

Аннотация

Настоящий документ содержит расчетно-пояснительную записку к дипломному проекту на тему «Разработка прототипа программного комплекса ПА10».

Программный продукт предназначен для решения систем ОДУ-ДАУ и предоставляет пользователю два альтернативных варианта интерфейса: схемный редактор и область ввода уравнений в текстовом виде.

В дипломном проекте рассмотрен процесс разработки математического модуля на основе программы manzhuk из библиотеки SADEL (Sets of Algebraic and Differential Equations solvers Library) - Си библиотеки для решения плохообусловленных алгебраических и жестких систем обыкновенных дифференциальных уравнений с максимально возможной компьютерной точностью, а также разработки модулей ввода и вывода данных. Рассмотрены средства создания лексических и синтаксических анализаторов, вопросы использования библиотеки Qt 4.8 для создания графического интерфейса программы. Приведено решение некоторых тестовых задач.

Оглавление

Техническое задание	6
Глава 1 Исследовательская часть	8
1.1 Моделирование динамики технических систем	8
1.2 Обобщенные формальные схемы	9
1.3 Базовые двухполюсники	11
1.4 Расширенный базис	12
1.5 Описание системы ПА9	13
Глава 2 Конструкторская часть	15
2.1 Библиотека manzhuk.....	15
2.1.1 Описание библиотеки.....	15
2.1.2 Работа с библиотекой	16
2.1.1 Функция fct	23
2.2 Библиотека Qt	24
2.3 Механизм слотов и сигналов	26
Глава 3 Технологическая часть	29
3.1 Вычислительный модуль	29
3.1.1 Представление системы уравнений	29
3.1.2 Представление выражений.....	29
3.1.3 Получение Якобиана	30
3.1.4 Взаимодействие с решателем.....	31
3.2 Модуль ввода	32
3.2.1 Генератор лексических анализаторов Flex.....	32
3.2.2 Генератор синтаксических анализаторов Bison.....	33
3.2.3 Ввод системы уравнений.....	35
3.2.4 Начало вычислений	36
3.2.5 Схемный редактор	38

3.2.6	Моделирование схемы	41
3.3	Qwt. Отображение данных	42
3.4	Тесты	43
3.4.1	Тест Ван дер Поля	43
3.4.2	RLC-цепь	44
Глава 4	Экономическая часть	46
4.1	Введение	46
4.2	Основные этапы проекта разработки программного продукта	46
4.3	Расчет трудоемкости разработки программного продукта	47
4.3.1	Трудоемкость разработки технического задания	49
4.3.2	Трудоемкость разработки эскизного проекта	50
4.3.3	Трудоемкость разработки технического проекта	51
4.3.4	Трудоемкость разработки рабочего проекта	52
4.3.5	Трудоемкость выполнения стадии «Внедрение»	53
4.4	Расчет затрат на реализацию программного продукта	54
4.4.1	Расчет материальных затрат	54
4.4.2	Расчет амортизационных отчислений	54
4.4.3	Расчет заработной платы	55
4.4.4	Расчет отчислений в социальные фонды	57
4.4.5	Прочие расходы	58
4.5	Определение цены программного продукта	58
4.6	Выводы	60
Глава 5	Экологическая часть	61
5.1	Введение	61
5.2	Основные факторы	61
5.3	Общие положения организации рабочего места	62
5.3.1	Обеспечение параметров микроклимата	64

5.3.2	Эргономичность.....	65
5.3.3	Оптимальное размещение оборудования	67
5.3.4	Обеспечение электробезопасности.....	67
5.3.5	Обеспечение допустимого уровня шума.....	68
5.3.6	Обеспечение допустимых эргономических характеристики дисплеев	68
5.3.7	Обеспечение пожаробезопасности	69
5.3.8	Освещенность	69
5.4	Расчет системы искусственного освещения	72
5.4.1	Выбор источников света	72
5.4.2	Выбор системы освещения	72
5.4.3	Выбор осветительных приборов.....	73
5.4.4	Размещение осветительных приборов.....	73
5.4.5	Выбор освещенности и коэффициента запаса	75
5.4.6	Расчет системы искусственного освещения.....	75
5.4.7	Утилизация элементов системы искусственного освещения....	78
5.5	Выводы.....	79
	Заключение	80
	Список литературы	81

Техническое задание

Разработать прототип программного комплекса ПА10.

1. Основания для разработки.

Основанием для разработки является учебный план кафедры РК6 на 11-й семестр, утвержденный заведующим кафедрой.

2. Назначение разработки.

Система ПА10 предназначена для анализа динамики различных (механических, гидравлических, пневматических, тепловых и смешанных) технических систем, функционирование которых описывается системой ДАУ

Одним из применений разрабатываемой системы является использование её для проведения лабораторных работ по курсу «Основы Автоматизированного Проектирования» с целью обучения студентов современным методикам проектирования технических устройств и анализа их динамических характеристик.

3. Требования к программе или программному изделию.

3.1. Требования к функциональным характеристикам.

Разрабатываемая система должна обладать следующими функциями:

- а. производить расчет задачи с помощью программы manzhuk;
- б. предоставлять пользователю интерфейс непосредственного ввода системы уравнений для расчета;
- в. предоставлять пользователю интерфейс создания модели электрической схемы и ее расчета;
- г. иметь возможность настройки параметров интегрирования;
- д. иметь модуль визуализации выходных данных математического модуля;

3.2. Требования к надежности.

Система не должна аномально завершать свою работу при возникновении критической ошибки в одном из составляющих её модулей.

3.3. Требования к информационной и программной совместимости

Система должна быть разработана в среде Microsoft Visual Studio 2008 и иметь совместимость операционной системой ОС Windows XP/Vista7/8.

3.4. Требования к составу и параметрам технических средств.

Для нормальной работы клиентской части необходимо:

- а. ОС Windows XP/Vista7/8, установленная на ПК;
- б. наличие компилятора Microsoft Visual C++ ;
- в. наличие установленной библиотеки Qt 4.8.

3.5. Требования к маркировке и упаковке.

Не предъявляются.

3.6. Требования к транспортированию и хранению.

Не предъявляются.

3.7. Специальные требования.

Не предъявляются.

4. Требования к программной документации.

Программной документацией к разрабатываемой системе является расчётно-пояснительная записка.

5. Этапы разработки:

- а. разработка текстового анализатора,
- б. разработка виджета редактора уравнений,
- в. разработка виджета редактора схем,
- г. разработка виджета вывода графиков,
- д. разработка общего интерфейса и объединение модулей.

6. Порядок контроля и приемки.

Провести тесты программы, основанные на расчете заданных систем и электрических схем при заданных параметрах интегрирования.

1.1 Моделирование динамики технических систем

CAE, Computer-Aided Engineering – наукоемкий компьютерный инжиниринг, основанный на эффективном применении мультидисциплинарных надотраслевых систем инженерного анализа проектных решений (CAE Systems). CAE-системы предназначены для создания и анализа математических моделей, обладающих высоким уровнем адекватности реальным объектам и реальным физическим процессам, с целью оптимизации, обнаружения ошибок, снижения временных затрат и стоимости разработки проектного решения. Как правило, модели в CAE-системах описываются нестационарными нелинейными дифференциальными уравнениями в частных производных в пространственных областях сложной формы.

В настоящее время значительно возрастает важность компьютерных инженерных расчетов промышленных изделий и объектов с помощью CAE-систем. Опишем основные требования, предъявляемые к подобным системам.

- а. Точность. CAE-системы применяются для быстрой и точной оценки принимаемых решений, однако достоверность и точность получаемых решений систем уравнений важнее, чем затраты вычислительных ресурсов на их получение.
- б. Простота. Простота взаимодействия с программой дает возможность быстрого создания и изменения модели во время процесса проектирования. Многие из программных комплексов ориентированы на математиков-программистов и инженеров-расчетчиков, требуют высокой квалификации и специальных знаний.
- в. Универсальность. Большинство средств моделирования динамики технических объектов предполагают возможность работать в конкретной предметной области, например PSPICE в электронике, MSC.ADAMS в механике, MBTU для моделирования систем управления. В случаях, когда встают задачи комплексного моделирования сложных технических систем, состоящих из физически разнородных объектов, становится более

актуальным подход, в котором математическая модель составляется на основе метода аналогий переменных и уравнений математических моделей таких объектов. К системам моделирования такого типа относятся: Wolfram SystemModeler, последние программы серии ПА (версии 6 – 10).

1.2 Обобщенные формальные схемы

При математическом моделировании состояние любого технического объекта (ТО) в конкретный момент времени можно описать множеством некоторых физических величин, имеющих как постоянные значения, так и переменные значения в течение заданного отрезка времени моделирования. Обозначим множество переменных модели ТО вектором $V_{mo}(t)$. Конечной математической моделью динамических процессов в ТО в общем случае будет система дифференциально-алгебраических уравнений (ДАУ), замкнутая относительно этих переменных:

$$F(V_{mo}, \frac{dV_{mo}}{dt}, t) = 0 \quad 1,1$$

Вместе с тем любой ТО можно представить как множество элементов, связанных между собой множеством связей, поэтому его можно отобразить схемной моделью в виде некоторой обобщенной формальной схемы (ФС), состоящей из набора элементов, связанных между собой в некоторых узлах этой схемы. Такая схемная математическая модель технического объекта должна отображаться аналогичным множеством переменных $V_{\phi c}(t) = V_{mo}(t)$, а конечной математической моделью такой ФС должна быть система ДАУ, аналогичная системе (1).

Элементный состав обобщенной ФС и типы связей между элементами должны давать возможность моделирования как любых дифференциально-алгебраических уравнений вида (1), так и разнообразных элементов технических объектов любой физической природы (электрических, магнитных, механических, гидравлических, тепловых, оптических, акустических и др.) и разнообразных связей между такими элементами.

Состояние каждого узла связи в ФС в некоторый момент времени t опишем двумя типами безразмерных переменных:

- а. Переменная типа потока, втекающего в узел связи или вытекающего из узла связи $I_k(t)$. Эти переменные удовлетворяют уравнению непрерывности потоков для каждого узла ФС, которое будет базовым для получения модели ФС на основе моделей отдельных элементов ФС:

$$I_1 + I_2 + \dots + I_k = 0 \quad 1,2$$

- б. Переменная типа потенциала $P_j(t)$, которая определяется по отношению к некоторому базовому узлу, потенциал которого принимается равным нулю. Разновидностью переменной этого типа будет переменная типа напряжения (разности потенциалов) $U_{ij}(t) = P_j(t) - P_i(t)$, которая определяется как разность потенциалов между узлами связи схемных моделей элементов j и i .

Назовем узлы связи со стороны элементов ФС полюсами, тогда простейшими базовыми элементами ФС будут двухполюсники.




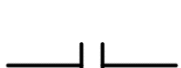
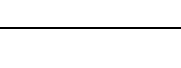
Двухполюсники опишем двумя типами переменных, которые будут составлять исходный базис переменных формальной схемы:

I_d – поток, протекающий через двухполюсник;

U_d – напряжение между полюсами двухполюсника.



1.3 Базовые двухполюсники

Таблица 1. Базовые двухполюсники

Название	Условное изображение	Задаваемые переменные		Уравнения
		Дифференциальные	Алгебраические	
Источник потока		-	I	$I - I_c = 0$
Источник потенциала		-	U	$U - E = 0$ $U - (\phi_1 - \phi_0) = 0$
Сопротивление		-	I_r, U_r	$I_r - U_r / R = 0$ $U_r - (\phi_1 - \phi_0) = 0$
Емкость		U_c	I_c	$I_c - C \frac{dU_c}{dt} = 0$ $U_c - (\phi_1 - \phi_0) = 0$
Индуктивность		I_l	U_l	$U_l - L \frac{dI_l}{dt} = 0$ $U_l - (\phi_1 - \phi_0) = 0$

Кроме того, базовыми элементами следует считать и узлы: обычный и базовый. Они так же, как и указанные выше элементы, задают свои переменные (потенциал в узле) и уравнения. Уравнения узлов описывают топологию схемы.

Таблица 2. Типы узлов

Название	Условное изображение	Задаваемые переменные		Уравнения
		Дифференциальные	Алгебраические	
Узел		-	φ	$I_1 + I_2 + \dots + I_k = 0$
Базовый узел		-	φ	$\phi_{node} = 0$

1.4 Расширенный базис

Обозначим через m общее количество двухполюсников типа C и L в некоторой схеме, тогда вектор базисных переменных ΦC , состоящей из n двухполюсников, будет иметь размерность $2n+m$, поскольку в этот вектор добавлены производные x_{pC} и x_{pL} для каждой емкости и индуктивности. Обозначим через k общее количество узлов, связывающих между собой двухполюсники, за исключением базового узла, обозначим также через $\Phi(t)$ – вектор узловых потенциалов ΦC , размерностью k и расширим базис переменных, включив этот вектор в вектор базисных переменных ΦC : $\Phi(t) \in V_{\phi c}(t)$, который будет иметь в результате размерность $2n+m+k$. Назовем этот базис обобщенным базисом ΦC .

Обозначим через φ_i и φ_j узлы подключения некоторого двухполюсника, тогда для этого двухполюсника можно добавить второе линейно-независимое уравнение, помимо основных уравнений (алгебраического для двухполюсников типа I, G, E, R или дифференциального для двухполюсников типа C и L):

$$U_d = \varphi_i - \varphi_j \quad 1,3$$

Таким образом, для n двухполюсников мы будем иметь $2n$ линейно-независимых алгебро-дифференциальных уравнений. Для каждого узла ΦC в свою очередь можно добавить линейно-независимое уравнение для суммы потоков, втекающих в этот узел:

$$I_1 + I_2 + \dots + I_k = 0 \quad 1,4$$

В результате мы получим $2n+k$ линейно-независимых уравнений, замкнутых относительно базисных переменных ΦC . m уравнений для m производных базисных переменных по времени дадут формулы интегрирования, в результате на каждом шаге интегрирования необходимо будет решать замкнутую систему нелинейных алгебраических уравнений (НАУ) размерностью $2n+m+k$ относительно неизвестного вектора $V_{\phi c}$. Это весьма высокая размерность, как будет видно из примеров, но система НАУ будет чрезвычайно разреженной (для

линейных систем как правило 2-3 ненулевых элемента в строке соответствующей матрицы Якоби). Предложенный обобщенный базис снимает многие ограничения при разработке моделей элементов технических объектов, присущие базисам переменных состояния, узловых потенциалов и расширенному базису, которые так или иначе связаны с методами интегрирования ДАУ. Введение производных дифференцируемых переменных (хрС и хрL) в базис позволяет разрабатывать абсолютно независимо и автономно программы интегрирования систем ДАУ, основанные, как на явных, так и неявных формулах интегрирования.

1.5 Описание системы ПА9

ПА9 – комплекс программ, предназначенный для анализа динамики электрических, механических, гидравлических, пневматических, тепловых и разнородных технических систем, основанный на методе физических аналогий. Моделируемый объект задается графическим изображением эквивалентной схемы, которая представляет собой совокупность связанных между собой по определенным правилам элементов, являющихся математическими моделями компонентов анализируемой технической системы. По графическому изображению эквивалентной схемы ПА9 автоматически формирует математическую модель в виде системы ДАУ, описывающей динамические процессы в исходной технической системе. Для интегрирования системы ДАУ в ПА9 применяются неявные А-устойчивые методы интегрирования: метод Эйлера (1-го порядка точности) и метод трапеций (2-го порядка точности). Графический редактор обеспечивает весь необходимый набор функций для формирования эквивалентной схемы моделируемого технического объекта. Результаты моделирования отображаются в виде графиков зависимостей фазовых переменных моделируемого объекта от времени.

Основную часть окна графического редактора (рис. 1.1) занимает поле схемы. Если поле схемы целиком не помещается в окне, то справа и снизу добавляются полосы прокрутки. Поле схемы разбито сеткой на квадратные ячейки.

Все составные части схемы всегда обязательно привязываются к этой сетке и занимают всегда целое число ячеек. Для выполнения основных функций редактирования могут быть использованы мышь и клавиатура.

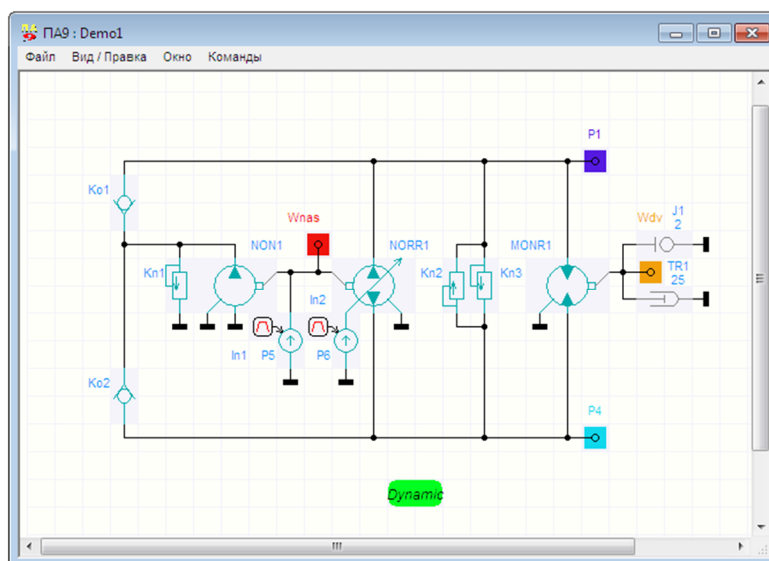


Рис. 1.1 Графический редактор системы ПА9.

Для математического моделирования технических систем различной физической природы в ПА9 используется метод физических аналогий. Согласно этому методу любой технической системе, функционирование которой описывается системой ДАУ, можно поставить в соответствие некоторую формальную эквивалентную схему, которая описывается точно такой же системой ДАУ. Для каждого элемента, который может быть включен в эквивалентную схему, в составе комплекса ПА9 имеется математическая модель в виде подсистемы ДАУ (компонентные уравнения схемы). Помимо набора элементов, входящих в эквивалентную схему, необходимо задать также их взаимосвязи между собой (топологические уравнения схемы), а также параметры элементов (численные значения коэффициентов компонентных уравнений).

Система ПА9 представляет собой кроссплатформенное приложение, разработанное с использованием языка Java. Язык описания моделей в ПА9 основан на родном для системы языке – генерируемый системой код модели конвертируется в код Java, с последующей его компиляцией.

2.1 Библиотека manzhuk

2.1.1 Описание библиотеки

Си программа manzhuk предназначена для решения систем ОДУ-ДАУ в расширенном координатном пространстве переменных, который в качестве зависимых переменных систем ОДУ-ДАУ включает дифференциальные переменные, алгебраические переменные, а также производные дифференциальных переменных. Все переменные рассчитываются с гарантированной (на уровне решения систем линейных алгебраических уравнений) достоверностью и точностью результатов решения и с одинаковой математической и компьютерной точностью для всех переменных на каждом шаге интегрирования. Программа ориентирована в первую очередь на решение жестких систем ОДУ-ДАУ с многопериодным характером решения, которые сводятся к решению плохо обусловленных систем ЛАУ, и может использоваться в качестве математического ядра при математическом и компьютерном моделировании динамических систем, в первую очередь для программного комплекса ПА10.

В программе-решателе систем ДАУ manzhuk реализованы 3 метода интегрирования систем ДАУ:

М1 – А-устойчивый одностадийный неявный метод Эйлера первого порядка точности;

М2 – АL-устойчивый одностадийный неявный метод второго порядка точности из работы;

М3 – АL-устойчивый двухстадийный неявный метод четвертого порядка точности из работы;

Во всех методах на каждом шаге интегрирования оценивается относительная, локальная погрешность интегрирования для каждой дифференциальной переменной (по отношению к максимальному абсолютному

значению каждой дифференциальной переменной на заданном отрезке интегрирования). Вычисленная погрешность сравнивается с заданной погрешностью ϵ_{rs} и если она ее превышает, то шаг интегрирования h уменьшается. Максимальное абсолютное значение каждой дифференциальной переменной определяется в ходе интегрирования автоматически, эти значения можно также задать перед началом интегрирования.

Во всех методах на каждом шаге интегрирования решается система НАУ относительно векторов расширенного пространства переменных X, PX, Y методом Ньютона-Рафсона, для которого необходимо вычислять матрицу Якоби. Матрица Якоби должна вычисляться в специальной подпрограмме пользователя `fct`, наряду с вычислением вектор-функции F . Эта матрица состоит из двух подматриц:

$$RJ = (RJ1, RJ2)$$

где RJ - матрица Якоби размером $(n*nm)$; $nm=n+m$. $RJ1$ - матрица частных производных вектор-функции F по переменным PX размером $(n*m)$, т.е. матрица $RJ1 = [dF/dPX]$. $RJ2$ - матрица частных производных вектор-функции F по переменным z размером $(n*n)$, т.е. матрица $RJ2 = [dF/dZ]$, где Z - вектор переменных системы ОДУ размерностью n , который включает дифференциальные переменные и алгебраические переменные системы (2), т.е. $Z = (X, Y)$.

2.1.2 Работа с библиотекой

Формат вызова `manzhuk`

```
void manzhuk(double z[], double px[], double z1[], double xp1[], double f[],
double rj1[], double rj2[], double t, double t0, double tk, double h, double
hmn, double hmx, double eps, double *tkv, int n, int m, int nm, int ncon,
int *nbad, int *ier, int ip[], void fct(double z[], double px[], double
f[], double rj1[], double rj2[], int n, int m, double t, double h, int ncon,
int *nbad, int ip[]), void out(double z[], double px[], int n, int m, double
t, double t0, double tk, double h, double *tkv, int ncon, int ip[]))
```

, где z – массив переменных системы ДАУ размерности n , в первых m элементах этого массива размещаются дифференциальные переменные X системы ДАУ, в остальных элементах размещаются алгебраические переменные, поэтому всегда должно выполняться условие $(n \geq m)$. Перед началом интегрирования в

первых m элементах этого массива размещаются начальные значения дифференциальных переменных – X_0 .

px – массив производных дифференциальных переменных по времени размерностью m , т.е. $px=dx/dt$.

$z1$ – рабочий массив размерностью n , перед началом интегрирования в первых m элементах этого массива размещаются максимальные абсолютные значения дифференциальных переменных. Если они известны, то их считывают из файла и присваивают соответствующим значениям массива $z1$. Если они неизвестны, то в этот массив пересылаются абсолютные величины начальных значений дифференциальных переменных X_0 , как в вышеприведенном примере.

$px1$ – рабочий массив размерностью m .

f – массив размерностью n для вычисления вектор-функции F системы ДАУ в подпрограмме fct .

$rj1$ – матрица размером $(n*m)$ для вычисления частных производных вектор-функции F по переменным px в подпрограмме fct , т.е. $rj1(1,1) = df(1)/dpx(1)$, $rj1(2,2) = df(2)/dpx(2)$ и т.д., хранится по строкам, т.е. $rj1(1,1) == rj1[1*n+1]$, $rj1(2,2) == rj1[2*n+2]$ и т.д.

$rj2$ – матрица размером $(n*n)$ для вычисления частных производных вектор-функции F по переменным z в подпрограмме fct , т.е. $rj2(1,1) = df(1)/dz(1)$, $rj2(1,2) = df(1)/dz(2)$ и т. д., хранится по строкам, т.е. $rj2(1,1) == rj2[1*n+1]$, $rj2(1,2) == rj2[1*n+2]$ и т.д. В программе предусмотрена возможность численного вычисления элементов этой матрицы. Если перед обращением к программе установить параметр $ier=-1$, то все элементы матрицы $rj2$ будут вычисляться автоматически методом приращений так же, как во всех пакетах и библиотеках математических программ. Однако этот метод вычисления частных производных следует применять только на этапе отладки задачи решения систем ОДУ-ДАУ. Общее правило – все, что можно вычислить аналитически следует вычислять аналитически. Для сложных функциональных зависимостей следует использовать численное вычисление частных производных покомпонентно.

t – текущее время интегрирования.

t_0 – заданное время начала интегрирования.

t_k – заданное время окончания интегрирования.

h – текущий шаг интегрирования.

h_{mn} – заданный минимальный шаг интегрирования.

h_{mx} – заданный максимальный шаг интегрирования.

eps – заданная математическая точность интегрирования, одинаковая для всех переменных расширенного координатного пространства переменных (математическая точность решения системы ДАУ, обычно $1e-3$ - математическая точность интегрирования систем ОДУ, задаваемая по умолчанию в пакете математических программ MATLAB).

tkv – переменная, с помощью которой можно точно табулировать задаваемые пользователем моменты времени в ходе интегрирования. Эта переменная устанавливается в подпрограмме пользователя `out` и обычно используется для табуляции результатов. Выше в задачах `task001` и `task002` был приведен пример табуляции результатов расчета, ниже приведен пример получения значений переменных в целочисленные моменты времени 0, 1, 2, для этого в подпрограмму `out` вставлены соответствующие операторы (обнуляя предварительно вспомогательную переменную `tp` (задача `task02`)): `ar` – рабочий массив размером не менее, чем $(3*n*m+5*n*n+9*n+25*m+3)$, в первых трех элементах этого массива размещаются максимальные (среди всех дифференциальных переменных) текущие значения относительных погрешностей интегрирования для методов M1, M2 и M3 соответственно. В последующих m элементах этого массива размещаются максимальные абсолютные значения дифференциальных переменных.

n – общее число уравнений в системе ДАУ.

m – количество дифференциальных переменных в системе ДАУ.

nm – заданный номер метода интегрирования:

$nm = 1$ – метод M1;

$nm = 2$ – метод M2;

$nm = 3$ – метод M3;

$ncon$ – ключ расчета начальных значений переменных:

$ncon = 0$ – выполняется расчет начальных значений переменных $px0$ и $uy0$ при заданном $x0$ для момента времени $t0$ от нулевых значений $px0$ и $uy0$. Значения $x0$ соответствуют исходным значениям первых m элементов массива z . До начала интегрирования с этим значением ключа $ncon$ выполняется одно обращение к подпрограммам fct и out для выполнения однократных вычислений параметров, которые в ходе интегрирования не изменяются. После выполнения однократных вычислений значение $ncon=1$ устанавливается автоматически.

$ncon = 1$ – расчет начальных значений переменных $px0$ и $uy0$ не производится. Этот признак следует использовать в случаях повторных обращений к программ-решателю $manzhuk$, если надо продолжить интегрирование решаемой системы ДАУ. До этого должно быть хотя бы одно обращение к решателю $manzhuk$ с $ncon=0$ или с $ncon=2$.

При нормальном выходе из решателя $manzhuk$ устанавливается $ncon=1$.

$ncon = 2$ – выполняется расчет начальных значений переменных $px0$ и $uy0$ при заданном $x0$ для момента времени $t0$ от вводимых начальных значений $px0$ и $uy0$. Эти значения либо вводятся из заранее сформированного файла исходных данных, либо присваиваются вычисленным заранее значениям этих переменных. Значения $x0$ соответствуют исходным значениям первых m элементов массива z .

$nbad$ – (принципиально новый, важнейший параметр) - ключ уменьшения шага интегрирования, устанавливаемый в подпрограммах fct вычислений элементов век-тор-функции F и матрицы Якоби RJ системы ДАУ. Перед обращением к подпрограммам fct устанавливается $nbad=0$ на каждой итерации. Если отдельные переменные в подпрограммах fct принимают значения, которые могут привести к останову вычислений (корень из отрицательного числа, превышение допустимого порядка в степенных функциях, то необходимо установить $nbad=1$, тогда итерации будут прерваны, произойдет уменьшение шага

интегрирования до тех пор, пока можно будет продолжать вычисления, либо до значения минимального шага с сообщением о не сходимости итераций. Если в моделях установить $nbad=2$, то произойдет уменьшение шага до учетверенного минимального с дальнейшим продолжением интегрирования с этим шагом (фактически расчет с новыми начальными условиями). $nbad=2$ можно установить только в том случае, если $nbad$ не равно 1. $nbad=2$ следует использовать для идентификации точек разрыва производных в кусочно-нелинейных.

Рассмотрим пример использования параметра $nbad=1$ для программирования функций с ограниченной областью определения аргумента. Вычисление любой вектор-функции системы ДАУ можно свести к вычислению базисных функций одного аргумента типа $f(x)$. Рассмотрим функцию $f(x)$ с ограниченной областью определения x , например $f(x)=\sqrt{x}$ определена только для $x \geq 0$. Поэтому при каждом обращении к базисной функции необходимо проверить значение аргумента x и, если аргумент выходит из области определения функции, то следует установить параметр $nbad=1$, не вычисляя значения функции. Фрагмент программы для функции \sqrt{x} будет:

ier – код ошибки:

$ier = 0$ - нет ошибок;

$ier = 1$ – m отрицательно или больше n ;

$ier = 2$ – hmn отрицательно или больше hmx ;

$ier = 3$ – $t0$ больше tk ;

$ier = 4$ – eps меньше $1e-12$;

$ier = 5$ – nm меньше 0, или больше 3;

$ier = 6$ – $ncon$ меньше 0, или больше 2;

$ier = 7$ – нет сходимости итераций при решении нелинейных алгебраических уравнений на минимальном шаге интегрирования $hmin$. Следует проверить правильность формул для вычисления элементов матриц $rj1$ и $rj2$, уменьшить заданный минимальный шаг интегрирования и проверить не расходится ли система ДАУ.

ier = 8 – Вырожденность системы линейных алгебраических уравнений при выполнении Ньютоновских итераций.

ier=-1 – Это значение параметра используется для установки ключа, обеспечивающего численное вычисление элементов матрицы $rj2$.

ip – рабочий массив размером не менее, чем $(20+2*n+6*m)$, в первых 20-ти элементах этого массива размещаются специальные счетчики и ключи для методов интегрирования:

ip(1) – номер итерации (начиная с 0) при решении системы нелинейных алгебраических уравнений на текущем шаге интегрирования (не более 10-ти).

ip (2) – суммарное количество итераций при решении нелинейных алгебраических уравнений на всех выполненных и аннулированных шагах интегрирования.

ip (3) – суммарное количество выполненных шагов интегрирования.

ip (4) – суммарное количество делений шага интегрирования с отменой выполненного предыдущего шага, т.е. суммарное количество фактически аннулированных шагов.

Условно аннулированными называются шаги $h = h_{min}$, которые выполняются несмотря на то, что текущая погрешность превышает заданную при $nbad=2$.

ip (5) – суммарное количество аннулированных шагов из-за превышения заданной погрешности eps , включая условно аннулированные.

ip (6) – суммарное количество аннулированных шагов из-за не сходимости итераций при решении алгебраических уравнений на каждом шаге интегрирования, включая условно аннулированные.

ip (7) – суммарное количество выполненных итераций по методу M2.

ip (8) – суммарное количество выполненных шагов интегрирования по методу M2.

ip (9) – суммарное количество выполненных итераций по методу M3.

ip (10) – суммарное количество выполненных шагов интегрирования по методу МЗ.

ip (11) – плохая обусловленность системы.

ip (12) – количество коррекций производных для методов 2 и 4 порядка точности.

ip (13) – уменьшение шага из-за prwar и asr.

ip (14) – уменьшение шага из-за nbad=1.

ip (15) - уменьшение шага из-за превышения числа итераций больше 10.

ip (16) -уменьшение шага из-за того, что невязка и приращения не уменьшаются.

fct – имя внешней подпрограммы пользователя. В этой подпрограмме пользователь вычисляет вектор-функцию F решаемой системы ДАУ и матрицы g_{j1} и g_{j2} частных производных df/dp_x и df/dz . Нулевые значения элементов матриц g_{j1} и g_{j2} в подпрограмме fct присваивать не нужно. К подпрограмме fct происходит обращение на каждой итерации. При первом обращении к решателю (ключ ncon=0 или ncon=2) организовано одно обращение к подпрограмме fct при $t=t_0$ и $h=h_{min}$ для выполнения однократных вычислений параметров, не изменяющихся в ходе интегрирования. После выполнения однократных вычислений устанавливается ключ ncon=1. При каждом обращении в массив f заносится рекомендуемый вектор приращений dz для переменных z с целью выполнения численного вычисления частных производных $df(i)/dz(j) = (f_i(z_j+dz_j)-f_i(dz_j))/dz_j$; Эту возможность следует использовать только при невозможности аналитического вычисления частных производных.

Задаваемые параметры интегрирования

Перед обращением к программе должны быть заданы значения следующих параметров, ключей и переменных:

n – общее число уравнений в системе ДАУ;

m – количество дифференциальных переменных в системе ДАУ;

t0 – заданное время начала интегрирования;

tk – заданное время окончания интегрирования;
 hmn – заданный минимальный шаг интегрирования;
 hmx – заданный максимальный шаг интегрирования;
 eps – заданная относительная погрешность интегрирования, одинаковая для всех дифференцируемых переменных;
 nm – заданный номер метода интегрирования;
 $ncon$ – ключ расчета начальных значений переменных;
 $ier = -1$ – ключ для установки автоматического численного вычисления элементов матрицы g_j .
 $z(1)...z(m)$ - начальные значения дифференциальных переменных.
 $z1(1)...z1(m)$ - либо максимальные абсолютные значения дифференциальных переменных, либо $z1(i)=abs(z(i))$.

2.1.1 Функция **fct**

В простейшем случае функция **fct** формируется пользователем отдельно для каждой новой системы и компилируется вместе с остальными файлами расчетной программы. Были рассмотрены два варианта обеспечения возможности решения множества различных систем без перезапуска.

Первый предполагает реализацию универсальной функции, использующей некоторую структуру данных, например вычисление значений может производиться с помощью деревьев математических выражений или стековой машины для работы с обратной польской записью. При этом значительные расходы влечет обработка используемой структуры данных, что крайне нежелательно для функции, вызываемой на каждом шаге интегрирования.

Во втором варианте осуществляется следующая последовательность действий:

- а. Генерация файла с исходным кодом функции **fct**;
- б. Вызов компилятора для получения динамически подключаемой библиотеки (DLL или SO, в случае с Unix);

- в. Динамическое подключение полученной библиотеки к выполняемой программе;
- г. Запуск расчета с использованием новой функции fct;
- д. Выгрузка библиотеки из оперативной памяти и удаление файлов библиотеки.

Был выбран второй подход, как обеспечивающий большую производительность.

Поскольку разработка ведется в среде программирования Visual Studio 2008, в качестве вызываемого компилятора используется Microsoft Visual C++.

Команда вызова имеет вид:

```
«cl /Ot /I"..." /Tp "manzhuk/fcttask/fcttask.cpp" libcpmt.lib libcmnt.lib  
oldnames.lib Kernel32.lib /link /LIBPATH:"..." /LIBPATH:"manzhuk\fcttask" /DLL  
/out:"manzhuk/fcttask/fcttask.dll"»
```

2.2 Библиотека Qt

Библиотека Qt задумывалась и начиналась как кроссплатформенное средство для быстрой разработки графических интерфейсов (GUI) приложений на языке C++, с целью упростить жизнь программистов, пишущих на C++ кроссплатформенные, переносимые GUI-приложения, которые должны работать и в среде Windows, и в среде Unix/Linux под X11, и на компьютерах Macintosh.

В настоящее время Qt значительно переросла рамки средства разработки графических интерфейсов приложений. Она предоставляет использующему её программисту целостный фреймворк (framework), позволяющий при написании большей части приложения использовать только «родные» классы Qt и практически полностью отказаться от написания системно-зависимого кода и использования системных вызовов. Классы Qt покрывают почти все потребности программиста. В Qt предусмотрены классы и для работы со строками, и для работы с файлами, сетью, базами данных, XML, и для обеспечения многопоточности в приложении, и многое-многое другое. По своим возможностям и богатству библиотека Qt сравнима с .NET Framework или с системой классов Java.

Qt предоставляет программисту не только удобный набор библиотек классов, но и определённую модель разработки приложений, определённый каркас их структуры. Следование принципам и правилам «хорошего стиля программирования на C++/Qt» существенно снижает частоту таких трудно отлавливаемых ошибок в приложениях, как утечки памяти (memory leaks), необработанные исключения, незакрытые файлы или неосвобождённые дескрипторы ресурсных объектов, чем нередко отличаются программы, написанные на чистом C++ без использования библиотеки Qt.

Важным преимуществом Qt является хорошо продуманный, логичный и стройный набор классов, предоставляющий программисту очень высокий уровень абстракции. Благодаря этому программистам, использующим Qt, приходится писать значительно меньше кода. Сам же код выглядит стройнее и проще, логичнее и понятнее. Его легче поддерживать и развивать.

Кроме того, даже если программисту в данный конкретный момент не нужна кроссплатформенность для его конкретного приложения (например, планируется версия только для Windows или только для Macintosh), никто не может знать, что понадобится завтра. Бизнес-планы могут поменяться, и может оказаться и нужным, и выгодным выпустить версию для другой операционной системы или другой аппаратной платформы. В случае использования Qt для этого понадобится всего лишь перекомпиляция исходного кода. В случае же использования «родных» системных API понадобится много тяжёлой работы по портированию, адаптации и отладке, а возможно, переписыванию с нуля существующего исходного кода для другой ОС.

Многие компании-разработчики приложений Windows используют Qt ещё по одной причине: даже если код пишется и в обозримом будущем будет писаться только для платформы Windows и тестируется только на ней, возможность откомпилировать один и тот же исходный код на одной и той же платформе Windows двумя разными компиляторами (Microsoft Visual C++ и GCC/Win32) гарантирует лучшее качество исходного кода и лучшую его совместимость со

стандартом C++. Что немаловажно для кода, который планируется длительно поддерживать и развивать.

2.3 Механизм слотов и сигналов

Сигналы и слоты в библиотеке Qt используются для коммуникации между объектами. Механизм сигналов и слотов – главная особенность Qt и, вероятно, та часть, которая отличается от особенностей, предоставляемых другими фреймворками.

В программировании графического интерфейса, когда изменяется один виджет, часто требуется чтобы другой виджет получил об этом уведомление. В общем случае нужно, чтобы объекты любого типа могла общаться с другими. Например, если пользователь нажимает кнопку «Заккрыть», вероятно, необходимо чтобы была вызвана функция окна `close()`. Другие библиотеки добиваются такого рода общения используя «обратный вызов». «Обратный вызов» – это указатель на функцию, таким образом, если нужно чтобы функция уведомила программу о каких-нибудь событиях, мы передаем указатель на другую функцию обратно вызываемую этой функцией. Функция в таком случае делает обратный вызов когда необходимо. Обратный вызов имеет два основных недостатка. Во-первых, он не является типобезопасным. Никогда нельзя быть уверенным, что функция делает обратный вызов с корректными аргументами. Во-вторых, обратный вызов жестко связан с вызывающей его функцией, так как эта функция должна точно знать какой обратный вызов надо делать.

В Qt используется другая техника – сигналы и слоты. Сигнал вырабатывается когда происходит определенное событие. Слот – это функция, которая вызывается в ответ на определенный сигнал. Виджеты Qt имеют много предопределенных сигналов и слотов, но мы всегда можем сделать дочерний класс и добавить наши сигналы и слоты в нем. Механизм сигналов и слотов типобезопасен. Сигнатура сигнала должна совпадать с сигнатурой слота-получателя. (Фактически слот может иметь более короткую сигнатуру чем сигнал

который он получает, так как он может игнорировать дополнительные аргументы). Так как сигнатуры сравнимы, компилятор может помочь нам обнаружить несовпадение типов. Сигналы и слоты слабо связаны. Класс, который вырабатывает сигнал не знает и не заботится о том, какие слоты его получают. Механизм сигналов и слотов Qt гарантирует, что если мы подключим сигнал к слоту, слот будет вызван с параметрами сигнала в нужное время. Сигналы и слоты могут принимать любое число аргументов любого типа. Они полностью типобезопасны. Все классы, наследуемые от QObject или его дочерних классов (например, QWidget) могут содержать сигналы и слоты. Сигналы вырабатываются объектами когда они изменяют свое состояние так, что это может заинтересовать другие объекты. При этом он не знает и не заботится о том, что у его сигнала может не быть получателя. Слоты могут быть использованы для получения сигналов, но они так же нормальные функции-члены. Так же как объект не знает ничего о получателях своих сигналов, слот ничего не знает о сигналах, которые к нему подключены. Это гарантирует что полностью независимые компоненты могут быть созданы с помощью Qt. Мы можем подключать к одному слоту столько сигналов, сколько захотим, также один сигнал может быть подключен к стольким слотам, сколько необходимо. Так же возможно подключать сигнал к другому сигналу (это вызовет выработку второго сигнала немедленно после появления первого). Сигналы и слоты вместе составляют мощный механизм создания компонентов.

Сигналы вырабатываются объектами когда они изменяют свое состояние так, что это может заинтересовать другие объекты. Только класс, который определяет сигнал или его потомки могут вырабатывать сигнал. Когда сигнал вырабатывается, слот, к которому он подключен обычно выполняется немедленно, так же как и нормальный вызов процедуры. Когда это происходит, механизм сигналов и слотов полностью независим от любого цикла событий графического интерфейса. Выполнение кода, следующего за выпуском сигнала произойдет сразу после выхода из всех слотов. Ситуация слегка отличается когда используются

отложенные соединения (queued connections); в этом случае код после ключевого слова `emit` продолжает выполнение немедленно, а слоты будут выполнены позже. Если несколько слотов подключены к одному сигналу, слоты будут выполнены один за другим в произвольном порядке после выработки сигнала. Сигналы автоматически генерируются программой `QtCore` и не должны быть реализованы в исходном коде. Они могут не возвращать значение (т. е., используем тип `void`).

Слот вызывается когда вырабатывается сигнал, с которым он связан. Слот – это обычная функция в C++ и может вызываться обычным способом; единственная его особенность, что с ним можно соединять сигналы. Так как слоты – это нормальные функции-члены, они следуют обычным правилам C++ при прямом вызове. Тем не менее, как слоты, они могут быть вызваны любым компонентом, независимо от их уровней доступа, через соединение сигнал-слот. Это значит, что сигнал, выработанный объектом произвольного класса может вызвать защищенный (`private`) слот объекта несвязанного с ним класса. Слоты так же можно объявлять виртуальными, что иногда бывает довольно удобно. По сравнению с обратными вызовами, сигналы и слоты слегка медленнее из-за увеличенной гибкости, которую они обеспечивают, хотя разница для реальных приложений незаметна. В общем, выработка сигнала, который подключен к некоторым слотам, в среднем в 10 раз медленнее, чем вызов получателя напрямую, при вызове не виртуальной функции. Эти накладные расходы требуются для нахождения объекта, для безопасного перебора всех его соединений (т. е. проверка что последующий получатель не был уничтожен во время выпуска сигнала) и передачи любых параметров в общем виде. Хотя вызов десяти неvirtуальных процедур может показаться дорогим, это менее затратно, чем, например, операция создания или удаления объекта.

Глава 3 Технологическая часть

3.1 Вычислительный модуль

3.1.1 Представление системы уравнений

Для хранения системы уравнений в программе используется класс DaeSystem.

Класс DaeSystem хранит следующие данные:

- а. список уравнений,
- б. список констант,
- в. список переменных системы.

Каждая переменная обладает уникальным числовым идентификатором, именем, флагом наличия в системе производной этой переменной по времени, полем значения переменной в начале интегрирования, по умолчанию равным нулю. Числовой идентификатор производной дифференциальной переменной равен ее идентификатору по модулю, но отличается знаком. Это позволяет легко соотнести переменную с ее производной, если она существует, а также убирает необходимость отдельного хранения ее производной.

Во время разбора текстового представления системы используется список констант, после завершения разбора список более нигде не участвует. Память, выделяемая под список очищается.

3.1.2 Представление выражений

Математические выражения в программе используются в качестве промежуточных данных между заданной пользователем в том или ином виде системой уравнений и генерацией функции fct решателя manzhuk.

На описанном этапе каждое выражение представлено в виде бинарных дерева. В узлах такого дерева находятся математические операторы и функции,

листья представляют переменные и константы. Так, например, для выражения $2 \cdot x + 3$ дерево будет иметь вид:

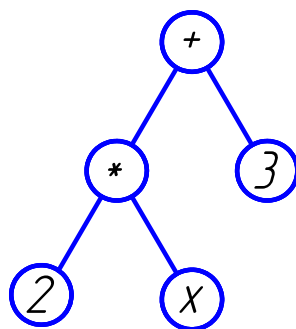


Рис. 3.1 Пример представления математического выражения в виде бинарного дерева

3.1.3 Получение Якобиана

При составлении функции `fct` пользователь должен рассчитать матрицу Якоби – матрицу частных производных вектор функции F по производным дифференциальных переменных системы. Чтобы облегчить пользователю задачу, в класс `Expression` добавлены функции `derivative` и `simplify`.

Функция `derivative` служит для символьного дифференцирования и возвращает указатель на производную того выражения, к которому применена. Вызов функции едет к созданию копии дерева. Затем к копии рекурсивно согласно обратному обходу дерева применяются операции взятия производной, согласно правилам дифференцирования.

Результирующее выражение значительно превосходит по размеру исходное и содержит большое количество констант, зачастую нулей и единиц. Для сокращения полученного выражения используется функция `simplify`, которая аналогичным проходит по элементам, вычисляя константные выражения, упрощая выражения, содержащие нули и единицы.

Таких простых мер достаточно, чтобы получить удовлетворительный результат. Стоит отметить, что для работы с большими и сложными выражениями,

функции упрощения требуется доработка, так как лишние операции могут замедлить процесс интегрирования системы.

3.1.4 Взаимодействие с решателем

Применение библиотеки `manzhuk` ведет к неизбежному использованию глобальных переменных. Чтобы свести количество таких переменных к минимуму используется один глобальный объект класса `Solver`. При этом во время вызова функции `out` производится вызов функции-члена глобального объекта класса `Solver`.

Класс `Solver` организует

- а. управление выделяемой для решателя памятью,
- б. операции, связанные с использованием функции `fct` – компиляцию, подключение, освобождение памяти, удаление файлов,
- в. вызов функции решателя,
- г. управление выходными данными расчета.

3.2 Модуль ввода

3.2.1 Генератор лексических анализаторов Flex

Генератор лексических анализаторов Flex использует конфигурационный файл для создания исходного кода на C/C++, из которого можно затем создать либо отдельное приложение, либо встроить этот исходный код в другое приложение. Конфигурационный файл определяет набор символов, ожидающихся в файле, который будет анализироваться, и какие действия надо выполнить, когда файл будет проанализирован. Формат файла является открытым; требуется только распознать лексемы во входном файле и определить, что нужно сделать при их обнаружении.

Правила задаются в виде регулярных выражений слева и, обычно, кода на языке C справа. Правила содержат три секции, отделяющиеся строкой «%%»:

Определения.

%%

Правила.

%%

Код пользователя.

Использование связки генератора лексических анализаторов Flex и синтаксических анализаторов Bison позволяет получить качественный и быстрый анализатор текста.

При этом сокращается время на разработку анализатора, но, самое главное, на его модификацию. Если в процессе разработки основной программы потребуется внести какие-либо изменения в правила оформления входных данных, время, потраченное на изменение анализатора текста будет сведено к минимуму.

Свойства генерируемого лексического: анализатора задаются секции определений. Используемые опции:

%option c++ – генерация объектно-ориентированного кода.

%option batch – оптимизация генерируемого кода.

%option yylineno – учет номера разбираемой строки.

%option yyclass = "Scanner" – указание названия класса лексического анализатора.

В потоке символов лексический анализатор находит лексемы и выполняет заданные для них действия. Для разбора текста, содержащего систему дифференциальных уравнений были выделены следующие типы лексем:

- а. числовые данные
- б. имена переменных
- в. имена математических функций
- г. символ дифференцирования
- д. математические операторы
- е. специальные символы, пробел и символ табуляции

3.2.2 Генератор синтаксических анализаторов Bison

Bison – это генератор лексических анализаторов общего назначения, который преобразует описание контекстно-свободной LALR(1) грамматики в программу на языке C/C++ для разбора этой грамматики.

Для того, чтобы Bison мог разобрать программу на каком-то языке, этот язык должен быть описан контекстно-свободной грамматикой. Это означает, что нужно определить одну или более синтаксических групп и задать правила их сборки из составных частей.

Наиболее распространённой формальной системой для представления таких правил в удобном для человека виде является форма Бэкуса-Наура (БНФ, Backus-Naur Form, BNF). Любая грамматика, выраженная в форме Бэкуса-Наура является контекстно-свободной грамматикой. Bison принимает на вход, в сущности, особый вид БНФ, адаптированный для машинной обработки.

БНФ имеет вид:

<определяемый символ> ::= <посл.1> | <посл.2> | . . . | <посл.n>.

Такое правило означает, что символ <определяемый символ> может заменяться на одну из последовательностей <посл.i>. Знак определения обычно выглядит как ::= но возможны и другие варианты, так в используемой версии Bison используется знак ⇐.

Bison может работать не со всеми контекстно-свободными грамматиками, а только с грамматиками класса LALR(1). Коротко, это означает, что должно быть возможно определить, как разобрать любую часть входа, заглядывая вперёд не более, чем на одну лексему. Строго говоря, это описание LR(1)-грамматики, класс LALR(1) имеет дополнительные ограничения, но в обычной практике редко встречаются LR(1)-грамматики, которые не являются LALR(1).

Правила синтаксического анализатора составлены в соответствии с правилами составления математических выражений. Правила связаны друг с другом так, что при разборе естественным путем образуется бинарное дерево математического выражения. При нахождении оператора в дерево добавляется новое дерево, корнем которого является оператор, а поддеревьями аргументы этого оператора. Аналогичным путем производится обработка функций.

При нахождении в потоке символов переменной, информация о переменной добавляется в систему уравнений.

Также представлены три правила, разделяющие документ на составляющие части: область констант, область уравнений и область начальных значений.

При несоответствии потока данных заданным правилам анализатор останавливает свою работу и вызывает функцию обработки ошибок.

3.2.3 Ввод системы уравнений

Большинство математических пакетов требуют приведения вводимой системы уравнений к нормальной форме Коши. Разработанная программа принимает для расчета систему в любой форме записи, отвечающий требованиям:

- а. Все константы описаны до начала описания уравнений;
- б. Для присвоения значения константе используется знак $:=$;
- в. Имена констант и переменных начинаются с буквы и не содержат специальных символов;
- г. Используются только математические функции, известные программе;
- д. Уравнения системы описываются после окончания описания констант;
- е. Для присвоения начального значения переменной используется знак $==$;
- ж. Секции описания констант и начальных значений не являются обязательными;
- з. Каждое новое определение начинается с новой строки.

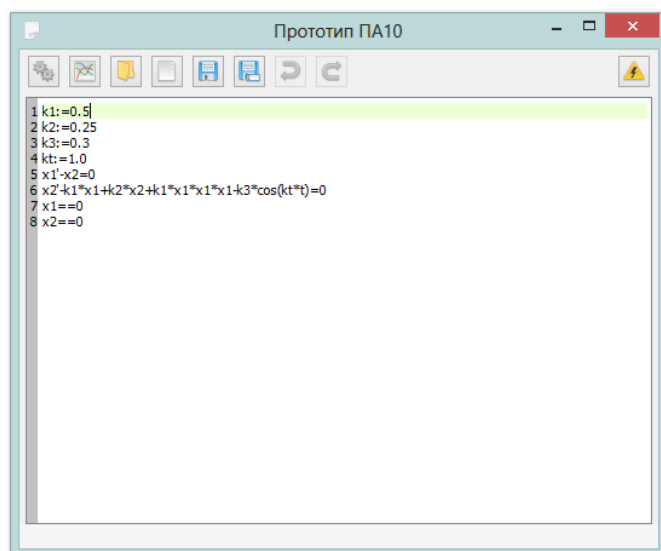


Рис. 3.2 Окно программы в режиме ввода/редактирования системы уравнений

В режиме текстового редактора предусмотрены следующие возможности работы с файлами:

- а. открытие созданного ранее файла,
- б. создание нового файла,
- в. сохранение.

Используются файлы, имеющие расширение «.txt».

3.2.4 Начало вычислений

Перед началом вычислений необходимо ввести параметры интегрирования. Для решения большинства задач достаточно задать начальное и конечное время отрезка интегрирования. Изначально выставлены параметры по умолчанию. Для некоторых специфических задач может потребоваться изменение шага. В случае неверного ввода программа не допустит запуска вычислений.

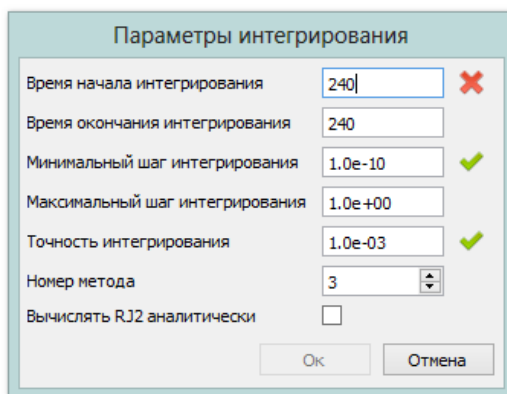


Рис. 3.3 Ввод неверных параметров интегрирования

Так время начала интегрирования не должно превышать время окончания, аналогичные условия накладываются на минимальное и максимальное значения шага. Максимальный шаг не может превышать длину отрезка интегрирования.

Параметры интегрирования

Время начала интегрирования	0	✓
Время окончания интегрирования	240	
Минимальный шаг интегрирования	1.0e-10	✓
Максимальный шаг интегрирования	1.0e+00	
Точность интегрирования	1.0e-03	✓
Номер метода	3	
Вычислять RJ2 аналитически	<input type="checkbox"/>	

Ok Отмена

Рис. 3.4 Ввод верных параметров интегрирования

После ввода желаемых параметров открывается окно выбора рассчитываемых переменных. Выбранные переменные будут выведены в файл на каждом шаге расчета и сохранены в памяти для вывода на график. В поле, расположенном рядом с именем переменных может быть введено альтернативное имя переменной, которое будет отображаться в легенде рядом с графиками.

Вывести:

☐ Select all

<input checked="" type="checkbox"/> x1	x1
<input type="checkbox"/> x2	x2
<input checked="" type="checkbox"/> x1'	dx1/dt
<input type="checkbox"/> x2'	dx2/dt

Ok

Рис. 3.5 Диалог выбора отображаемых переменных

3.2.5 Схемный редактор

Структура классов, относящихся к схемному режиму максимально упрощена – графическая составляющая объединена с математической.

Реализованы все типы элементов, описанных в главе 1 базовые двухполюсники:

- а. R_CircuitItem – резистор,
- б. C_CircuitItem – конденсатор,
- в. I_CircuitItem – источник тока,
- г. E_CircuitItem – источник напряжения,
- д. L_CircuitItem – индуктивность,
- е. G_CircuitItem – проводимость.

Каждый из них наследуется от общего родительского класса CircuitItem, который, в свою очередь, имеет предком класс библиотеки Qt QGraphicsItem.

Класс CircuitItem совмещает математическую модель двухполюсника с его графическим представлением. Уравнения, соответствующие типу элемента, а также имена и размерности параметров назначаются в классах-потомках. Также в классах-потомках определены функции рисования и функции геометрической формы элемента (для обозначения интерактивной области).

Объекты типа QGraphicsItem располагаются сцене QGraphicsScene, отображаемой в специальном виджете QGraphicsView.

Наследуемый от QGraphicsView класс SchemeView отвечает за взаимодействие пользователя с элементами (их перемещение, изменение), создание вызываемых пользователем меню и диалоговых окон. Кроме того, содержит функции компоновки модели, определение узлов при соединении элементов, редактирование элементов, нумерацию, создание и удаление.

Добавить новый элемент на схему можно, открыв меню добавления узла, которое вызывается щелчком правой кнопки мыши на свободных клетках сетки.

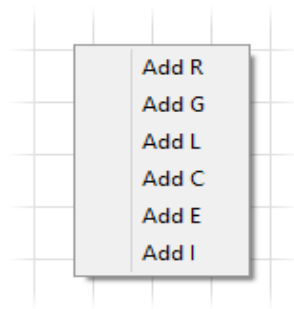


Рис. 3.6 Меню выбора элемента для добавления его на схему

Элемент – интерактивный объект, который можно свободно перемещать вдоль линий сетки. Повернуть элемент можно, выбрав соответствующий пункт в меню элемента, которое по щелчку правой кнопкой мыши будет вызвано на выбранном элементе.

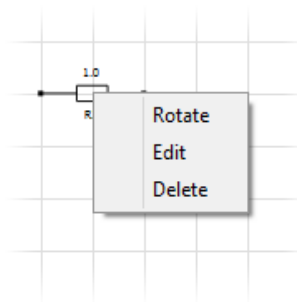


Рис. 3.7 Меню элемента схемы.

Для изменения параметров элемента, таких как имя элемента, его уравнение, начальные значения тока и напряжения, требуется выбрать пункт Edit.

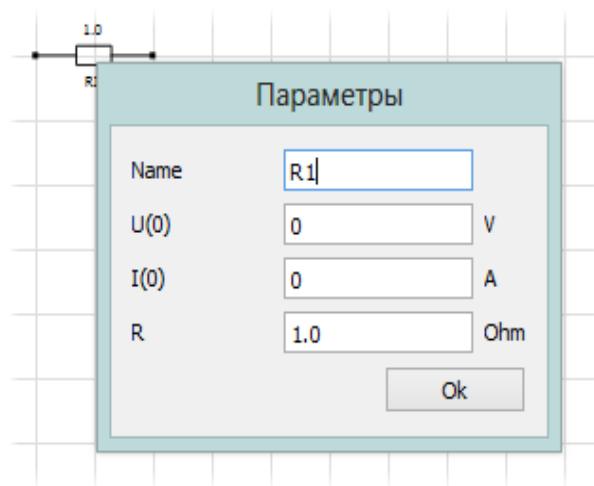


Рис. 3.8 Окно редактирования параметров элемента

Для соединения двух элементов требуется захватить левой кнопкой мыши один из контактов элемента и поместить его в нужную позицию сетки. Элементы считаются соединенными, если соединены их контакты.

После каждого геометрического изменения схемы выполняется проверка списка узлов, удаление разомкнутых и создание новых узлов.

После получения желаемой схемы необходимо установить нулевой потенциал на одном из узлов, для этого нужно с помощью щелчка правой кнопкой мыши на выбранном узле и в открывшемся меню выбрать Ground. Пункт меню Unground служит для отмены этого действия.

3.2.6 Моделирование схемы

Для упрощения манипуляций с данными уравнения элементов имеют тип `QString`, по свойствам аналогичного типу `std::string`, но имеющего большее количество методов. Функция `getSystem` класса `SchemeView` генерирует по схеме систему уравнений в обобщенном базисе переменных. Система уравнений формируется в текстовом виде.

От каждого элемента схемы функция `getSystem` получает его уравнение, в котором имена переменных обозначающих токи и напряжения зависят от имени элемента. Так для элемента `R1` переменная типа потока будет иметь имя `iR1`, а переменная типа потенциала `uR1`.

Система дополняется уравнениями падения напряжения на элементах схемы и суммами токов в узлах.

Можно просмотреть полученную систему уравнений, экспортировав ее в текстовый редактор, либо сразу вызвать анализатор текста для ее разбора.

Шаги, необходимые для начала расчета выполняются тем же образом, что и в текстовом режиме.

3.3 Qwt. Отображение данных

Qwt (Qt Widgets for Technical Applications) — набор Qt-виджетов и вспомогательных классов для программирования приложений, имеющих техническую направленность, в основном, необходимых для создания графического представления числовых данных. Помимо виджета для двумерного отображения данных (QwtPlot) он включает в себя классы для отображения данных в разных масштабах осей, различные стили отображения кривых и маркеров на виджете QwtPlot, а также некоторые другие вспомогательные виджеты.

Данные для рисования кривых могут иметь различные типы, в основном используются контейнеры Qt.

При написании виджеты вывода графиков использовался класс QwtPlot с небольшими изменениями в области приема событий. Виджет в состоянии без значительных задержек отобразить достаточно большие объемы данных.

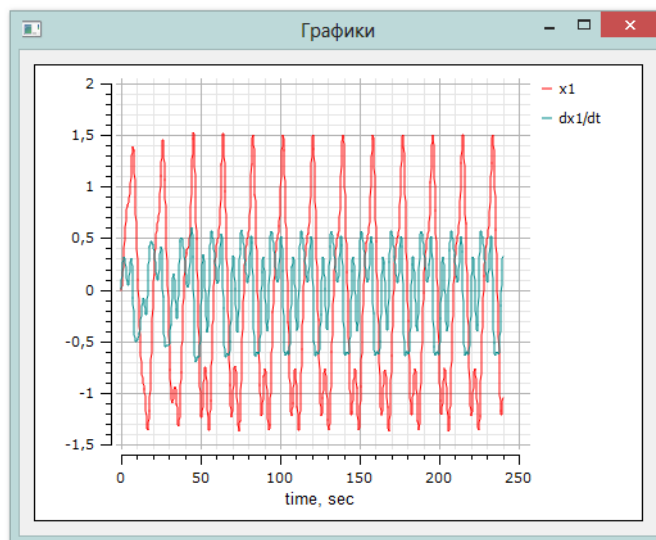


Рис. 3.9 Пример графика. Решение системы ОДУ осциллятора Дуффинга

3.4 Тесты

3.4.1 Тест Ван дер Поля

Пример расчета жесткой системы ОДУ 2-го порядка (MU – параметр жесткости) – тест Ван дер Поля.

$$\begin{aligned} dx_1 / dt &= x_2 \\ dx_2 / dt &= -x_1 + MU \cdot (1 - x_1^2) \cdot x_2 \end{aligned}$$

Начальные условия:

$$x_1(0) = -1,$$

$$x_2(0) = 1$$

$$t \in (0, 8.4 \cdot MU)$$

На практике встречаются значения жесткости $MU = 10^6$ и $MU = 10^9$, ниже приведено решение для значения жесткости $MU = 10^6$.

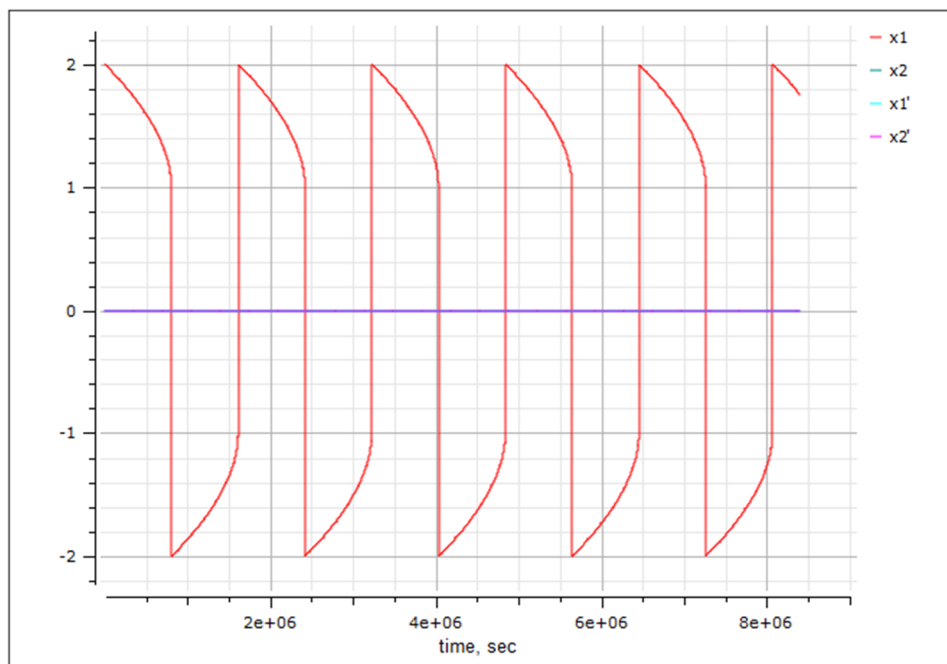


Рис. 3.10 Результаты решения системы уравнений Ван дер Поля методом МЗ

Большинство математических программ, в том числе программная система ПА9 дает неверное решение для данной задачи.

3.4.2 RLC-цепь

Пример моделирования электронной схемы с многопериодным решением.

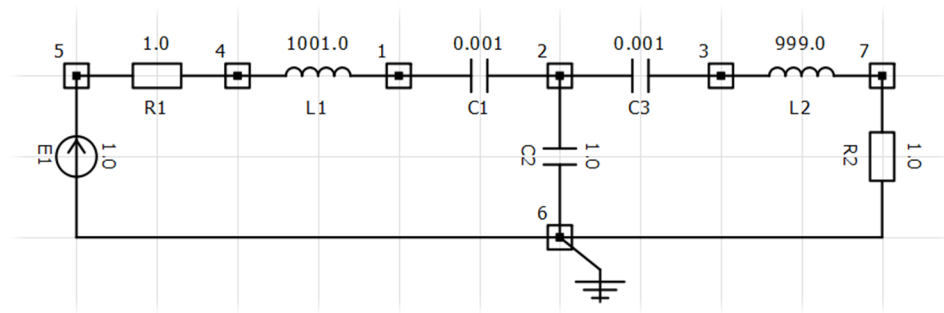


Рис. 3.11 Электронная схема, созданная в разработанной программе

Система ОДУ сгенерированная программой по схеме имеет вид:

$$\begin{aligned}
 u_{R1} &= i_{R1} * 1.0 \\
 u_{R1} &= -\phi_1 + \phi_5 \\
 u_{L1} &= i_{L1}' * 1001.0 \\
 u_{L1} &= \phi_1 - \phi_2 \\
 i_{C1} &= u_{C1}' * 0.001 \\
 u_{C1} &= \phi_2 - \phi_3 \\
 i_{C2} &= u_{C2}' * 1.0 \\
 u_{C2} &= \phi_3 \\
 i_{C3} &= u_{C3}' * 0.001 \\
 u_{C3} &= \phi_3 - \phi_4 \\
 u_{L2} &= i_{L2}' * 999.0 \\
 u_{L2} &= \phi_4 - \phi_7 \\
 u_{R2} &= i_{R2} * 1.0 \\
 u_{R2} &= \phi_7 \\
 u_{E1} &= 1.0 \\
 u_{E1} &= \phi_5 \\
 -i_{R1} + i_{L1} &= 0
 \end{aligned}$$

$$-iL1+iC1=0$$

$$-iC1+iC2+iC3=0$$

$$-iC3+iL2=0$$

$$iR1+iE1=0$$

$$-iL2+iR2=0$$

Результаты расчета – выходное значение $U_{\text{вых}} (\phi_7)$:

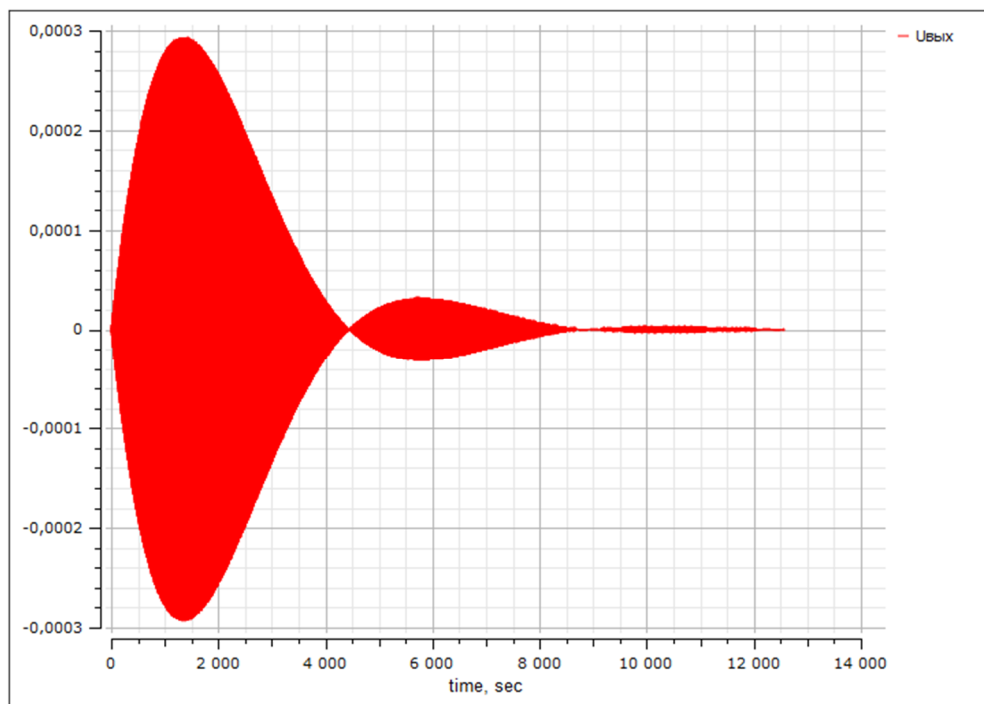


Рис. 3.12 График изменения $U_{\text{вых}} (\phi_7)$ во времени

Стоит отметить, что решение получено при значениях параметров интегрирования по умолчанию, то есть был задан лишь отрезок моделирования.

К примеру, программный комплекс Multisim 7.0 дает неверное решение этой задачи, верное решение может быть получено только после настройки и подбора множества параметров интегрирования в сочетании с выбором определенного метода интегрирования. [Equation Chapter \(Next\) Section 1](#)

4.1 Введение

Дипломный проект посвящен разработке и реализации прототипа программного комплекса (ПК) ПА10, представляющего собой систему моделирования динамических процессов в технических объектах, предназначенную для анализа процессов в различных физических системах.

В процессе проектирования инженеру часто приходится прибегать к перебору и сравнению множества различных вариантов проектируемой системы в условиях жестких временных ограничений. Поэтому простота и скорость реализации отдельного варианта при использовании системы моделирования существенно влияет на качество проектирования. Применение систем моделирования позволяет уменьшить количество ошибок в процессе проектирования, а также сократить временные и денежные затраты за счет использования виртуальных моделей.

Организационно-экономическая часть работы посвящена разработке комплекса мероприятий организационно–экономического и финансового планов, которые необходимо выполнить для разработки прототипа ПК ПА10. Проект изначально не преследует цели получения прибыли.

Расчет производится по методике Ю.В. Сажина.

4.2 Основные этапы проекта разработки программного продукта

Разработка программного продукта (ПП) разбивается на следующие этапы (стадии): техническое задание, эскизный проект, технический проект, рабочий проект, внедрение.

В таблице 3 можно посмотреть содержание каждого из этапов.

Таблица 3. Основные этапы разработки ПП

№	Этап разработки	Содержание работ
1	Техническое задание (ТЗ)	Предпроектное исследование предыдущих программ серии ПА и нового математического ядра. Постановка задачи, выбор основных требований. Расчёт технико-экономического обоснования разработки. Выбор технологии программирования. Разработка календарного плана выполнения работ
2	Эскизный проект (ЭП)	Разработка структуры программного продукта. Формирование структуры и формы представления входных и выходных данных, а также внутреннего представления математических объектов. Разработка общих алгоритмов решения задач. Выработка плана внедрения системы. Создание пояснительной записки в соответствии с ГОСТ. Согласование и утверждение эскизного проекта.
3	Технический проект (ТП)	Разработка конкретных алгоритмов решения задач, структуры программы, пояснительной записки. Выбор среды программирования, конфигурации технических средств. Создание программной документации в соответствии с ГОСТ. Согласование и утверждение технического проекта.
4	Рабочий проект (РП)	Реализация модуля на языке программирования в интегрированной среде Visual Studio 2008. Разработка программной документации и методики испытаний. Проведение всех видов испытаний. Отладка программы. Сдача проекта в опытную эксплуатацию.
5	Документация и внедрение (В)	Подготовка и передача программы и программной документации для оформления и утверждения акта о передаче. Передача программного продукта заказчику. Настройка и внедрение.

4.3 Расчет трудоемкости разработки программного продукта

Одним из основных затратных показателей являются совокупные затраты на оплату труда исполнителей. Расчёт трудоёмкости является основополагающим

для определения общих (совокупных) затрат на реализацию проекта, поэтому расчёту трудоёмкости уделено особое внимание.

Трудоёмкость разработки программной продукции зависит от степени новизны разработки, сложности алгоритма ее функционирования, объема используемой информации и вида ее обработки, уровня используемого алгоритмического языка программирования.

По степени новизны разрабатываемая программная продукция может быть отнесена к одной из четырех групп таблице 4.

Таблица 4. Классификация программных продуктов по степени новизны

Название группы	Описание
А	Разработка программных комплексов, требующих использования принципиально новых методов их создания, проведение НИРС и т.п.
Б	Разработка программной продукции, не имеющей аналогов, в том числе разработка пакетов прикладных программ.
В	Разработка программной продукции, имеющей аналоги.
Г	Разработка программной продукции, основанной на привязке типовых проектных решений.

В нашем случае разрабатываемая программа относится к группе «В».

Трудоёмкость оценивают на основе известной трудоёмкости разработки аналогичного ПО с учетом отличительных особенностей данного проекта, отражаемых введением поправочных коэффициентов. Этот подход предполагает наличие специального фонда алгоритмов и программ–аналогов.

Трудоёмкость разработки программной продукции $\tau_{ПП}$ может быть определена по формуле 1

$$\tau_{ПП} = \tau_{ТЗ} + \tau_{ЭП} + \tau_{ТП} + \tau_{РП} + \tau_{В}, \quad 4,1$$

где $\tau_{ТЗ}$ — трудоёмкость разработки технического задания на создание ПП; $\tau_{ЭП}$ — трудоёмкость разработки эскизного проекта; $\tau_{ТП}$ — трудоёмкость разработки технического проекта ПП; $\tau_{РП}$ — трудоёмкость разработки рабочего проекта ПП; $\tau_{В}$ — трудоёмкость внедрения разработанного ПП.

4.3.1 Трудоемкость разработки технического задания

Трудоёмкость разработки технического задания рассчитывается по формуле 3:

$$\tau_{ТЗ} = T_{ЗРЗ} + T_{ЗРП}, \quad 4,2$$

где $T_{ЗРЗ}$ — затраты времени разработчика постановки задач (проектировщика) на разработку ТЗ, чел.–дни; $T_{ЗРП}$ — затраты времени разработчика программного обеспечения на разработку ТЗ, чел.–дни.

Значения величин ТЗРЗ и ТЗРП рассчитываются по формулам 3 и 4:

$$T_{ЗРЗ} = t_{ТЗ} \cdot K_{ЗРЗ} \quad 4,3$$

$$T_{ЗРП} = t_{ТЗ} \cdot K_{ЗРП} \quad 4,4$$

где $t_{ТЗ}$ — норма времени на разработку ТЗ на ПП в зависимости от функционального назначения и степени новизны разрабатываемого ПП, чел.–дни; $K_{ЗРЗ}$ — коэффициент, учитывающий удельный вес трудоёмкости работ, выполняемых разработчиком постановки задач на стадии технического задания (в случае совместной с разработчиком ПП разработки технического задания $K_{ЗРЗ} = 0.65$); $K_{ЗРП}$ — коэффициент, учитывающий удельный вес трудоёмкости работ, выполняемых разработчиком ПП на стадии технического задания (в случае совместной с разработчиком постановки задач $K_{ЗРП} = 0.35$).

По функциональному назначению программа относится к категории «Задачи расчетного характера», поэтому примем $t_{ТЗ} = 24$

Таким образом, получим

$$\tau_{ТЗ} = T_{ЗРЗ} + T_{ЗРП} = t_{ТЗ} \cdot K_{ЗРЗ} + t_{ТЗ} \cdot K_{ЗРП} = 24 \cdot 0.65 + 24 \cdot 0.35 = 24 \text{ чел.–дней.}$$

4.3.2 Трудоемкость разработки эскизного проекта

Трудоёмкость разработки эскизного проекта ПП $\tau_{ЭП}$ рассчитывают по формуле 5:

$$\tau_{ЭП} = T_{ЭРЗ} + T_{ЭРП}, \quad 4,5$$

где $T_{ЭРЗ}$ — затраты времени разработчика постановки задач на разработку эскизного проекта, чел.–дни; $T_{ЭРП}$ — затраты времени разработчика ПП на разработку эскизного проекта, чел.–дни.

Значения величин $T_{ЭРЗ}$ и $T_{ЭРП}$ рассчитываются по формулам 6 и 7:

$$T_{ЭРЗ} = t_{ЭП} \cdot K_{ЭРЗ} \quad 4,6$$

$$T_{ЭРП} = t_{ЭП} \cdot K_{ЭРП} \quad 4,7$$

где $t_{ЭП}$ — норма времени на разработку эскизного проекта на ПП в зависимости от функционального назначения и степени новизны разрабатываемого ПП, чел.–дни; $K_{ЭРЗ}$ — коэффициент, учитывающий удельный вес трудоёмкости работ, выполняемых разработчиком постановки задач на стадии эскизного проекта; $K_{ЭРП}$ — коэффициент, учитывающий удельный вес трудоёмкости работ, выполняемых разработчиком ПП на стадии эскизного проекта.

Таким образом, для разрабатываемой программы $t_{ЭП} = 70$ чел. дней, приняв, что при совместной разработке $K_{ЭРЗ} = 0.35$ и $K_{ЭРП} = 0.65$, получим по формуле 5

$$\tau_{ЭП} = T_{ЭРЗ} + T_{ЭРП} = t_{ЭП} \cdot K_{ЭРЗ} + t_{ЭП} \cdot K_{ЭРП} = 70 \cdot 0.65 + 70 \cdot 0.35 = 70 \text{ чел. –дней.}$$

4.3.3 Трудоемкость разработки технического проекта

Трудоёмкость разработки технического проекта $\tau_{ТП}$ определяется по формуле 8, как сумма времени, затраченного разработчиком постановки задач и разработчиком ПП:

$$\tau_{ТП} = (t_{ТРЗ} + t_{ТПП}) \cdot K_B \cdot K_P \quad 4,8$$

где $t_{ТРЗ}$, $t_{ТПП}$ — норма времени, затрачиваемого на разработку технического проекта разработчиком постановки задач и разработчиком ПП соответственно, чел.-дни; K_B — коэффициент учёта вида используемой информации; K_P — коэффициент учёта режима обработки информации.

Для ПО, предназначенного для управления научно-технической информацией, для количества разновидностей входной информации $k_{вх}=2$ и количества разновидностей выходной информации $k_{вых}=2$ группы новизны В принимаем $t_{ТРЗ}=34$ чел.-дней, а $t_{ТПП}=13$ чел.-дней.

Для стадии технического проекта с режимом обработки информации — ТОУ и группы новизны «В», найдём $K_P=1,36$.

Значение коэффициента K_B определяют из формулы 9:

$$K_B = (K_{П} \cdot n_{П} + K_{НС} \cdot n_{НС} + K_B) / (n_B \cdot n_{П} + n_{НС} + n_B) \quad 4,9$$

где $K_{П}$, $K_{НС}$, K_B — значения коэффициентов учёта вида используемой информации для переменной, нормативно-справочной информации и баз данных соответственно; $n_{П}$, $n_{НС}$, n_B — количество наборов данных переменной, нормативно-справочной информации и баз данных соответственно.

Для группы новизны «В» примем $K_{П}=1,00$, $K_B=2.08$, так как для хранения моделей может использоваться база данных. $K_{НС}=0$. Примем $n_{П}=1$, $n_{НС}=0$, $n_B=1$. Подставив значения в формулу 9, получаем:

$$K_B = (K_{П} \cdot n_{П} + K_{НС} \cdot n_{НС} + K_B) / (n_{П} + n_{НС} + n_B) = (1.0 + 2.08) / 2 = 1.54$$

Таким образом, по формуле 4.8 вычислим трудоёмкость разработки технического проекта

$$\tau_{ТП} = (t_{ТРЗ} + t_{ТПП}) \cdot K_B \cdot K_P = (34 + 13) \cdot 1.36 \cdot 1.54 \approx 98$$

4.3.4 Трудоемкость разработки рабочего проекта

Трудоёмкость разработки рабочего проекта $\tau_{РП}$ определяется по формуле 10:

$$\tau_{РП} = (t_{РРЗ} + t_{РРП}) \cdot K_K \cdot K_P \cdot K_Y \cdot K_3 \cdot K_{ИА}, \quad 4,10$$

где $t_{РРЗ}$, $t_{РРП}$ — нормы времени, затрачиваемые на разработку рабочего проекта на алгоритмическом языке высокого уровня разработчиком постановки задач и разработчиком ПП соответственно, чел.–дней; K_K — коэффициент учёта сложности контроля информации; K_P — коэффициент учёта режима обработки информации; K_Y — коэффициент учёта уровня алгоритмического языка программирования; K_3 — коэффициент учёта степени использования готовых программных модулей; $K_{ИА}$ — коэффициент учёта вида используемой информации и сложности алгоритма ПП.

Для ПО, предназначенного для решения задач расчетного характера, $k_{\text{вых}}=2$, $k_{\text{вх}}=2$, найдём, что $t_{РРЗ}=13$ чел.–дней, а $t_{РРП}=62$ чел.–дней.

Приняв степень сложности контроля входной информации за 11, а выходной информации за 21, получим $K_K=1,16$. Для группа «В», рабочий проект, ТОУ найдём, что $K_P=1,44$. Для алгоритмического языка высокого уровня $K_Y=1,00$. При степени использования готовых программных модулей 20–25%, получим значение коэффициента $K_3=0,8$. Значение коэффициента $K_{ИА}$ определяют по формуле 11:

$$K_{ИА} = (K_{П'} \cdot n_{П} + K_{НС'} \cdot n_{НС} + K_{Б'} \cdot n_{Б}) / (n_{П} + n_{НС} + n_{Б}) \quad 4,11$$

где $K_{П'}$, $K_{НС'}$, $K_{Б'}$ — значения коэффициентов учёта сложности алгоритма ПП и вида используемой информации для переменной, нормативно-справочной информации и баз данных соответственно.

С учётом специфики проекта для 2-й группы сложности алгоритма, и группы новизны «В», найдём: $K_{П'}=1,1$, $K_{НС'}=0$, $K_{Б'}=0,48$. Приняв, как и в п. 3.3, $n_{П}=1$, $n_{НС}=0$, $n_{Б}=1$, получаем:

$$K_{IA} = (K_{II} \cdot n_{II} + K_{HC} \cdot n_{HC} + K_B \cdot n_B) / (n_{II} + n_{HC} + n_B) = (1.1 + 0.48) / 2 = 0.79$$

По формуле 5.10 найдём трудоёмкость разработки рабочего проекта

$$\tau_{PI} = (t_{PP3} + t_{PPI}) \cdot K_K \cdot K_P \cdot K_J \cdot K_3 \cdot K_{IA} = (13 + 62) \cdot 1.16 \cdot 1.44 \cdot 1.0 \cdot 0.8 \cdot 0.79 \approx 79 \text{ чел.-дней.}$$

4.3.5 Трудоёмкость выполнения стадии «Внедрение»

Трудоёмкость выполнения внедрения разработанного ПП рассчитывается по формуле 12

$$\tau_B = (t_{BP3} + t_{BPI}) \cdot K_K \cdot K_P \cdot K_3 \quad 4,12$$

где t_{BP3} , t_{BPI} — норма времени, затрачиваемого разработчиком постановки задач и разработчиком ПП соответственно на выполнение процедур внедрения ПП, чел.-дни.

Для ПО управления научно-технической информацией, $k_{ВЫХ}=2$, $k_{ВХ}=2$, находим $t_{BP3}=8$ чел.-дней и $t_{BPI}=13$ чел.-дней. Следовательно

$$\tau_B = (t_{BP3} + t_{BPI}) \cdot K_K \cdot K_P \cdot K_3 = (8 + 13) \cdot 1.16 \cdot 1.44 \cdot 0.8 = 28 \text{ чел.-дней.}$$

Общая трудоёмкость разработки

$$\tau_{III} = \tau_{T3} + \tau_{ЭП} + \tau_{ТП} + \tau_{PI} + \tau_B = 24 + 70 + 98 + 79 + 28 = 299 \text{ чел.-дней.}$$

Планирование и контроль хода выполнения разработки проводят по календарному плану выполнения работ. На рисунке 4.1 изображена диаграмма Ганта. При построении учитываем одного исполнителя – дипломника и нерабочие дни – выходные и праздники.

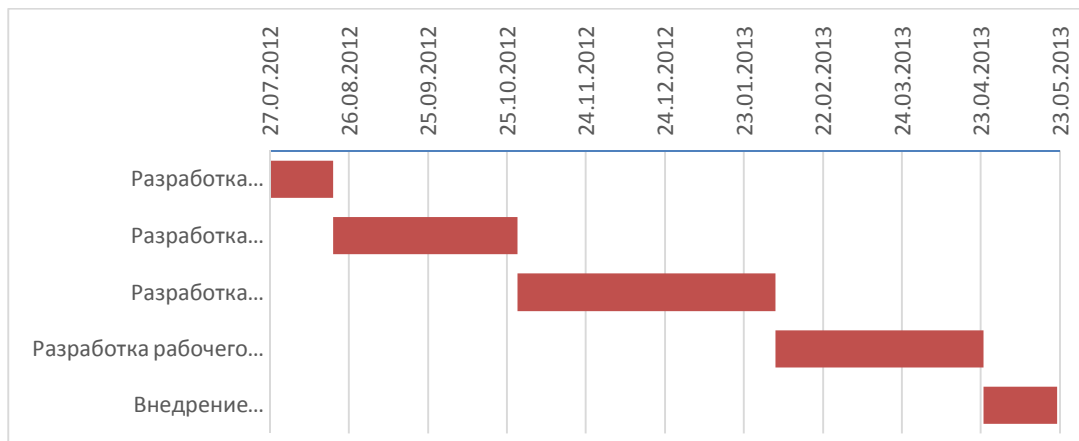


Рис. 4.1 Диаграмма Ганта

4.4 Расчет затрат на реализацию программного продукта

4.4.1 Расчет материальных затрат

Материальные затраты для реализации прототипа системы моделирование ПА10 отображены в таблице 5.

Таблица 5. Материальные затраты

Наименование материала	Количество, шт.	Цена за единицу, руб.	Сумма, руб.
ПЭВМ	1	18400	18400
Итого:			18400

Отсутствие затрат на программное обеспечение обусловлено наличием бесплатных лицензий, полученных кафедрой при сотрудничестве с компанией Microsoft, а также отсутствием приобретения коммерческой лицензии Qt Commercial, т.к. полученный продукт не предназначен для коммерческого использования.

Таким образом материальные затраты составляют:

$$C_m = 18400 \text{ руб.}$$

4.4.2 Расчет амортизационных отчислений

Амортизационные отчисления производятся предприятиями ежемесячно исходя из установленных норм амортизации и балансовой (первоначальной или восстановительной) стоимости основных фондов по отдельным группам или инвентарным объектам, состоящим на балансе предприятия. Нормы амортизации устанавливаются государством и они едины для всех предприятий и организаций.

Амортизационные отчисления на полное восстановление активной части основных фондов (машин, оборудования и транспортных средств) производятся в течение нормативного срока их службы или срока, за который балансовая стоимость этих фондов полностью переносится на себестоимость.

По всем другим основным фондам амортизационные отчисления на полное восстановление производится в течение всего фактического срока их службы.

Предприятиям допускается применение ускоренной амортизации их активной части в более короткие сроки, нормы амортизации при этом повышаются, но не более чем в два раза. Применение повышенных или пониженных норм амортизации должно быть предусмотрено в учетной политике предприятия, определяемой его руководителем.

Амортизационные отчисления определяются по формуле 13:

$$AO = \frac{K_{c.o.} \cdot H_a \cdot t_{об}}{\Phi_o} \quad 4,13$$

Где $K_{c.o.}$ - материальные затраты, руб.;

H_a - норма годовых амортизационных отчислений, % (электронные цифровые вычислительные машины общего назначения, специализированные и управляющие 12,5%);

$t_{об}$ - машинное время, необходимое для выполнения НИОКР, час;

Φ_o - действительный фонд времени работы оборудования за год, час.

$$AO = 18400 \cdot 0.125 \cdot (299 \cdot 8) / (8 \cdot 22 \cdot 12) = 2604 \text{ руб.}$$

4.4.3 Расчет заработной платы

В статью «Основная заработная плата» включается основная заработная плата всех исполнителей, непосредственно занятых разработкой данной программной продукции, с учётом их должностного оклада и времени участия в разработке. Расчёт ведётся по формуле 14:

$$C_{з.полн} = C_{з.осн.} + C_{з.доп} + C_{з.отч} , \quad 4,14$$

где $C_{з.осн.}$ — основная заработная плата, $C_{з.доп}$ — дополнительная заработная плата, $C_{з.отч.}$ — отчисление с заработной платы.

Расчет основной заработной платы (оплаты труда непосредственных исполнителей) производится по формуле 15:

$$C_{з.осн} = T_{зан} \cdot O_{дн}, \quad 4,15$$

где $T_{зан}$ — число дней, отработанных исполнителем проекта, $O_{дн}$ — дневной оклад исполнителя.

При 8-и часовом рабочем дне оклад исполнителя рассчитывается по соотношению в формуле 16:

$$O_{дн} = O_{мес} \cdot 8 / F_M, \quad 4,16$$

где $O_{мес}$ - месячный оклад, F_M - месячный фонд рабочего времени.

С учетом налога на доходы физических лиц размер оклада увеличивается, что отражено в формуле 17:

$$O_{мес} = O \cdot (1 + H_{НДФЛ} / 100), \quad 4,17$$

где O — «чистый» оклад, НДФЛ— налог на доходы физических лиц в размере 13%.

Результаты расчета с перечнем исполнителей и их месячных и дневных окладов, а также их трудозатратах и рассчитанной основной заработной платой каждого исполнителя приведены в таблице 4.4.

В статье «Дополнительная заработная плата» учитываются все выплаты непосредственным исполнителям за время, не проработанное на производстве, и определяются по формуле 18:

$$C_{з.доп} = C_{з.осн} \cdot \alpha_d, \quad 4,18$$

где α_d — коэффициент отчислений на дополнительную зарплату, примем $\alpha_d=0,2$.

Рассчитанные значения заработной платы приведены в таблице 6.

Таблица 6. Затраты на основную заработную плату сотрудников

Исполнитель	Месячный оклад, руб.	Дневная заработная плата, руб.	Продолжительность работы, дн.	Основная заработная плата, руб.	Дополнительная заработная плата, руб.
Инженер-программист	50000	2995	299	895505	179101

Получаем расходы на заработную плату

$$C_{\text{ЗАРП}} = C_{\text{З.ОСН}} + C_{\text{З.ДОП}} = 895505 + 179101 = 1074606 \text{ руб.}$$

4.4.4 Расчет отчислений в социальные фонды

Отчисления с заработной платы состоят в настоящее время в уплате единого социального налога. Налоговым кодексом РФ определяются ставки налога для отчисления в пенсионный фонд РФ, фонд социального страхования, фонды обязательного медицинского страхования (федеральный и территориальный фонды). На момент расчета экономической части продукта сумму ставок ЕСН соответствует 30%.

Отчисления с заработной платы составят:

$$C_{\text{З.ОТЧ}} = (C_{\text{З.ОСН}} + C_{\text{З.ДОП}}) \cdot H_{\text{СОЦ}} = 1074606 \cdot 0.3 = 322382 \text{ руб.}$$

Таким образом, общие затраты на заработную плату составят:

$$C_{\text{З.ПОЛН}} = 895505 + 179101 + 322382 = 1396988 \text{ руб.}$$

Ставка взносов при общей системе налогообложения разбивается по фондам как показано в таблице 7.

Таблица 7. Ставка взносов

Наименование фонда		Ставка взносов, %
Пенсионный фонд	Страховая часть	16
	Накопительная часть	6
Фонд медицинского страхования	ФФОМС	2,9
	ТФОМС	5,1
Всего:		30

4.4.5 Прочие расходы

Расходы на интернет:

$$C_{\text{пр}} = 650 \cdot 10 = 6500 \text{ руб.}$$

Таблица 8. Итоговые значение затрат на реализацию.

Тип	Значение, руб.
Материальные затраты	18400
Амортизационные отчисления	2604
Заработная плата	1074606
Отчисления в социальные фонды	322382
Прочие затраты	6500

4.5 Определение цены программного продукта

Таким образом, затраты на разработку программной продукции (сметная себестоимость): $C=958181,6$ руб.

На рисунке 4.2 приведена структура затрат на выполнение проекта.

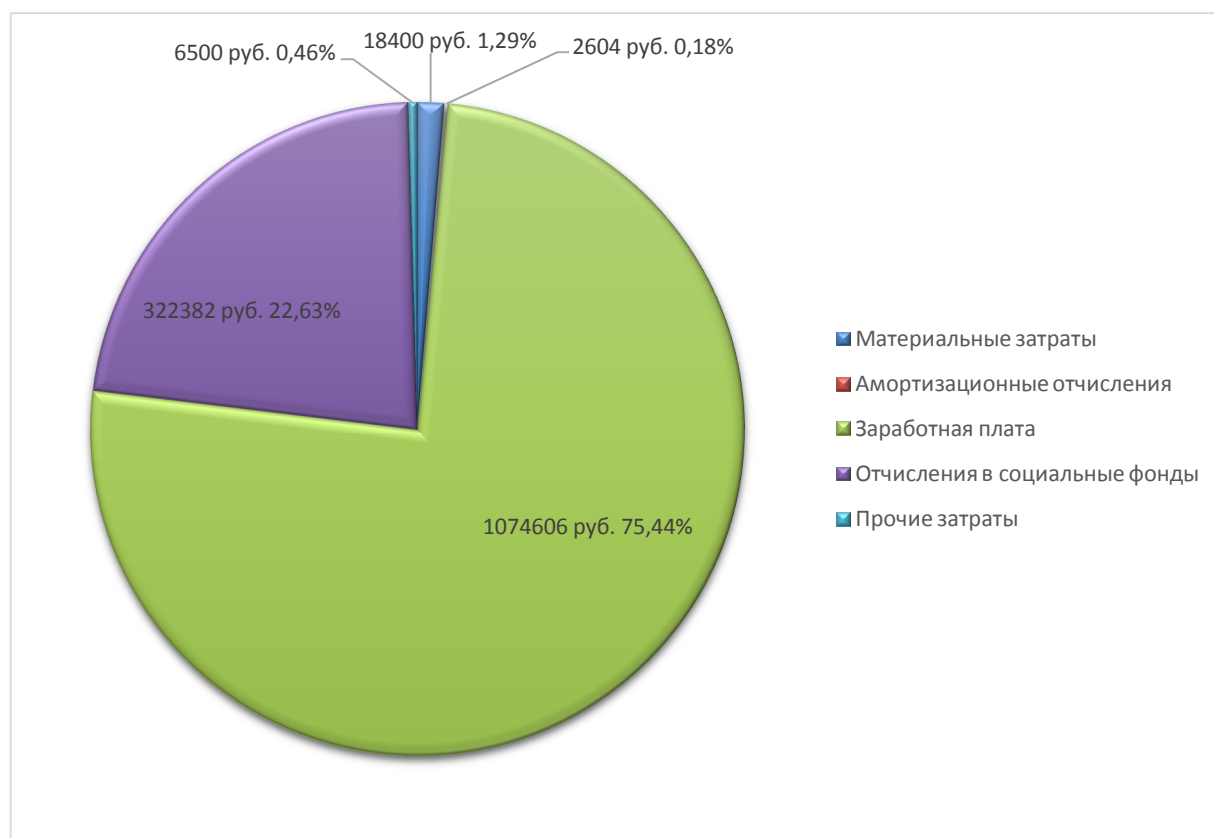


Рис. 4.2 Структура затрат

Результаты расчётов затрат на разработку программного продукта сведены в таблице 9.

Таблица 9. Затраты на программный продукт

№ п/п	Статьи затрат на НИОКР	Сумма, руб.	Затраты, %
1	Материальные затраты	18400	1,2
2	Амортизационные отчисления	2604	0,2
3	Заработная плата	1074606	75,4
4	Отчисления в социальные фонды	322382	22,6
5	Прочие расходы	6500	0,6
Итого	Итого:	1424492	100

Цена программной продукции определяется по формуле 19:

$$C_{ПП} = \Delta C_1 + D_{ПРИБ}, \quad 4,19$$

где ΔC_1 — часть стоимости разработки, приходящаяся на одну копию программы; $D_{ПРИБ}$ — процент прибыли, заложенный в цену(в нашем случае не учитывается, т.к. продукт носит некоммерческий характер).

Частичная стоимость разработки, приходящаяся на каждый комплект ПП, определяется по формуле 20 на основании данных о планируемом объеме установок:

$$\Delta C = C_C \cdot N_K, \quad 4,20$$

где C_C — сметная стоимость проекта; N_K — планируемое число копий ПП.

Примем $N_K=70$, тогда:

$$C_{ПП} = \Delta C + D_{ПРИБ} = C_C / N_K + D_{ПРИБ} = 1424492 / 70 = 20350 \text{ руб.}$$

4.6 Выводы

В результате расчётов было получено общее время выполнения проекта, которое составило 299 дней.

Для выполнения проекта будет задействован один исполнитель – студент-дипломник.

Подсчитаны затраты на создание системы моделирования, которые составили 1424492 рублей.

Результаты объясняются неоднозначностью классификации некоторых элементов в используемой методике. Кроме того, согласно методике программист работает на всех стадиях создания проекта, на самом же деле это время значительно меньше.

Была получена цена одной копии программы 20350 рублей. Копии рассчитаны на использование внутри вуза.

Глава 5 Экологическая часть **Equation Chapter (Next) Section 1**

5.1 Введение

Разработка и использование программного продукта, являющегося результатом дипломного проектирования сопряжены с использованием ПЭВМ.

При работе на ПЭВМ оператор подвергается воздействию совокупности физических, химических, биологических, социально-психологических и эстетических факторов. Данный раздел посвящен анализу вредных факторов, действующих на пользователя ПЭВМ и проектированию средств защиты от воздействия этих факторов. Анализ производится на соответствие СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы», а также ГОСТ 12.1.004-91 – пожарная безопасность.

5.2 Основные факторы

Воздействие, которое компьютерная техника способна оказать на человека можно разделить на три группы.

Первая группа – физическое воздействие: компьютер (в частности видеотерминалы) является источником электромагнитного поля промышленной частоты, электромагнитного излучения радиодиапазона, электростатического и постоянного магнитного полей, рентгеновского излучения. Так же компьютер и периферийное оборудование: вентиляционные устройства ПЭВМ, агрегаты кондиционирования и вентилирования воздуха могут создавать шум, а так же изменять микроклимат и ионизацию воздуха в рабочем помещении.

Вторая группа – нагрузка на опорно-двигательный аппарат человека: интенсивная работа с клавиатурой и мышью может вызывать болевые ощущения в пальцах рук, кистях, запястьях, предплечьях и локтевых суставах. Длительное

пребывание в неподвижной, неудобной позе приводит к усталости и болям в позвоночнике, шее, плечевых суставах и мышцах спины.

Третья группа – напряженность труда: работа с компьютером предполагает визуальное восприятие и анализ больших объемов информации, что вызывает утомление зрительного аппарата человека и перегрузку его мозга и центральной нервной системы.

При организации рабочего места оператора ПЭВМ должны быть соблюдены следующие основные условия:

- a) допустимые параметры микроклимата;
- b) допустимые уровни электромагнитное излучение;
- c) эргономичность рабочего места и используемых устройств;
- d) оптимальное размещение оборудования, входящего в состав рабочего места;
- e) достаточное рабочее пространство, позволяющее осуществлять все необходимые движения и перемещения;
- f) обеспечение естественным и искусственным светом в пределах нормы;
- g) контроль уровня акустического;
- h) вентиляция рабочего места;
- i) обеспечение электробезопасности и пожаробезопасности.

5.3 Общие положения организации рабочего места

Согласно СанПиН 2.2.2/2.4.1340-03, помещения для работы с компьютерами должны иметь естественное и искусственное освещение, оборудоваться системами отопления, кондиционирования воздуха или эффективной приточно-вытяжной вентиляции.

Эксплуатация ПЭВМ 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 регулирующими устройствами типа: жалюзи, занавесей, внешних козырьков и др.

Площадь на одно рабочее место пользователей ПЭВМ с ВДТ на базе электронно-лучевой трубки (ЭЛТ) должна составлять не менее 6 м², в помещениях культурно-развлекательных учреждений и с ВДТ на базе плоских дискретных экранов (жидкокристаллические, плазменные) – 4,5 м².

При использовании ПЭВМ с ВДТ на базе ЭЛТ (без вспомогательных устройств – принтер, сканер и др.), отвечающих требованиям международных стандартов безопасности компьютеров, с продолжительностью работы менее 4-х часов в день допускается минимальная площадь 4,5 м² на одно рабочее место пользователя (взрослого и учащегося высшего профессионального образования).

Для внутренней отделки интерьера помещений, где расположены ПЭВМ, 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 ПЭВМ при наличии санитарно-эпидемиологического заключения. Помещения, где размещаются рабочие места с ПЭВМ, должны быть оборудованы защитным заземлением в соответствии с техническими требованиями по эксплуатации.

Не следует размещать рабочие места с ПЭВМ вблизи силовых кабелей и вводов, высоковольтных трансформаторов, технологического оборудования, создающего помехи в работе ПЭВМ. Расчет воздухообмена следует проводить по теплоизбыткам от оборудования, людей, солнечной радиации и искусственного освещения. Параметры микроклимата, ионного состава воздуха, содержание вредных веществ в нем должны отвечать нормативным требованиям.

Звукоизоляция помещений и звукопоглощение ограждающих конструкций помещения должны отвечать гигиеническим требованиям и обеспечивать нормируемые параметры шума на рабочих местах. Помещения должны иметь естественное и искусственное освещение.

В помещениях ежедневно должна проводиться влажная уборка.

Помещения с компьютерами должны быть оснащены аптечкой первой помощи и углекислотными огнетушителями.

5.3.1 Обеспечение параметров микроклимата

Производственные помещения, в которых работа с ПЭВМ является основной (операторские, расчетные, залы вычислительной техники и др.), должны обеспечиваться оптимальные параметры микроклимата. В соответствии с СанПиН 2.2.2/2.4.1340-03 устанавливаются категории тяжести работ 1а и 1б. Продолжительность работы оператора за ПЭВМ подразумевает 50% времени от смены. Все работы производятся сидя и не требуют особо тяжёлого физического напряжения, расход энергии составляет до 120 ккал/час, поэтому необходимо обеспечить сотрудника помещением с учётом оптимальных норм микроклимата для категории работ 1а.

Таблица 10. Оптимальные параметры микроклимата для помещений с ВДТ и ПЭВМ

Нормы микроклимата для помещений с ВДТ и ПЭВМ.			
Температура, град. С	Относительная влажность, %	Абсолютная влажность, г/см ³	Скорость движения воздуха, м/с
19	62	10	< 0,1
20	58	10	< 0,1
21	55	10	< 0,1

Примечание: Скорость движения воздуха – не более 0,1 м/с.

Проветривание и вентиляция воздуха в помещениях позволяют поддерживать требуемые параметры микроклимата, а также поддерживать постоянный уровень ионизации воздуха. Для повышения влажности воздуха в

помещениях с компьютерами следует также применять увлажнители воздуха, заправляемые ежедневно дистиллированной или прокипяченной питьевой водой.

В помещениях, оборудованных ПЭВМ, также необходимо проводить ежедневную влажную уборку после каждого часа работы на ПЭВМ.

Недостаток аэроионов пагубно сказывается на здоровье пользователя, у него снижается иммунитет к различным заболеваниям. Нормы содержания аэроионов в воздухе производственного помещения отражены в таблице 11.

Таблица 11. Нормы содержания аэроионов в воздухе рабочих помещений

Уровни ионизации	Число ионов в 1 см ³ воздуха помещения	
	положительных	отрицательных
Минимально необходимые	400	600
Оптимальные	1 500 - 3 000	3 000 - 5 000
Максимально допустимые	50 000	50 000

Содержание вредных химических веществ в производственных помещениях не должно превышать предельно допустимых концентраций (ПДК) загрязняющих веществ в атмосферном воздухе населенных мест в соответствии с действующими гигиеническими нормативами (ГН 2.1.6.1338-03).

5.3.2 Эргономичность

Проектирование рабочих мест, снабженных видеотерминалами, относится к числу важнейших проблем эргономического проектирования в области вычислительной техники. Визуальные эргономические параметры ВДТ являются параметрами безопасности, и их неправильный выбор приводит к ухудшению здоровья пользователей.

Рабочее место и взаимное расположение всех его элементов должно соответствовать (согласно СанПиН 2.2.2/2.4.1340-03) антропометрическим, физическим и психологическим требованиям. Большое значение имеет также характер работы. В частности, при организации рабочего места программиста должны быть соблюдены следующие основные условия:

- a) оптимальное размещение оборудования, входящего в состав рабочего места;
- b) достаточное рабочее пространство, позволяющее осуществлять все необходимые движения и перемещения;
- c) необходимо естественное и искусственное освещение для выполнения поставленных задач;
- d) уровень акустического шума не должен превышать допустимого значения;
- e) достаточная вентиляция рабочего места.

Главными элементами рабочего места программиста являются письменный стол и кресло. Основным рабочим положением является положение сидя. Работа с вычислительной техникой занимает большую часть времени (до 8 часов машинного времени в сутки). Чтобы такая работа не приводила к быстрому утомлению необходимо создать комфортные для работы условия. Нормальная и безопасная работа пользователя ЭВМ (оператора) во многом зависит от того, в какой мере условия его работы соответствуют оптимальным: комплекс физических, химических, биологических и психофизиологических факторов.

Выбор рабочей позы производится из следующих соображений: рабочая поза сидя вызывает минимальное утомление программиста. Рациональная планировка рабочего места предусматривает четкий порядок и постоянство размещения предметов, средств труда и документации. 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. При проектировании рабочего места тщательно подбирается рабочее кресло (подлокотники и подножки, регулировка по высоте, горизонту, по углу наклона спинки, их размеры и свободное пространство вокруг кресла), стол (оптимальная высота – 750 мм над уровнем пола), а также дополнительные тумбы и полки. 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. Рабочий стул (кресло) должен быть подъемно-поворотным и регулируемым по высоте и углам наклона сиденья и спинки, а так же - расстоянию спинки от переднего края сиденья.

5.3.3 Оптимальное размещение оборудования

Площадь на одно рабочее место с компьютером для взрослых пользователей должна составлять не менее 6,0 м², а объем – не менее 20,0 м³.

Рабочие места по отношению к световым проемам должны располагаться так, чтобы естественный свет падал сбоку, преимущественно слева.

5.3.4 Обеспечение электробезопасности

Производителями персональных компьютеров предусмотрены все существующие способы обеспечения электробезопасности. Конструкция использованного в дипломной работе компьютера обеспечивает надежную электробезопасность для работающего с ним человека: по способу защиты от поражения электрическим током удовлетворяет требованиям 1 класса ГОСТ 12.2.007.0 и ГОСТ Р50377; по обеспечению электробезопасности обслуживающего персонала соответствует ГОСТ Р50377.

Защита от поражения электрическим током обеспечивается различными способами: размещением разъемов электропитания на тыльной стороне системного блока и монитора; применением надежных изоляционных материалов; использованием для электропитания клавиатуры, ручных манипуляторов, в интерфейсных кабелях и в элементах регулировки и индикации на лицевой панели системного блока и монитора низковольтных напряжений (не более 12В).

Системный блок и монитор подключены к трехфазной сети переменного тока напряжением 220 В и частотой 50 Гц.

5.3.5 Обеспечение допустимого уровня шума

Основной вклад в шум в помещении вносят работающие вентиляторы систем кондиционирования.

В рабочем пространстве, где осуществляется речевой обмен информацией, уровень шума должен быть менее 55 дБ.

При выполнении основной работы на ВДТ и ПЭВМ уровень шума на рабочем месте не должен превышать 60 дБа. На рабочих местах в помещениях для размещения шумных агрегатов вычислительных машин (АЦПУ, принтеры и т.п.) уровень шума не должен превышать 75 дБа. Шумящее оборудование (АЦПУ, принтеры и т.п.), уровни шума которого превышают нормированные, должно находиться вне помещения с ВДТ и ПЭВМ.

Допустимые значения уровней звукового давления в октавных полосах частот и уровня звука, создаваемого ПЭВМ приведены в таблице 12.

Таблица 12. Допустимые значения уровней звукового давления

Уровни звукового давления в октавных полосах со среднегеометрическими частотами									Уровни звука в дБа
31,5 Гц	63 Гц	125 Гц	250 Гц	500 Гц	1000 Гц	2000 Гц	4000 Гц	8000 Гц	
86 дБ	71 дБ	61 дБ	54 дБ	49 дБ	45 дБ	42 дБ	40 дБ	38 дБ	50

Измерение уровня звука и уровней звукового давления проводится на расстоянии 50 см от поверхности оборудования и на высоте расположения источника (источников) звука.

5.3.6 Обеспечение допустимых эргономических характеристики дисплеев

Требования к набору параметров для расчета характеристик дисплея отображены в таблице 13.

Наиболее критичным параметром, влияющим на физическое здоровье оператора ПЭВМ, является временная нестабильность изображения. Не рекомендуется использовать дисплеи, имеющие данный показатель меньше 80 Гц.

Таблица 13. Допустимые параметры дисплея

№	Параметры	Допустимые значения
1.	Яркость белого поля	Не менее 35 кд/кв. м
2.	Неравномерность яркости рабочего поля	Не более +20
3.	Контрастность (для монохромного режима)	Не менее 3:1
4.	Временная нестабильность изображения (мелькания)	Не должна фиксироваться
5.	Пространственная нестабильность изображения (дрожание)	Не более $2 \times 10 (-4L)$, где L – проектное расстояние наблюдения, мм

5.3.7 Обеспечение пожаробезопасности

Помещения, где располагается разработанный модуль или где он будет функционировать, относятся к категории "В" пожаробезопасности - в них находятся твердые сгораемые вещества, не способные взрываться при контакте с кислородом воздуха.

При использовании ПЭВМ возможны аварийные ситуации: короткие замыкания, перегрузки, повышение переходных сопротивлений в электрических контактах, перенапряжение, возникновение токов утечки. Эти случаи могут привести к резкому выделению тепловой энергии, которая может стать причиной возникновения пожара. Для снижения вероятности подобной ситуации необходимо, чтобы помещения были оснащены противопожарной сигнализацией, ручными углекислотными огнетушителями, например, ОУ-3. Требования к пожаробезопасности зданий и сооружений определяются согласно СНиП 21.01-97.

5.3.8 Освещенность

Правильное освещение рабочего места является одним из важнейших факторов, влияющих на эффективность трудовой деятельности человека, предупреждающих травматизм и профессиональные заболевания. Работник должен быть обеспечен такими условиями освещенности, при которых он может

без напряжения для зрения выполнять свою работу. Утомляемость органов зрения зависит от ряда причин:

- недостаточность освещенности,
- чрезмерная освещенность,
- неправильное направление света.

Недостаточность освещения приводит к напряжению зрения, ослабляет внимание, приводит к наступлению преждевременной утомленности. Чрезмерно яркое освещение вызывает ослепление, раздражение и резь в глазах. Неправильное направление света на рабочем месте может создавать резкие тени, блики, дезориентировать работающего. Все эти причины могут привести к несчастному случаю или профзаболеваниям, поэтому столь важен правильный расчет освещенности.

Требования к искусственному освещению изложены в разделе 6 СанПиН 2.2.2/2.4.1340-03. Освещение в помещениях для эксплуатации ПЭВМ 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 (светильники и др.), находящихся в поле зрения, должна быть не более 200 кд/м².

Следует ограничивать отраженную блесткость на рабочих поверхностях (экран, стол, клавиатура и др.) за счет правильного выбора типов светильников и расположения рабочих мест по отношению к источникам искусственного

освещения, при этом яркость бликов на экране ПЭВМ 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.

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.

Следует ограничивать неравномерность распределения яркости в поле зрения пользователя ПЭВМ, 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.

В качестве источников света при искусственном освещении следует применять преимущественно люминесцентные лампы типа ЛБ и компактные люминесцентные лампы (КЛЛ). При устройстве отраженного освещения в производственных и административно-общественных помещениях допускается применение металлогалогенных ламп. В светильниках местного освещения допускается применение ламп накаливания, в том числе галогенные. Для освещения помещений с ПЭВМ следует применять светильники с зеркальными параболическими решетками, укомплектованными электронными пускорегулирующими аппаратами (ЭПРА). Допускается использование многоламповых светильников с электромагнитными пускорегулирующими 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.

На практике же помещение для выполнения работ имеет систему освещения не соответствующую нормам. Она представляет собой двенадцать люминесцентных светильников с лампами ЛД мощностью по 40 Ватт. Такая система дает освещенность стола в зоне рабочего места порядка 150 – 200 лк. Как следствие, соотношение яркости между рабочей поверхностью и поверхностями стен превышает значение 10:1. Расчет рекомендуемой системы искусственного освещения, соответствующей санитарным нормам, приведен далее.

5.4 Расчет системы искусственного освещения

5.4.1 Выбор источников света

К числу источников искусственного света, выпускаемых отечественной и зарубежной промышленностью, относятся лампы накаливания, люминесцентные лампы и лампы ДРЛ. Предпочтение чаще всего отдается люминесцентным лампам, как более экономичным и обладающим более благоприятной цветностью излучения. В системах одного общего освещения офисных помещений, конструкторских бюро и т.п., а также для общего освещения в системе комбинированного освещения во всех случаях также рекомендуется использовать люминесцентные лампы.

5.4.2 Выбор системы освещения

В практике проектирования осветительных установок используются следующие системы освещения:

- а) общего освещения,
- б) комбинированного освещения.

Общее освещение в свою очередь подразделяется на общее равномерное (при равномерном распределении светового потока без учета положения

оборудования) и общее локализованное (при распределении светового потока с учетом расположения рабочих мест).

Для применения в офисных помещениях, где нет теней на рассматриваемой поверхности (помещение для работы инженеров – разработчиков) рекомендуется система общего равномерного освещения.

5.4.3 Выбор осветительных приборов

При проектировании осветительной установки следует руководствоваться основными показателями выбора светильника: его конструктивное исполнение с учетом условий среды, его светораспределение, экономичность.

Для нашего случая выбираем открытые двухламповые люминесцентные светильники ОДОР – они рекомендуются для нормальных помещений с хорошим отражением потолка и стен, допускаются при умеренной влажности и запыленности. Их минимальная высота подвеса составляет 3,5 метра.

5.4.4 Размещение осветительных приборов

Критерии расположения светильников в помещении:

- а) обеспечение высокого качества освещения, ограничение ослеплённости и необходимой направленности света на рабочее место;
- б) наиболее экономичное создание нормированной освещенности.

Размещение рабочих столов должно быть таким образом, чтобы видеодисплейные терминалы были ориентированы боковой стороной к световым проемам, чтобы естественный свет падал преимущественно слева.

Для равномерного общего освещения светильники с люминесцентными лампами должны располагаться рядами параллельно стенам с окнами.

Относительное расстояние – λ , между светильниками должно определяться по формуле 1:

$$\lambda = L / h, \quad 5,1$$

где L – расстояние между соседними светильниками, м;

h – высота подвеса светильника над рабочей поверхностью, м.

При этом различают наиболее оптимальное светотехническое расположение светильников λ_c , при котором достигается наибольшая равномерность освещенности по площади помещения, и энергетически наиболее оптимальное расположение λ_3 , когда обеспечивается нормируемая освещенность при наименьших энергетических затратах.

Для светильников ОДОР $\lambda_c = 1.4$ Показатель λ_3 для светильников ОДОР не используется. Для средней высоты рабочего стола 0.75 м, вычисляем h .

$$h = H - 0.75 = 2.85 - 0.75 = 2.1 \text{ м}, \quad 5,2$$

где $H = 2.85$ м – высота потолка в помещении.

$$L = 1.4 \cdot h = 1.4 \cdot 2.1 = 2.94 \text{ м} \quad 5,3$$

Полагаем число рядов светильников в помещении $N_1 = 2$, а число двухламповых светильников в ряду $N_2 = 1$.

Схема светотехнической установки показана на рисунке 5.1.

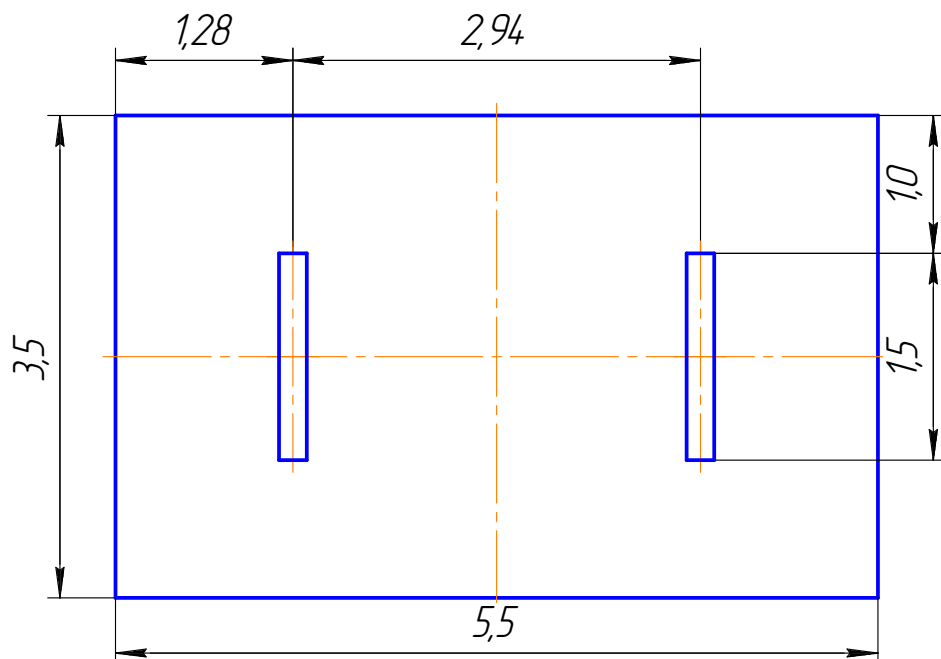


Рис 5.1 Схема светотехнической установки

5.4.5 Выбор освещенности и коэффициента запаса

Основные требования и значения нормируемой освещенности рабочих поверхностей изложены в санитарных нормах и правилах. Освещённость должна быть в пределах $300 - 500$ лк. Освещение не должно создавать бликов на поверхности экрана. Нормированная минимальная освещенность (лк) определяется по таблице 1 раздел 5.3 СНиП 23-05-95. Работу инженера, в соответствии с этой таблицей, можно отнести к разряду точных работ (3 разряд зрительной работы, подразряд В), следовательно, освещенность должна быть $E \leq 300$ лк при использовании газоразрядных ламп. Коэффициент запаса K , учитывающий уменьшение светового потока лампы в результате загрязнения светильников в процессе эксплуатации (его значение определяется по таблице 3 разд. 5.3 СНиП 23-05-95) для данного расчёта равен 1,2.

5.4.6 Расчет системы искусственного освещения

Определим световой поток каждой лампы для создания нормируемой освещенности. Методически различают два способа расчета: метод коэффициента использования светового потока и точечный метод.

В расчётах взят метод коэффициента использования светового потока. Он пригоден для помещений с равномерным размещением светильников; при расчете учитывается световой поток, отраженный от стен и потолка, а поэтому данный метод наиболее пригоден для помещений со светлыми потолком и стенами, особенно при использовании светильников рассеянного и отраженного света. Метод пригодится для определения освещенности только на горизонтальной поверхности, не применим для расчета локализованного освещения. Широко используется для расчета осветительных установок с люминесцентными лампами в конструкторских бюро.

Световой поток F двух ламп светильника рассчитывается по формуле

$$F = \frac{E_H \cdot S \cdot k \cdot z}{\eta \cdot N} \quad 5,4$$

где E_H – выбранная нормируемая освещенность, лк;

S – площадь помещения, м²;

k – коэффициент запаса;

z – отношение средней освещенности к минимальной ($z = 1,1 \dots 1,15$);

N – число светильников – 2 шт.;

η – коэффициент использования светового потока ламп, зависящий от типа светильника, коэффициентов отражения потолка $\rho_{\text{п}}$ и стен $\rho_{\text{с}}$ и индекса помещения i .

$E_H = 300 \text{ Лк}$ (в соотв. с СанПиН 2.2.2/1340-03 раздел 6). Индекс помещения выражает геометрические соотношения в помещении и определяется по формуле

$$i = \frac{S}{h \cdot (A + B)}, \quad 5,5$$

где h – высота подвеса светильника над рабочей поверхностью;

S – площадь помещения;

A и B – длина и ширина помещения соответственно.

Получаем

$$i = \frac{19.25}{2.1 \cdot (5.5 + 3.5)} = 1.01.$$

Принимаем $\rho_{\text{п}} = 50\%$, $\rho_{\text{с}} = 30\%$. По табличным данным определяем $\eta = 43\%$.

Из (5) определяем

$$F = \frac{300 \cdot 19.25 \cdot 1.15}{0.43 \cdot 2} = 7720 \text{ Лм},$$

где F – световой поток в расчете на один двухламповый светильник.

После определения светового потока, подбираем ближайшую стандартную лампу по таблице 14.

Таблица 14. Тип лампы в соответствии со значением светового потока

Люминесцентные лампы	
Тип лампы	Световой поток, лм
ЛДЦ 20	820
ЛД 20	920
ЛБ 20	1180
ЛДЦ 30	1450
ЛД 30	1640
ЛБ 30	2100
ЛДЦ 40	2100
ЛД 40	2340
ЛБ 40	3000
ЛДЦ 80	3560
ЛД 80	4070
ЛБ 80	5220

В практике допускается отклонение потока выбранной лампы от расчетного до -10 и $+20\%$, в противном случае задается другая схема расположения светильников. В нашем случае для двухлампового светильника выбираем лампу ЛД 80. Тогда световой поток двухлампового светильника составляет $F=4070 \cdot 2=8140$ (Лм) выбранный.

В конце расчета подсчитаем фактическое значение минимальной освещенности рабочей поверхности с учетом выбранной лампы:

$$E_{\min} = E_H \frac{F_{\text{выбранный}}}{F_{\text{расчетный}}} = 300 \cdot \frac{4070 \cdot 2}{7720} = 316$$

Потребная мощность всей осветительной установки (2 двухламповых светильников) P_{Σ} составит

$$P_{\Sigma} = P \cdot N = 80 \cdot 4 = 320 \text{ Вт},$$

где P – мощность одной лампы, Вт.

5.4.7 Утилизация элементов системы искусственного освещения

Для освещения были выбраны лампы ЛДЦ 80, которые относятся к классу ртутьсодержащих ламп. Утилизация данного вида ламп соответствует Постановлению № 681 Правительства Российской Федерации от 3 сентября 2010г. «Об утверждении правил обращения с отходами производства и потребления в части осветительных устройств, электрических ламп, ненадлежащие сбор, накопление, использование, обезвреживание, транспортирование и размещение которых может повлечь причинение вреда жизни, здоровью граждан, вреда животным, растениям и окружающей среде».

Все люминесцентные лампы содержат ртуть (в дозах от 1 до 70 мг), ядовитое вещество 1-го класса опасности. Эта доза может причинить вред здоровью, если лампа разбилась, и если постоянно подвергаться пагубному воздействию паров ртути, то они будут накапливаться в организме человека, нанося вред здоровью. Такие лампы нельзя выкидывать в мусоропровод или уличные контейнеры.

Существуют различные фирмы, специализирующиеся на утилизации ламп. Юридические лица, а также индивидуальные предприниматели обязаны сдавать лампы на переработку и разрабатывать паспорт опасного отхода. Кроме того, в ряде городов существуют полигоны по утилизации токсичных отходов, принимающие отходы от частных лиц бесплатно. В Москве перегоревшие люминесцентные лампы бесплатно принимаются для дальнейшей переработки в районных отделениях ЖЭК, ДЭЗ или РЭУ, где установлены специальные контейнеры.

Процесс утилизации ламп на специализированном предприятии выглядит следующим образом. Лампы транспортируются на мусоросжигательный завод в специальных контейнерах плотно прижатыми для избегания разбивания и накрытые брезентом для избегания намокания. Обезвреживание ламп происходит в вибромеханической установке. Устройство дробит их на мелкие кусочки и делит на три фракции: алюминиевые цоколи, стекло и опасное ртутьсодержащее

вещество (люминофор). Люминофор отделяется от стекла благодаря интенсивному трению осколков лампы друг о друга. Далее аппарат распределяет составляющие люминесцентных ламп по разным емкостям. После чего металл отправляют на заводы по переплавке, а люминофор, содержащий ртуть, транспортируют в герметичных бочках на переработку. Битое стекло хоронят на полигонах твердых бытовых отходов.

5.5 Выводы

В данном разделе дипломной работы был проведен анализ основных вредных и опасных факторов, воздействующих на разработчика, выявлен самый неблагоприятный фактор – условия зрительного восприятия, и проведен расчет оптимальной осветительной установки, обеспечивающей нормированное освещение на рабочем месте.

В результате расчетов была разработана осветительная установка, состоящая из двух двухламповых светильников общей потребляемой мощностью 320 Ватт. Фактическое значение минимальной освещенности рабочей поверхности при использовании такой установки составляет 316 лк.

Заключение

В ходе дипломной работы был разработан программный продукт, позволяющий получить численное решение систем ОДУ-ДАУ, а также смоделировать работу электрической цепи, состоящей из заданных элементов.

Конечно, полученная система не может сравниваться ни с программами серии ПА, ни с их аналогами, поскольку разработана в рамках дипломного проекта, максимально упрощена и, скорее, просто призвана доказать концепцию и сделать небольшой шаг в виде реализации модуля ввода данных.

Можно с уверенностью сказать, что использование выбранных средств себя оправдало. Библиотека Qt позволила быстро разработать графические компоненты программы. Использование конфигурационных файлов генераторов лексических и синтаксических анализаторов позволит в перспективе вместо изменения кода сложной структуры, легко модифицировать модуль ввода.

Исходный код части программы, ответственной за ввод данных и обращение к математической библиотеке ляжет в основу дальнейшей разработки в этом направлении.

Список литературы

1. Кнут Д.Э. Искусство программирования, том 2, выпуск 3. М.: Вильямс, 2007. - 800 с.
2. Шлее М. Qt 4.8 Профессиональное программирование на C++. СПб.: БВХ-Петербург, 2012. - 894 с.
3. Д.М. Жук, Маничев В.Б., Папсуев А.Ю. Обобщенный метод моделирования динамики технических систем // Информационные технологии, №8, 2004.
4. Боровков А.И. Компьютерный инжиниринг. СПб.: Изд-во Политехн. ун-та, 2012. - 93 с.
5. Арсеньев В.В. Сажин Ю.В. Методические указания к выполнению организационно-экономической части дипломных проектов по созданию программной продукции. М.: Изд-во МГТУ, 1994. - 52.
6. Документация по Qt 4.8 (<http://qt-project.org/doc/qt-4.8>) (Дата просмотра 01.06.2013)
7. Flex Bison C++ Example (<http://www.stack.nl/~dimitri/doxygen/index.html>) (Дата просмотра 03.04.2013)
8. ГОСТ 2.105-95. Общие требования к текстовым документам.