zin,Xex,Yex,Zex,Muin,Muex:real;

Xin1,Yin1,Zin1,Xex1,Yex1,Zex1,Muin1,Muex1:real;

Ang,Rs,Rd,Ksi,Ksi1,Xc,Yc,Zc1,Xc1,Yc1,F0,F1,Ficx:real;

N\_Uch,N\_Point,Np1,i,j,N\_Half:integer;

dTeta,R\_Limit1,R\_Limit2,xL1,yL1,zL1,xL2,yL2,zL2,xL3,yL3,zL3,

fex,Alf1,Alf2,Alf3,L1,L2,L3,Step:real;

xL1f,yL1f,zL1f,xL2f,yL2f,zL2f,xL3f,yL3f,zL3f:real;

N\_Break:array [1..2,0..4,0..100] of integer;

N\_SpL:integer;

//LtrR,LtrP:extended;

fiA{,fiAmax},Vfi,fit\_max,dX,dY,dZ:real;

Xv03,Yv03,Zv03:real;

K\_Razb:real;

Fex\_Min,Fex\_Max:array [1..3] of real;

fit0,{fit\_st,fit\_fin,}dS\_Min:real;

Empt:array [1..12] of integer;

a1\_p,an\_p,Sn\_p,r\_p,Ltr\_Pfin:real;

NSect:integer;

procedure Sil(N\_Tr:integer);

var

i:integer;

ff1:textfile;

fixX,fixY,fixZ:real;

fixX1,fixY1,fixZ1:real;

fixX2,fixY2,fixZ2:real;

procedure Vint(N\_Tr,N\_Half,N\_Uch,j:integer);

begin

XvT[N\_Tr,N\_Half,N\_Uch,j]:=Xv[N\_Tr,N\_Half,N\_Uch,j];

YvT[N\_Tr,N\_Half,N\_Uch,j]:=Yv[N\_Tr,N\_Half,N\_Uch,j]\*cos({FiA}0)+

Zv[N\_Tr,N\_Half,N\_Uch,j]\*sin({FiA}0);

ZvT[N\_Tr,N\_Half,N\_Uch,j]:=-Yv[N\_Tr,N\_Half,N\_Uch,j]\*sin({FiA}0)+

Zv[N\_Tr,N\_Half,N\_Uch,j]\*cos({FiA}0);

fixX:=XvT[N\_Tr,N\_Half,N\_Uch,j];

fixY:=YvT[N\_Tr,N\_Half,N\_Uch,j];

fixZ:=ZvT[N\_Tr,N\_Half,N\_Uch,j];

writeln(ff1,XvT[N\_Tr,N\_Half,N\_Uch,j]:8:5,',',

YvT[N\_Tr,N\_Half,N\_Uch,j]:1:5,',',

ZvT[N\_Tr,N\_Half,N\_Uch,j]:1:5);

end;

procedure Create\_Bspline(N\_Tr,N\_Sp,N\_Uch,N\_Half:integer);

var

j:integer;

x3,y3,z3:real;

begin

inc(N\_Spl);

case N\_Sp of

begin

writeln(ff1,'(create\_curve\_on\_surface :wire\_part "/Sil\_',N\_Tr:1,'.',N\_Spl:1,'" ',

':start :selected\_part "/a1/Krug" :face\_3d :by\_feature ',

':full\_name :start\_name "/a1/Krug" :feature "feat1" ',

':end\_name :select\_done ');

for j:=0 to N\_Razb[N\_Act] do

if (N\_Break[N\_Half,N\_Uch,j]=0) then

begin

Empt[N\_Sp]:=1;

Vint(N\_Tr,N\_Half,N\_Uch,j);

if (N\_Spl=4) and (j=N\_Razb[N\_Act]) then

Vint(N\_Tr,1,1,j);

if (N\_Spl=6) and (j=N\_Razb[N\_Act]) then

Vint(N\_Tr,1,3,j);

end;

writeln(ff1,')');

writeln(ff1,'(remove\_from\_vp\_drawlist "vport1" :with-wp ',

' "/Sil\_',N\_Tr:1,'.',N\_Spl:1,'")');

end;

begin

{ Vint(N\_Tr,1,1,N\_Razb[N\_Act]);

fixX1:=fixX; fixY1:=fixY; fixZ1:=fixZ;

Vint(N\_Tr,1,3,N\_Razb[N\_Act]);

fixX2:=fixX; fixY2:=fixY; fixZ2:=fixZ;

x3:=(fixX1+fixX2)/2;

z3:=(fixZ1+fixZ2)/2;

y3:=(fixY1+fixY2)/2+Ddet;

writeln(ff1,'(create\_bspline :wire\_part "/Sil\_',N\_Tr:1,'.',N\_Spl:1,'" ',

':input\_mode :control ');

Vint(N\_Tr,1,1,N\_Razb[N\_Act]);

writeln(ff1,x3:8:8,',',y3:1:8,',',z3:1:8);

Vint(N\_Tr,1,3,N\_Razb[N\_Act]);}

writeln(ff1,'(create\_bspline :wire\_part "/Sil\_',N\_Tr:1,'.',N\_Spl:1,'" ',

':start\_condition :edge :cv\_edge ',

':start :selected\_part "/Sil\_',N\_Tr:1,'.',1:1,'" :edge\_3d :by\_vertex\_3d');

Empt[N\_Sp]:=1;

Vint(N\_Tr,1,1,N\_Razb[N\_Act]);

writeln(ff1,':select\_done ');

Vint(N\_Tr,1,1,N\_Razb[N\_Act]);

writeln(ff1,':tangent 0.2 :accept ',

':end\_condition :edge :cv\_edge ',

':start :selected\_part "/Sil\_',N\_Tr:1,'.',3:1,'" :edge\_3d :by\_vertex\_3d');

Vint(N\_Tr,1,3,N\_Razb[N\_Act]);

writeln(ff1,':select\_done ');

Vint(N\_Tr,1,3,N\_Razb[N\_Act]);

writeln(ff1,':tangent 0.2 :accept ');

writeln(ff1,')');

writeln(ff1,'(remove\_from\_vp\_drawlist "vport1" :with-wp ',

' "/Sil\_',N\_Tr:1,'.',N\_Spl:1,'")');

end;

begin

writeln(ff1,'(create\_bspline :wire\_part "/Sil\_',N\_Tr:1,'.',N\_Spl:1,'" ',

':start\_condition :edge :cv\_edge ',

':start :selected\_part "/Sil\_',N\_Tr:1,'.',5:1,'" :edge\_3d :by\_vertex\_3d ');

Empt[N\_Sp]:=1;

Vint(N\_Tr,2,1,N\_Razb[N\_Act]);

fixX1:=fixX; fixY1:=fixY; fixZ1:=fixZ;

writeln(ff1,':select\_done ');

Vint(N\_Tr,2,1,N\_Razb[N\_Act]);

writeln(ff1,':accept ',

':end\_condition :edge :cv\_edge ',

':start :selected\_part "/Sil\_',N\_Tr:1,'.',7:1,'" :edge\_3d :by\_vertex\_3d ');

Vint(N\_Tr,2,3,N\_Razb[N\_Act]);

fixX2:=fixX; fixY2:=fixY; fixZ2:=fixZ;

writeln(ff1,':select\_done ');

Vint(N\_Tr,2,3,N\_Razb[N\_Act]);

writeln(ff1,':accept )');

writeln(ff1,'(modify\_bspline :curve :start :selected\_part "/Sil\_',N\_Tr:1,'.8" ',

':all\_3d :select\_done ',

':insert\_point ',fixX1:8:5,',',

fixY1:1:5,',',

fixZ1:1:5,' ',

(fixX1+fixX2)/2:8:5,',',

(fixY1+fixY2)/2:1:5,',',

(fixZ1+fixZ2)/2:1:5,' ');

writeln(ff1,')');

end;

end;

end;

begin

for i:=1 to 10 do

Empt[i]:=0;

AssignFile(ff1,Path+'LSP\Sil.lsp');

Append(ff1);

N\_Spl:=0;

writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug" :select\_done ',

':translate :x ',dX:8:5,')');

writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug" :select\_done ',

':translate :y ',dY:8:5,')');

writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug" :select\_done ',

':translate :z ',dZ:8:5,')');

{ if N\_Tr=N\_Sech[N\_Act] then

writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug.1" :select\_done ',

':rotate :axis :x :rotation\_angle ',-fiA/pi\*180:8:5,' :done :translate :x ',

-dX:8:5,')');

}

{ if N\_Tr=N\_Sech[N\_Act] then

begin

writeln(ff1,'(create\_multiple\_pa :copy :source "/a1/surf1" :owner "/" )');

writeln(ff1,'(create\_multiple\_pa :copy :source "/a1/surf2" :owner "/" )');

writeln(ff1,'(create\_multiple\_pa :copy :source "/a1/surf3" :owner "/" )');

end; }

Create\_Bspline(N\_Tr,1,1,1);

Create\_Bspline(N\_Tr,2,2,1);

Create\_Bspline(N\_Tr,3,3,1);

Create\_Bspline(N\_Tr,4,1,3);

for i:=1 to N\_Spl do

writeln(ff1,'(position\_pa :part\_assembly :start "/Sil\_',N\_Tr:1,'.',i:1,'" ',

':select\_done ',

':rotate :axis :x :rotation\_angle ',(fiAmax-fiA)/pi\*180:8:5,')');

if N\_Tr<N\_Sech[N\_Act] then

begin

// writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug" :select\_done ',

// ':rotate :axis :x :rotation\_angle ',fiA/pi\*180:8:5,')');

writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug" :select\_done ',

':translate :x ',-dX:8:5,')');

writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug" :select\_done ',

':translate :y ',-dY:8:5,')');

writeln(ff1,'(position\_pa :part\_assembly :start "/a1/Krug" :select\_done ',

':translate :z ',-dZ:8:5,')');

end;

//W\_Str1('N\_spl='+w\_i(N\_Spl,1));

Flush(ff1);

CloseFile(ff1);

end;

Приложение Б

Параметры проекта Delphi:

-$A8

-$B-

-$C+

-$D+

-$E-

-$F-

-$G+

-$H+

-$I+

-$J-

-$K-

-$L+

-$M-

-$N+

-$O+

-$P+

-$Q-

-$R-

-$S-

-$T-

-$U-

-$V+

-$W-

-$X+

-$YD

-$Z1

-cg

-AWinTypes=Windows;WinProcs=Windows;DbiTypes=BDE;DbiProcs=BDE;DbiErrs=BDE;

-H+

-W+

-M

-$M16384,1048576

-K$00400000

-LE"c:\program files\borland\delphi7\Projects\Bpl"

-LN"c:\program files\borland\delphi7\Projects\Bpl"

-w-UNSAFE\_TYPE

-w-UNSAFE\_CODE

-w-UNSAFE\_CAST

[FileVersion]

Version=7.0 //версия файла

[Compiler]

A=8 //начальные параметры

B=0

C=1

D=1

E=0

F=0

G=1

H=1

I=1

J=0

K=0

L=1

M=0

N=1

O=1

P=1

Q=1

R=1

S=0

T=0

U=0

V=1

W=1

X=1

Y=1

Z=1

ShowHints=1

ShowWarnings=1

UnitAliases=WinTypes=Windows;WinProcs=Windows;DbiTypes=BDE;DbiProcs=BDE;DbiErrs=BDE;

NamespacePrefix=

SymbolDeprecated=1

SymbolLibrary=1

SymbolPlatform=1

UnitLibrary=1

UnitPlatform=1

UnitDeprecated=1

HResultCompat=1

HidingMember=1

HiddenVirtual=1

Garbage=1

BoundsError=1

ZeroNilCompat=1

StringConstTruncated=1

ForLoopVarVarPar=1

TypedConstVarPar=1

AsgToTypedConst=1

CaseLabelRange=1

ForVariable=1

ConstructingAbstract=1

ComparisonFalse=1

ComparisonTrue=1

ComparingSignedUnsigned=1

CombiningSignedUnsigned=1

UnsupportedConstruct=1

FileOpen=1

FileOpenUnitSrc=1

BadGlobalSymbol=1

DuplicateConstructorDestructor=1

InvalidDirective=1

PackageNoLink=1

PackageThreadVar=1

ImplicitImport=1

HPPEMITIgnored=1

NoRetVal=1

UseBeforeDef=1

ForLoopVarUndef=1

UnitNameMismatch=1

NoCFGFileFound=1

MessageDirective=1

ImplicitVariants=1

UnicodeToLocale=1

LocaleToUnicode=1

ImagebaseMultiple=1

SuspiciousTypecast=1

PrivatePropAccessor=1

UnsafeType=1

UnsafeCode=1

UnsafeCast=1

[Linker]

MapFile=0

OutputObjs=0

ConsoleApp=1

DebugInfo=0

RemoteSymbols=0

MinStackSize=16384

MaxStackSize=1048576

ImageBase=4194304

ExeDescription=

[Directories] //параметры проекта

OutputDir=

UnitOutputDir=

PackageDLLOutputDir=

PackageDCPOutputDir=

SearchPath=

Packages=vcl;rtl;dbrtl;adortl;vcldb;vclx;bdertl;vcldbx;ibxpress;dsnap;cds;bdecds;qrpt;teeui;teedb;tee;dss;teeqr;visualclx;visualdbclx;dsnapcrba;dsnapcon;VclSmp;vclshlctrls;inetdb;inet;nmfast;vclie;dbexpress;dbxcds;indy;dclOffice2k

Conditionals=

DebugSourceDirs=

UsePackages=0

[Parameters]

RunParams=

HostApplication=

Launcher=

UseLauncher=0

DebugCWD=

[Language]

ActiveLang=

ProjectLang=

RootDir=

[Version Info]

IncludeVerInfo=0

AutoIncBuild=0

MajorVer=1

MinorVer=0

Release=0

Build=0

Debug=0

PreRelease=0

Special=0

Private=0

DLL=0

Locale=1049

CodePage=1251

[Version Info Keys] //информация о проекте

CompanyName=

FileDescription=

FileVersion=1.0.0.0

InternalName=

LegalCopyright=

LegalTrademarks=

OriginalFilename=

ProductName=

ProductVersion=1.0.0.0

Comments=

[HistoryLists\hlUnitAliases]

Count=1

Item0=WinTypes=Windows;WinProcs=Windows;DbiTypes=BDE;DbiProcs=BDE;DbiErrs=BDE;

Приложение B

Текст настроечного файла программного модуля:

3

2

5.260 5.000 105.000 1.000 1.250

0.700 0.900 7.410 0.350 120.000

10.000 25.000 0.900 30.000 0.200

85.000 1.300 50.000 71.000 0.060 3

3

99.86600 9.70600 90.00000

0.45000 0.20000 50.78000

-58.71300 53.29000 -196.78000 -180.36000

125.06600 4.05600 90.00000

1.95000 1.95000 50.73000

-71.26300 65.89000 -147.21000 -136.44000

100.16600 9.97300 90.00000

0.10000 1.10000 50.69000

-58.77300 53.44000 -105.12700 -88.44000

130.00000 88.00000 6.71400

6

Канавка

0.000 0.00000 40.00000 53.50000 1 1 0.00000 0.71966 0.85300 -0.2000 99999999.000 0 2.000 2 0.000 0 1 1 1 1 20 0.0100 1

0.000 0.000 0.000 0.000 0.000 0.000

Затылок

0.000 -0.01719 40.00000 47.00000 1 1 0.00000 5.14988 -4.85300 -1.0000 999999.000 0 1.000 2 0.000 0 1 1 1 1 20 0.0100 1

0.000 0.000 0.000 421.303 0.000 0.000

1 задний угол

58.525 0.00000 80.00000 80.00000 3 2 0.25000 -30.00000 0.00000 0.5000 999999.000 0 2.000 2 0.000 0 1 1 1 1 20 0.0100 1

0.000 0.000 0.000 -70.110 0.000 0.000

2 задний угол

51.710 0.00000 80.00000 80.00000 3 2 -0.65000 -30.00000 -1.26552 0.5000 999999.000 0 2.000 2 0.000 0 0 1 1 1 20 0.0100 1

0.000 0.000 0.000 -46.533 0.000 0.000

Подточка

30.000 40.00000 25.00000 24.00000 2 1 1.00000 -0.80000 -0.24500 -9.0000 999999.000 0 2.000 2 0.000 0 0 1 1 0 20 0.0100 1

0.000 0.000 0.000 -45.000 0.000 0.000

Канавка под СОЖ

14.767 79.65473 25.00000 20.00000 3 2 -3.21700 2.11332 -1.94988 0.5000 999999.000 0 2.000 2 0.000 0 0 0 0 0 20 0.0100 1

0.000 0.000 0.000 0.000 0.000 0.000

4

0.000 0.000 90.000 0.000

10.520 0.000 -0.016 0.000

10.477 77.000 45.246 0.000

12.000 77.755 0.000 0.000

12.000 130.000 0.000 0.000

39.000 19.000 59.000 40.000

1 1.900 6.000 45.000 1.000 15.000 45.000

1

1

1

1

0

0

0

0

0

0

0

0

0.000

180.000

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 -15.000 0.000 0.000

0.000 0.000 0.000 -7.500 0.000 0.000

0 0 0 20 0 0

2

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0

0.000 0.000 0.000 421.303 0.000 0.000

0 0 0 20 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 -50.512 0.000 0.000

0.000 -0.240 0.000 -25.256 0.000 0.000

0 20 0 20 0 0

7

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0

0.000 0.000 0.000 -70.110 0.000 0.000

0 0 0 20 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

1

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0

0.000 0.000 0.000 -46.533 0.000 0.000

0 0 0 20 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

1

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0

0.000 0.000 0.000 -45.000 0.000 0.000

0 0 0 20 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

1

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

1

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0

0.000 0.000 0.000 0.000 0.000 0.000

0 20 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 20 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 20 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 20 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 20 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 20 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

0.000 0.000 0.000 0.000 0.000 0.000

0 0 0 0 0 0

1 1 1 1

15.00000 2

0.500 2.500 2.475 2.550 30.000

0.000 0.00000

0.100 0.100 7.500 0.500

1 2 2 1

50.51161 7

0.500 2.500 2.475 2.550 30.000

0.000 0.00000

0.700 0.900 7.419 0.350

1 3 3 1

0.00000 1

0.000 0.000 5.260 0.000 30.000

0.000 10.00000

0.100 0.100 7.500 0.500

1 3 3 1

0.00000 1

0.500 2.500 2.475 2.550 30.000

4.000 25.00000

0.100 0.100 7.500 0.500

0 0 5 1

0.00000 1

0.500 2.500 2.475 2.550 30.000

0.000 0.00000

0.100 0.100 7.500 0.500

0 0 6 1

0.00000 1

0.500 2.500 2.475 2.550 30.000

0.000 0.00000

0.100 0.100 7.500 0.500

Приложение Г

Текст файлов связи с пакетом 3D моделирования:

(create\_workplane :new)

(bspline\_int

0.00000,-49.48300 :tangent -90.00000 //точки сплайна

0.00139,-49.51831 :tangent -85.50000

0.00554,-49.55340 :tangent -81.00000

0.01243,-49.58805 :tangent -76.50000

0.02202,-49.62206 :tangent -72.00000

0.03425,-49.65521 :tangent -67.50000

0.04905,-49.68730 :tangent -63.00000

0.06631,-49.71812 :tangent -58.50000

0.08594,-49.74750 :tangent -54.00000

0.10782,-49.77525 :tangent -49.50000

0.13180,-49.80120 :tangent -45.00000

0.15775,-49.82518 :tangent -40.50000

0.18550,-49.84706 :tangent -36.00000

0.21488,-49.86669 :tangent -31.50000

0.24570,-49.88395 :tangent -27.00000

0.27779,-49.89875 :tangent -22.50000

0.31094,-49.91098 :tangent -18.00000

0.34495,-49.92057 :tangent -13.50000

0.37960,-49.92746 :tangent -9.00000

0.41469,-49.93161 :tangent -4.50000

0.45000,-49.93300 :tangent 0.00000

0.90280,-49.93300 :tangent 0.00000

1.35560,-49.93300 :tangent 0.00000

1.80840,-49.93300 :tangent 0.00000

2.26120,-49.93300 :tangent 0.00000

2.71400,-49.93300 :tangent 0.00000

3.16680,-49.93300 :tangent 0.00000

3.61960,-49.93300 :tangent 0.00000

4.07240,-49.93300 :tangent 0.00000

4.52520,-49.93300 :tangent 0.00000

4.97800,-49.93300 :tangent 0.00000

5.43080,-49.93300 :tangent 0.00000

5.88360,-49.93300 :tangent 0.00000

6.33640,-49.93300 :tangent 0.00000

6.78920,-49.93300 :tangent 0.00000

7.24200,-49.93300 :tangent 0.00000

7.69480,-49.93300 :tangent 0.00000

8.14760,-49.93300 :tangent 0.00000

8.60040,-49.93300 :tangent 0.00000

9.05320,-49.93300 :tangent 0.00000

9.50600,-49.93300 :tangent 0.00000

9.52169,-49.93238 :tangent 4.50000

9.53729,-49.93054 :tangent 9.00000

9.55269,-49.92747 :tangent 13.50000

9.56780,-49.92321 :tangent 18.00000

9.58254,-49.91778 :tangent 22.50000

9.59680,-49.91120 :tangent 27.00000

9.61050,-49.90353 :tangent 31.50000

9.62356,-49.89480 :tangent 36.00000

9.63589,-49.88508 :tangent 40.50000

9.64742,-49.87442 :tangent 45.00000

9.65808,-49.86289 :tangent 49.50000

9.66780,-49.85056 :tangent 54.00000

9.67653,-49.83750 :tangent 58.50000

9.68420,-49.82380 :tangent 63.00000

9.69078,-49.80954 :tangent 67.50000

9.69621,-49.79480 :tangent 72.00000

9.70047,-49.77969 :tangent 76.50000

9.70354,-49.76429 :tangent 81.00000

9.70538,-49.74869 :tangent 85.50000

9.70600,-49.73300 :tangent 90.00000

)

(polygon 0.00000,-49.48300 0.00000,0.00000 9.70600,0.00000 9.70600,-49.73300)

(rotate\_2d :select :start :all\_2d :select\_done :angle -90.0)

( turn :sel\_part "/Krug" :keep\_wp :yes :keep\_profile :no :axis :v :rotation\_angle 360.0)

(define\_feature :selection :start :selected\_part "/Krug" :spline\_sf :all\_3d :select\_done)

(create\_assembly)

(change\_pa\_owner :new\_owner "/a1" :parts\_assemblies :start "/Krug" :select\_done)

(position\_pa :part\_assembly :start "/a1" :select\_done :rotate :axis :x :rotation\_angle -90.0 :done :translate :y 50.65266000 :rotate :axis :y :rotation\_angle -0.00000000)

(position\_pa :part\_assembly :start "/a1" :select\_done :translate :x 40.00000000)

(position\_pa :part\_assembly :start "/a1" :select\_done :translate :z 0.85300000)

(position\_pa :part\_assembly "/a1/Krug" :rotate :axis :two\_pta 40.00000000,0.71966000,0.85300000 40.00000000,50.65266000,0.85300000 :rotation\_angle -0.20000000)

(create\_multiple\_pa :copy :source "/a1/Krug")

(remove\_from\_vp\_drawlist "vport1" :with-wp "/a1/Krug" )

(remove\_from\_vp\_drawlist "vport1" :with-wp "/a1/Krug.1" )

( delete\_3d "/w1")

Приложение Д

Текст управляющей программы ЧПУ:

%\_N\_CIP6\Emul\Bdd\9013\_MPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

DEF INT OP1,OP2,OP3,OP4,OP5,OP6

OP1=1

OP2=1

OP3=1

OP4=1

OP5=1

OP6=1

IF OP1==1

Op\_1

ENDIF

IF OP2==1

Op\_2

ENDIF

IF OP3==1

Op\_3

ENDIF

IF OP4==1

Op\_4

ENDIF

IF OP5==1

Op\_5

ENDIF

IF OP6==1

Op\_6

ENDIF

M10 M19

M02

%\_N\_NACH\_UST\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

G01 G90 Y176.080 F5000 G09

G01 G90 A=0 F5000 G09

G01 G90 B=0 F5000 G09

G01 G90 Z=0 F5000 G09

G01 G90 X-496.007 F5000 G09

G01 G90 C=0 F5000 G09

M17

%\_N\_Op\_1\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

;======= Канавка =======

EXTERN GLAV\_DVIG\_Op1(INT,REAL)

EXTERN OTVOD\_Op1(INT)

EXTERN PRIPUSK\_Op1(INT,INT)

EXTERN CYCLOBR\_Op1(INT)

EXTERN VIX\_Op1

DEF REAL X\_KOR,Z\_KOR,B\_KOR,A\_KOR,Y\_KOR,C\_KOR

DEF REAL NZ\_FI[3],FPR[4]

DEF INT NZ\_ACT[3]

DEF INT I\_1,I\_2

R306=0

MSG(" ")

NACH\_UST

Коррекция:

X\_KOR=0;

Z\_KOR=0;

B\_KOR=0;

A\_KOR=0;

Y\_KOR=0;

C\_KOR=0;

Подвод:

DISPOUT\_Op1

G01 G91 X= 405.225+X\_KOR G09 F=5000

DISPOUT\_Op1

G01 G91 Z=-193.959+Z\_KOR G09 F=5000

DISPOUT\_Op1

G01 G91 B= 0.200+B\_KOR G09 F=5000

DISPOUT\_Op1

G01 G91 Y=-125.427+Y\_KOR G09 F=5000

DISPOUT\_Op1

G01 G91 A= -0.000+A\_KOR G09 F=5000

DISPOUT\_Op1

G01 G91 C= 0.000+C\_KOR G09 F=5000

Перемещения гл. движения:

R298= 93.499;X

R299= 0.000;Y

R300= -0.326;Z

R301= 0.000;A

; ПОЛОЖЕНИЕ ЗУБЬЕВ

; У Г О Л Акт.зуб

NZ\_FI[1]= 0.000 NZ\_ACT[1]=1

NZ\_FI[2]= 180.000 NZ\_ACT[2]=1

; П Р И П У С К И

; 1проход Подача 2проход Подача 3проход Подача

R213=0.000 R231= 20 R219=0.000 R237= 0 R225=0.000 R243= 0 ;Y

FPR[1]=R231

FPR[2]=R237

FPR[3]=R243

; ЦИКЛИЧЕСКАЯ ОБРАБОТКА

; СТАРТ Экстрем. ФИНИШ Подача

R264= 0.000 R270= -7.500 R276= -15.000 R282= 20 ;A

R260=2;Кол-во циклов обработки

; О Б Р А Б О Т К А

R306=0

FOR I\_1=1 TO 3

R307=0

IF FPR[I\_1]>0

PRIPUSK\_Op1(I\_1,1)

FOR I\_2=1 TO 2

DISPOUT\_Op1

G01 G91 A=NZ\_FI[I\_2] F=5000

IF NZ\_ACT[I\_2]==1

R308=I\_2

CYCLOBR\_Op1(R260,FPR[I\_1])

ENDIF

ENDFOR

DISPOUT\_Op1

G01 G91 A=360-NZ\_FI[2] F=5000

PRIPUSK\_Op1(I\_1,-1)

ENDIF

ENDFOR

M17

%\_N\_GLAV\_DVIG\_Op1\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC GLAV\_DVIG\_Op1(INT NAPR,REAL FGLDV)

DISPOUT\_OP1

G01 G91 X=NAPR\*R298 Z=NAPR\*R300 F=FGLDV

M17

%\_N\_PRIPUSK\_Op1\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC PRIPUSK\_Op1(INT NUM\_PR,INT NAPR\_PR, REAL FPR)

IF (NUM\_PR==1) AND (FPR>0)

R307=NUM\_PR

DISPOUT\_Op1

G01 G91 Y=NAPR\_PR\*R213 F=200

ENDIF

IF (NUM\_PR==2) AND (FPR>0)

R307=NUM\_PR

DISPOUT\_Op1

G01 G91 Y=NAPR\_PR\*R219 F=200

ENDIF

IF (NUM\_PR==3) AND (FPR>0)

R307=NUM\_PR

DISPOUT\_Op1

G01 G91 Y=NAPR\_PR\*R225 F=200

ENDIF

M17

%\_N\_OTVOD\_Op1\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC OTVOD\_Op1(INT NAPR\_OTV)

M17

%\_N\_CYCLOBR\_Op1\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC CYCLOBR\_Op1(INT N\_CYCL,REAL FPR)

EXTERN GLAV\_DVIG\_Op1(INT,REAL)

EXTERN OTVOD\_Op1(INT)

DEF INT I\_1,I\_2

DEF REAL STEP1,STEP2

FOR I\_1=1 TO N\_CYCL

R309=I\_1

IF R282>0

IF N\_CYCL==2

G01 G91 A=(I\_1-1)\*(R276-R264)

ENDIF

IF N\_CYCL>2

STEP1=2\*(R270-R264)/(N\_CYCL-1)

STEP2=2\*(R276-R270)/(N\_CYCL-1)

I\_2=ROUND((N\_CYCL-1)/2)

IF I\_1<=I\_2

IF I\_1>1

G01 G91 A=STEP1

ENDIF

ENDIF

IF I\_1>I\_2

G01 G91 A=STEP2

ENDIF

ENDIF

ENDIF

GLAV\_DVIG\_Op1(1,FPR)

OTVOD\_Op1(1)

GLAV\_DVIG\_Op1(-1,500)

OTVOD\_Op1(-1)

ENDFOR

IF R282>0

DISPOUT\_OP1

G01 G91 A=-(R276-R264) F=R282

ENDIF

M17

%\_N\_DISPOUT\_Op1\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

R306=R306+1

MSG (" Поз. N " <<R306<< " Прип.N " <<R307<< " Зуб.N " <<R308<< "Цикл.N " <<R309)

M17

%\_N\_Op\_2\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

;======= Затылок =======

EXTERN GLAV\_DVIG\_Op2(INT,REAL)

EXTERN OTVOD\_Op2(INT)

EXTERN PRIPUSK\_Op2(INT,INT)

EXTERN CYCLOBR\_Op2(INT)

EXTERN VIX\_Op2

DEF REAL X\_KOR,Z\_KOR,B\_KOR,A\_KOR,Y\_KOR,C\_KOR

DEF REAL NZ\_FI[3],FPR[4]

DEF INT NZ\_ACT[3]

DEF INT I\_1,I\_2

R306=0

MSG(" ")

NACH\_UST

Коррекция:

X\_KOR=0;

Z\_KOR=0;

B\_KOR=0;

A\_KOR=0;

Y\_KOR=0;

C\_KOR=0;

Подвод:

DISPOUT\_Op2

G01 G91 X= 405.341+X\_KOR G09 F=5000

DISPOUT\_Op2

G01 G91 Z=-186.986+Z\_KOR G09 F=5000

DISPOUT\_Op2

G01 G91 B= 1.000+B\_KOR G09 F=5000

DISPOUT\_Op2

G01 G91 Y=-120.985+Y\_KOR G09 F=5000

DISPOUT\_Op2

G01 G91 A= 0.000+A\_KOR G09 F=5000

DISPOUT\_Op2

G01 G91 C= 0.000+C\_KOR G09 F=5000

Перемещения гл. движения:

R298= 86.987;X

R299= -0.026;Y

R300= -1.518;Z

R301= 0.000;A

; ПОЛОЖЕНИЕ ЗУБЬЕВ

; У Г О Л Акт.зуб

NZ\_FI[1]= 0.000 NZ\_ACT[1]=1

NZ\_FI[2]= 180.000 NZ\_ACT[2]=1

; ДОПОЛНИТЕЛЬНЫЕ СМЕЩЕНИЯ

; Смещение Подача

R203= 421.303 R209= 20 ;A

; П Р И П У С К И

; 1проход Подача 2проход Подача 3проход Подача

R213=0.000 R231= 20 R219=0.000 R237= 0 R225=0.000 R243= 0 ;Y

FPR[1]=R231

FPR[2]=R237

FPR[3]=R243

; ЦИКЛИЧЕСКАЯ ОБРАБОТКА

; СТАРТ Экстрем. ФИНИШ Подача

R262= 0.000 R268= -0.240 R274= 0.000 R280= 20 ;Y

R264= 0.000 R270= -25.256 R276= -50.512 R282= 20 ;A

R260=7;Кол-во циклов обработки

; О Б Р А Б О Т К А

R306=0

DOPSM\_Op2

FOR I\_1=1 TO 3

R307=0

IF FPR[I\_1]>0

PRIPUSK\_Op2(I\_1,1)

FOR I\_2=1 TO 2

DISPOUT\_Op2

G01 G91 A=NZ\_FI[I\_2] F=5000

IF NZ\_ACT[I\_2]==1

R308=I\_2

CYCLOBR\_Op2(R260,FPR[I\_1])

ENDIF

ENDFOR

DISPOUT\_Op2

G01 G91 A=360-NZ\_FI[2] F=5000

PRIPUSK\_Op2(I\_1,-1)

ENDIF

ENDFOR

M17

%\_N\_GLAV\_DVIG\_Op2\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC GLAV\_DVIG\_Op2(INT NAPR,REAL FGLDV)

DISPOUT\_OP2

G01 G91 X=NAPR\*R298 Y=NAPR\*R299 Z=NAPR\*R300 F=FGLDV

M17

%\_N\_DOPSM\_Op2\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC DOPSM\_Op2

IF R209>0

DISPOUT\_Op2

G01 G91 A=R203 F=R209

ENDIF

M17

%\_N\_PRIPUSK\_Op2\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC PRIPUSK\_Op2(INT NUM\_PR,INT NAPR\_PR, REAL FPR)

IF (NUM\_PR==1) AND (FPR>0)

R307=NUM\_PR

DISPOUT\_Op2

G01 G91 Y=NAPR\_PR\*R213 F=200

ENDIF

IF (NUM\_PR==2) AND (FPR>0)

R307=NUM\_PR

DISPOUT\_Op2

G01 G91 Y=NAPR\_PR\*R219 F=200

ENDIF

IF (NUM\_PR==3) AND (FPR>0)

R307=NUM\_PR

DISPOUT\_Op2

G01 G91 Y=NAPR\_PR\*R225 F=200

ENDIF

M17

%\_N\_OTVOD\_Op2\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC OTVOD\_Op2(INT NAPR\_OTV)

M17

%\_N\_CYCLOBR\_Op2\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

PROC CYCLOBR\_Op2(INT N\_CYCL,REAL FPR)

EXTERN GLAV\_DVIG\_Op2(INT,REAL)

EXTERN OTVOD\_Op2(INT)

DEF INT I\_1,I\_2

DEF REAL STEP1,STEP2

FOR I\_1=1 TO N\_CYCL

R309=I\_1

IF R280>0

IF N\_CYCL==2

G01 G91 Y=(I\_1-1)\*(R274-R262)

ENDIF

IF N\_CYCL>2

STEP1=2\*(R268-R262)/(N\_CYCL-1)

STEP2=2\*(R274-R268)/(N\_CYCL-1)

I\_2=ROUND((N\_CYCL-1)/2)

IF I\_1<=I\_2

IF I\_1>1

G01 G91 Y=STEP1

ENDIF

ENDIF

IF I\_1>I\_2

G01 G91 Y=STEP2

ENDIF

ENDIF

ENDIF

IF R282>0

IF N\_CYCL==2

G01 G91 A=(I\_1-1)\*(R276-R264)

ENDIF

IF N\_CYCL>2

STEP1=2\*(R270-R264)/(N\_CYCL-1)

STEP2=2\*(R276-R270)/(N\_CYCL-1)

I\_2=ROUND((N\_CYCL-1)/2)

IF I\_1<=I\_2

IF I\_1>1

G01 G91 A=STEP1

ENDIF

ENDIF

IF I\_1>I\_2

G01 G91 A=STEP2

ENDIF

ENDIF

ENDIF

GLAV\_DVIG\_Op2(1,FPR)

OTVOD\_Op2(1)

GLAV\_DVIG\_Op2(-1,500)

OTVOD\_Op2(-1)

ENDFOR

IF R280>0

DISPOUT\_OP2

G01 G91 Y=-(R274-R262) F=R280

ENDIF

IF R282>0

DISPOUT\_OP2

G01 G91 A=-(R276-R264) F=R282

ENDIF

M17

%\_N\_DISPOUT\_Op2\_SPF

;$PATH=/\_N\_WKS\_DIR/\_N\_CIP6\Emul\Bdd\9013\_WPD

R306=R306+1

MSG (" Поз. N " <<R306<< " Прип.N " <<R307<< " Зуб.N " <<R308<< "Цикл.N " <<R309)

M17