МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Системы управления аэрокосмической техникой

«Допущен к защите»
Заведующий кафедрой СУАТ
Шимшербаев К.К. к.т.н., профессор ВАК
(Ф.И.О., ученая степень, звание)
_____ «10» 06 2016 г.
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка программного обеспечения наноспутника

Специальность 5В074600 Космическая техника и технологии

Выполнил (а) Скударнова А.В. СВУДА 12-1
(Фамилия и инициалы) группа

Научный руководитель _____ / Полякова Н.И. /
(Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Бекишева А.Н., к.э.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ «30» 05 2016 г.
(подпись)

по безопасности жизнедеятельности:
Жусибаев М.К., профессор, доктор технических наук
(Фамилия и инициалы, ученая степень, звание)
_____ «25» мая 2016 г.
(подпись)

по применению вычислительной техники:
_____
(Фамилия и инициалы, ученая степень, звание)
_____ «____» _____ 20__ г.
(подпись)

Нормоконтролер: Шимшербаев К.К. к.т.н., профессор ВАК
(Фамилия и инициалы, ученая степень, звание)
_____ «10» 06 2016 г.
(подпись)

Рецензент: Турмухамбетов Б.Т. д.т.н.
(Фамилия и инициалы, ученая степень, звание)
_____ «____» _____ 20__ г.
(подпись)

Алматы 2016 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Аэрокосмических и информационных технологий
Специальность 5В074600 Космическая техника и технологии
Кафедра Систем управления аэрокосмической техникой

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Скударнова Анна Валерьевна
(фамилия, имя, отчество)
Тема проекта Разработка программного обеспечения наноспутника

утверждена приказом ректора № 21 от «10» марта 2016 г.
Срок сдачи законченной работы « ___ » _____ 20___ г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:
Введение
1. Специальная часть
1.1 Классификация и краткая характеристика программных обеспечений
1.2 Классификация и краткая характеристика наноспутников
1.3. Разработка программного обеспечения наноспутника.
2. Экономическая часть
2.1 Цели и задачи проекта
2.2 Определение трудоёмкости выполнения НИР
2.3 Расчет затрат на выполнение НИР
2.4 Определение вероятной (договорной) цены НИР

2.5 Оценка социально-экономических результатов выполнения НИР
3. Безопасность жизнедеятельности
3.1 Эвакуация при пожаре
3.2. Расчет времени эвакуации
3.3. Защитное заземление электрооборудования
Заключение
Список используемой литературы

Перечень графического материала (с точным указанием обязательных чертежей):

Основная рекомендуемая литература: Окоцимский Д.Е., Сихарулидзе Ю.Г. Основы механики космического полета: учебное пособие. – М.: Наука. Гл. ред. физ. мат. лит., 1990. – 448 с
Штернфельд А.А. Введение в космонавтику – М.: Наука, 1974. – 240 с

Консультации по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
БЖД	Дюсебаев М.К.	01.02-25.05.16	
Эколог. часть	Бекишева А.У.	20.04-30.05.16	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1.	Введение в космонавтику		
2.	Классификация и краткая хар-ка программного		
	обеспечения		
3.	Классификация и краткая хар-ка наноспутников		
4.	Разработка програм-		
	много обеспечения нсут.		
5.	Экономическая часть		
6.	Основы безопасности		
	жизнедеятельности		
7.	Заключение		

Дата выдачи задания «03» ноября 2015 г.

Заведующий кафедрой _____ Шимырбаев М.К.
(подпись) (Фамилия и инициалы)

Руководитель _____ Помякова И.М.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ Сударикова А.В.
(подпись) (Фамилия и инициалы)

АННОТАЦИЯ

Данный дипломный проект посвящен разработке программного обеспечения, предназначенного для приема - передачи данных с наноспутника. Эту разработку можно использовать в реальных условиях на космических аппаратах.

Дипломный проект состоит из введения, трех разделов (специальная часть, экономическая часть и часть безопасности жизнедеятельности), заключения, списка литературы и приложения.

В работе использовано 10 таблиц, 17 рисунков, 20 источников литературы. Объем дипломного проекта составляет 66 страниц.

АҢДАТПА

Бұл дипломдық жоба наноспутниктегі деректерді беру және қабылдау үшін пайдаланылатын бағдарламамен қамтамасыз етуге арналады. Бұл жобалауды ғарыш кемесінде нақты жағдайларда қолдануға болады.Дипломдық жұмыс жобасы кіріспеден, үш тараудан (арнаулы бөлігі, сондай—ақ денсаулық сақтау қауіпсіздігі және экономикалық бөліктен), қорытындыдан, пайдаланылған әдебиеттерден және бағдарламадан тұрады.Бұл жұмыста 10 таблица, 17 сурет, 20 пайдаланылған әдебиеттер қолданылды.Дипломдық жобаны көлемі 66 беттен турады.

ANNOTATION

This is diploma project devote software nano-satellites which uses for reception - transfer dates from nano-satellite. This is development can uses in really environment in the spacecraft.

Diploma project consists of an introduction, three chapters (special part, economic part, health and safety), conclusion, bibliography and apposition.

There are 10 tables, 17 figures, 20 literature sources in the diploma project. The total volume of the diploma is 66 pages

Содержание

Введение

Свое формирование космонавтика начала в XX веке. Такие ученые, как К.Э.Циолковский, Ю.В.Кондратюк, Р.Х.Годдард, Г.Оберд первыми начали изучать космонавтику и пытались определить, что она может дать людям и как повлияет на них. Идея, соединить два направления деятельности человечества – земной и космической – принадлежит К.Э.Циолковскому. Причем, он не считал освоение космоса альтернативой жизни на Земле, напротив, он считал, что человечество сможет рационально преобразовать природу нашей планеты в пределах Солнечной системы.

В СССР начало развития космоса связано с С.П.Королевым и М.К.Тихонравовым, который в 1945 году организовал группу, которая занималась разработкой проекта пилотируемого двухместного ракетного аппарата, используемого в верхних слоях атмосферы. По решению, проект создавался по типу одноступенчатой жидкостной ракеты, которая была предназначена для полета на высоту до 200 км.

Этот самый проект назывался — ВР-190. Создаваемый космический аппарат должен был решать следующие задачи:

—исследование условий невесомости для человека, который находится в свободном полете;

—исследование движения центра масс кабины космического аппарата и движение ее после того, как она отделится от ракето—носителя;

—изучение верхних слоев атмосферы;

—контроль за работоспособностью приборов, которые входят в конструкцию высотной кабины.

В этом проекте так же были предложены решения, которые до сих пор используются в современных космических аппаратах, например: парашютная система спуска, бескатапультная кабина, содержащая в себе систему обеспечения жизнедеятельности, электроконтактная штанга, предназначенная для упредительного зажигания двигателя мягкой посадки.

Уже множество спутников вращаются вокруг Земли и отправляют данные о других планетах на Землю. Космические аппараты решают множество задач – военные, исследовательские, социально-экономические, научные, прием и передача информации.

Возникшие при подготовке космических кораблей к полетам в космос задачи, стали начальной точкой для развития следующих общественных дисциплин — теоретическая и небесная механика. Разработка нового оборудования и новых математических методов позволили получать решения самых сложных задач при проектировании орбит космических аппаратов и управление полетом. Из этого появилась новая отрасль науки – динамика космического полета.

В 1960-ых годах была повышена точность систем управлений и при посадке на поверхность Луны отклонение от маршрута составляло уже не

несколько десятков километров, а всего 5. Системы управления, которые конструировал Н.А.Пилюгин были одними из лучших в мире.

В 1965 году космические аппараты уже могли передавать на Землю изображения Марса с расстояния, превышающего 200 млн. км, а уже в 1980 году было передано изображение Сатурна с расстояния 1.5 млрд. км.

Наряду с обычными крупно-габаритными спутниками, с развитием технологий, стали разрабатываться наноспутники. Вес наноспутников от 1 до 10 кг. Наноспутники запускают в космос чаще всего в научных и образовательных целях, в интересах образовательных учреждений. Обычно, студенческие наноспутники имеют открытую информацию, которую может получить любой желающий, имеющий специальную приемную аппаратуру и знающий частоту, на которую передают данные на Землю. Управление спутником можно осуществлять с обычного сотового телефона. Ученые считают, что в скором времени наноспутник будет доступен каждому.

Маленькие космические аппараты при изготовлении экономят время (на сбор наноспутника уходит примерно год) и бюджет, что делает их более доступными.

Сегодня наноспутники являются помощниками инженерам и ученым, они обладают большой функциональностью, несмотря на небольшой размер. С их помощью можно проводить тестирование микроэлектроники, изучать Земную атмосферу, следить за передвижением каких—либо объектов (например, морских судов в море).

В наше время развитию космоса уделяют особое внимание, ведь это будущее нашего поколения. Страны, имеющие возможность отправлять аппараты в космос, борются за лидерство в изучении новых планет и в разработке новых устройств, которые будут помогать людям из космоса. Уже были изобретены спутниковая телефонная связь, спутниковое телевидение, мы уже не представляем своей жизни без GPS и интернета, благодаря им мы можем поддерживать связь с людьми, проживающим в далеких от нас странах.

Вопрос об участии Казахстана в разных космических программах ставится довольно давно. На территории нашей страны находится одна из самых больших стартовых площадок в мире – космодром Байконур, который является стратегически важным объектом для сотрудничества Казахстана с Россией. В настоящее время, космодром Байконур находится во владении Российской Федерации и в городе российская администрация, но выбирается она совместным решением президентов двух стран. На сегодняшний день, космодром испытывает повышающийся интерес к себе со стороны многих стран, а также всемирного рынка услуг по отправке космических аппаратов на орбиту. Войдя в шестое десятилетие, космодром Байконур по—прежнему считается самым активным космодромом в мире, он представляет собой огромнейший объект для сотрудничества мирового сообщества в сфере космонавтики.

Первый казахстанский спутник с космодрома Байконур был запущен 18 июня 2006 года и этим был сделан новый шаг в развитие космической эры.

Этот спутник назывался «KazSat» и его функциями были обеспечение эффективного телевещания и мобильной связи. Но в ноябре 2008 года спутник прервал связь с управлением и отправился в бесконтрольный полет, так как из строя вышел комплекс Управляющих двигателей—маховиков. 6 — 13 августа 2009 года спутник KazSat перемещен на орбиту захоронения. Второй Казахстанский спутник был запущен с космодрома Байконур 16 июля 2011 года, он носил название KazSat 2.

Среди стран СНГ собственные спутники имеют только Казахстан и Россия. Освоение космоса занимает важное место в развитии Казахстана. Наша страна должна войти в ряд мировых космических держав. Байконур – первая космическая гавань, с ним связаны наиболее впечатляющие достижения мировой науки и техники.

1 Специальная часть

1.1 Классификация и краткая характеристика программных обеспечений

Программное обеспечение – это взаимодействие частей логической системы, которые управляют работой какого-либо устройства (в данном дипломном проекте – работой наноспутника), выполняет различные действия по обработки информации и расчеты.

Стоит различать программное обеспечение и программу. Программа – это узконаправленный одиночный продукт, функционирующий в информационно-вычислительной среде, а программное обеспечение — это взаимодействие различных программ, объединенных в одно для достижения определенного результата. Разница между ними состоит в объеме выполнения задач.

Разработку программного обеспечения осуществить в шесть этапов:

1) Определение требований и заданий;
2) Проектирование;
3) Основная часть – создание программы;
4) Компоновка;
5) Тестирование;
6) Документирование.

Любой процесс можно описать множеством различных “правильных” последовательностей команд. Значимые характеристики программ иной раз вступают в конфликт друг с другом, в пример можно привести то, как при конструировании летательных аппаратов размеры конфликтуют со скоростью.

Рассматривают программное обеспечение в двух ролях: как исполнителя задания (программное обеспечение используется определенный период), или в роли помощника, когда создается новое дополнительное программное обеспечение (используется в период разработки дополнительного программного обеспечения). Эти роли разные, но одну и ту же аппаратуру

возможно применять для выполнения этих двух разных функций. Так VAX-11 можно представлять как выполнителя операционной системы, который контролирует полет космического аппарата. Одновременно, эта же физическая машина, расположенная в том же самом месте, работающая с теми же самыми параметрами, уже как «инструментальную» машину, которая применяется для сопровождения программ, которые выполняются в то же самое время, когда и VAX-11 производит контроль полета космического аппарата.

Классифицируют три вида программных обеспечений – прикладное, системное и вспомогательное (инструментальное). На рисунке 1.1 представлены виды программного обеспечения.

Рисунок 1.1 — Виды программного обеспечения

Прикладные программы — это программы, которые непосредственно обеспечивают работу устройства: обработка, редактирование, прием и передача информации. Прикладные программы могут нести и общий характер, к примеру, обеспечивать составление и печать документов. Системные программы осуществляют служебные функции, такие как: проверка работоспособности устройства, создание копии информации, выдача справочной информации, управление ресурсами и т.д. К вспомогательному программному обеспечению относятся утилиты и инструментальные системы.

Ключевыми характеристиками программного обеспечения являются: производительность устройства, производительность процессора, частота синхронизации данных, разрядность процессора, объем памяти, плотность записи и скорость обмена информацией.

Производительность устройства – это потенциал компьютера обрабатывать большой объём информации.

Производительность процессора – это количество простых операций, которые выполняются за одну секунду.

Частота синхронизации данных – это количество тактов процессора за время, равное одной секунде. Такт — это такой интервал времени (микросекунды), за который осуществляется простая операция (например, вычитание). Из этого можно сформировать определение, что тактовая частота (частота синхронизации данных) — это количество формируемых импульсов за одну секунду, которые синхронизируют работу узлов устройства. Именно от частоты зависит быстродействие устройства.

Разрядность процессора – это максимальная длина двоичного кода, чтобы он имел возможность передаваться и обрабатываться полностью.

Объем памяти – это максимальный объем памяти, который может храниться в памяти устройства.

Скорость обмена информацией – является скоростью записи либо считывания информации на носитель. Эта скорость определяется быстротой вращения и передвижения этого носителя в устройстве.

Плотность записи представляет из себя объем информации, которую можно записать на единице длины дорожки, бит\мм.

Для того, чтоб создать программное обеспечение для наноспутника, нужно отдельно разработать ПО работы солнечной батареи, системы приема - передачи данных, траектории полета, навигации и функции, которую будет выполнять данный наноспутник.

1.1. Классификация и краткая характеристика наноспутников.

Имеются разные признаки, по которым классифицируются космические аппараты – по назначению полезной нагрузки, по цели пуска, по способу вывода на орбиту, по времени существования и так далее, но есть лишь один признак, который относится к массе и размерам космического аппарата, может рассматриваться как основной. По весовому признаку, все спутники разделяют на такие категорий, как: мини-спутники, вес которых колеблется от 100 до 500 кг; микроспутники, вес которых от 10 до 100 кг; наноспутники имеют вес от 1 до 10 кг, пикоспутники весом от 100 г до 1 кг и фемтоспутники весом до 100 г. Классификация спутников представлена на рисунке 1.2.

По виду управления космические аппараты можно разделить на автоматические и пилотируемые. По виду двигательных установок, которые устанавливают на космические аппараты, делятся на: КА с двигательными установками большой тяги, КА с двигательными установками на ядерном топливе, КА на двигательных установках на химическом (твердом или жидком) топливе. По сферам применения различают КА для ближнего и дальнего космоса. К космическим аппаратам для ближнего космоса можно отнести аппараты, находящиеся недалеко от Земли, например, ИСЗ, населенные обсерватории и орбитальные станции. К космическим аппаратам для дальнего космоса относят межпланетные аппараты, например, десантные и пролетные спутники.

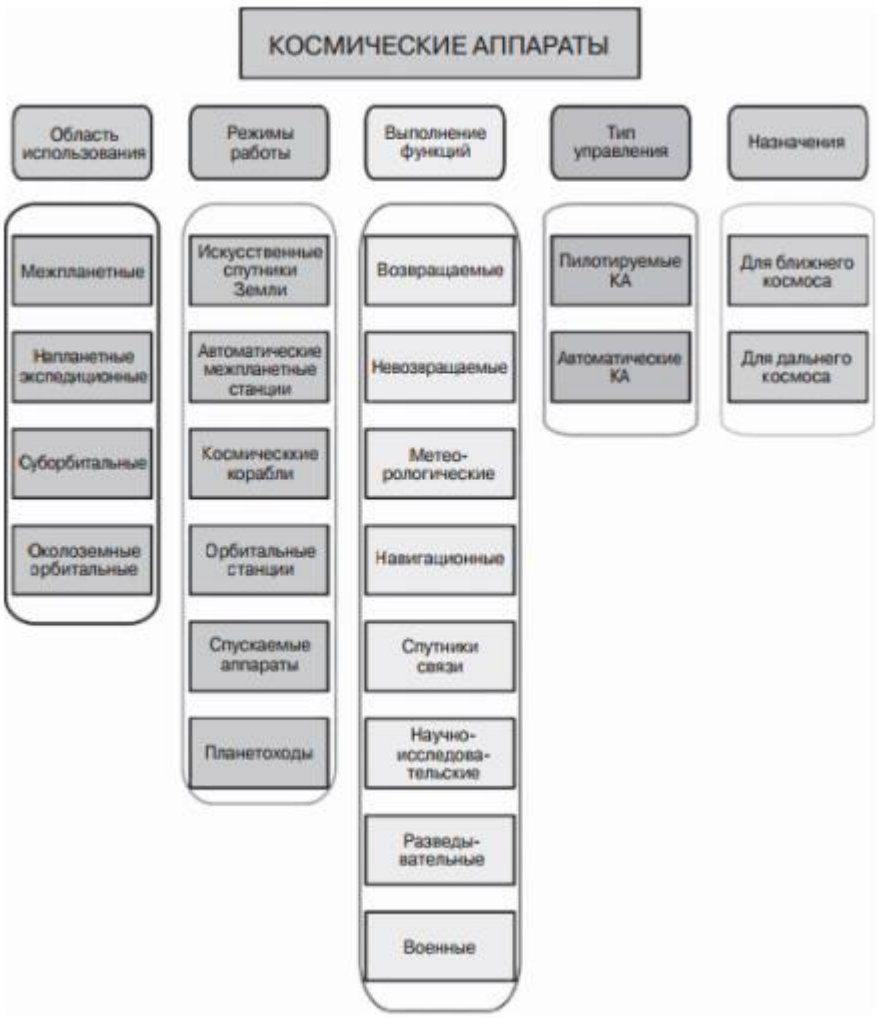

Рисунок 1.2 — Классификация КА

Обретение новых научных знаний и технический прогресс требует от современных космических аппаратов выполнения большего объема работы и получение новых сведений и возможность выполнения новых задач. Поэтому, КА можно классифицировать и по таким признакам, как: назначение, дальность полетов, тип двигательных установок и тип управления. Рассмотрим более детально некоторые из них.

Применение искусственных спутников Земли в народном хозяйстве берет свое начало с образования систем со спутниковой связью. Доработки и изучение новых частот, более ультракоротких радиоволн, которые принимаются только в зоне открытой видимости, привели к необходимости создавать дороге радиобашни и сети трансляционных станций. При удалении абонентов друг от друга более, чем на 500 км становится, наиболее выгодна, с точки зрения экономики — космическая связь, если сравнивать ее с кабельными и радиорелейными линиями связи. Используя для этих целей ИСЗ можно охватывать огромные расстояния. Стационарные спутники, находящиеся на высоте 35 000 км и застыв там, позволяют передавать связь без помех.

Высочайшая эффективность и четкость информации, передаваемой из космоса, позволяет искусственным спутникам Земли за время для одного витка — 1,5 ч — получать данные о метеоусловиях, которые в 100 раз точнее данных, поступающих от всех метеостанций, находящихся на Земле. Так же, метеорологические ИСЗ передают данные о состоянии солнечной активности и об измерениях теплового излучения Земли. Эти данные позволяют делать краткосрочные и долгосрочные прогнозы погоды.

Функционирование навигационных космических аппаратов производится с помощью особой наземной системы снабжения и они работают для осуществления навигации самолетов и кораблей. Для любого навигационного спутника известны те данные, которые необходимы для перемещения по орбите, но тем не менее, в зависимости от небесных тел они имеют возможность выполнять измерения в любое время дня и ночи и в любую погоду, это происходит в связи с тем, что их координаты ориентируются с поддержкою частотного радиомаркера. Эталоном навигационных ИСЗ являются спутники «Транзит» и «Космос—1000», остальные искусственные спутники употребляются в океанографии для получения большого объема информации о том, что происходит в морях-океанах, например — температура воды, атмосферы, скорость ветра, парообразование воды, размещение плавающих льдов и айсбергов, миграция рыб и многое другое.

Наноспутник – это искусственный спутник Земли, вес которого не превышает 10 кг. В современном мире наноспутники играют значимую роль для выполнения разного рода задач. Несмотря на небольшой размер, они выполняют важные вещи для изменений или наблюдений из космоса. Маленькие космические аппараты считаются незаменимой альтернативой больших космических средств, которые, за счет больших размеров и из—за большого количества аппаратуры на борту имеют большую стоимость создания и трудности запуска. Даже самый легкий ракетоноситель имеет возможность захватывать на орбиту сразу несколько наноспутников. Так же, если сравнивать время, которое затрачивается на создание наноспутника и обычного спутника, победу одерживает маленький спутник, время на сборку которого колеблется от 9 месяцев до года. Соотвестственно, исходя из вышесказанного, можно сделать вывод, что снижается риск финансовых потерь в случае гибели наноспутникапо причине аварии или неправильном выведении на орбиту. Соответственно, невысокая стоимость наноспутника обусловлена тремя основными факторами: недорогие запчасти, дешевая рабочая сила (чаще всего наноспутники собираются студентами) и бюджетный, иногда бесплатный, вывод на орбиту. Недостатком наноспутника из—за малых размеров и веса, является малая масса полезной нагрузки.

В работоспособности наноспутника в узкой зоне, большую роль играет способ крепления панелей солнечных батарей на корпусе. Как известно, действенным способом уменьшения размеров солнечной батареи, при его расположении на ракете-носителе, является использование складывающихся панелей, но в этом случае существенно усложняется устройство самих панелей,

повышается их вес и уменьшается надежность раскрытия панелей. В связи с этим, для наноспутников предпочитают использовать нераскрываемые панели солнечных батарей. Они располагаются на боковых поверхностях корпуса спутника.

Рассмотрим подробно устройство наноспутника, представленного на рисунке 1.3. Корпус наноспутника обозначен цифрой 1, он соединен с платой 2, которая является торцевой частью наноспутника. На ней находятся узлы, которые соединяют его с системой отделения (на рисунке они не показаны).

На корпусе расположена аппаратура: целевая и служебная. К этой аппаратуре относятся: блок электромагнитных механизмов 3, камера для произведения съемки Земной поверзности 4, передатчик с антенной, который передает фотографии поверхности Земли в центр управления 5, 6 бортовой комплекс управления и прочая аппаратура, которая на рисунке не обозначена.

На корпусе установлен ложемент 7, который закреплен шарнирными узлами 8. Ось отворота ложемента, расположенной относительно торца корпуса наноспутника, расположена параллельно. На ложементе установлены: панель с солнечной батареей 9, магнитометры 10, солнечный датчик 11. Поворот ложемента относительно корпуса и его фиксирование происходит с помощью винтового механизма 12 винт-гайка, который установлен в кронштейне 13 корпуса наноспутника и в кронштейне 14 ложемента с таким же винтовым механизмом 15. Корпус и ложемент выполнены таким образом, что винтовые механизмы обеспечивают поворот и фиксирование ложемента по отношению к опорной поверхности платы (относительно торца корпуса наноспутника) на угол меньше девяноста градусов. Благодаря тому, что панели солнечной батареи размещены именно так, обеспечивается повышенная плотность расположения деталей во всем наноспутнике, в главном обтекателе ракеты-носителя.

Радиосвязь с наноспутником поддерживается благодаря расположению на нем гибких ленточных антен 16.

При попутном запуске вместе с другим спутником, наноспутник крепится на основном спутнике 17, закрепленном в месте выведения полезной нагрузки ракеты—носителя 18. Основной и попутный спутник на участке выведения помещаются под главным аэродинамическим обтекателем (на рисунке 1.4 обтекатель не изображен) в зоне полезного груза 19. В данном образце, область монтажа наноспутника ограничевается конической частью в зоне полезного груза и штангой гравитационного устройства 20 головного спутника. Расположение камеры снизу на наноспутнике дает наиболее полную возможность использования зоны монтажа. Кроме того, освобождается район поворота панели солнечной батареи относительно корпуса наноспутника.

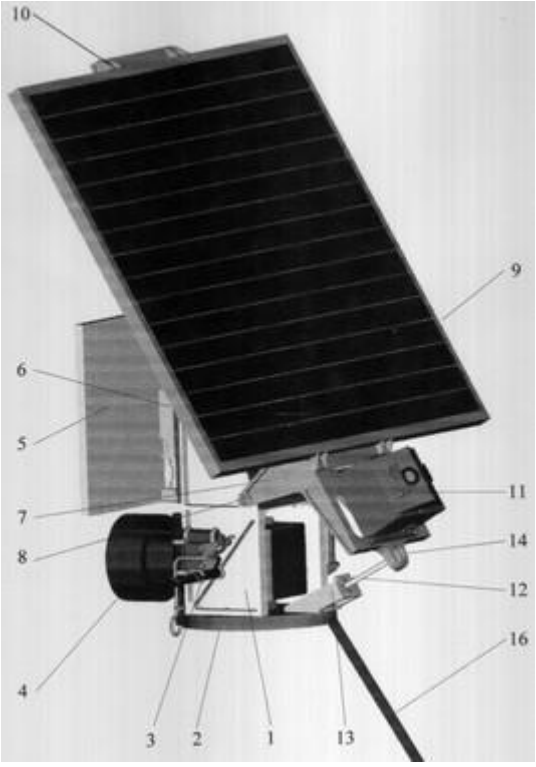

Рисунок 1.3 — Общий вид наноспутника

Устанавливается наноспутник с помощью адаптера 21, расположенного сверху основного спутника.

Устанавливается наноспутник с помощью системы отделения 22, обеспечивающей ориентацию наноспутника в направлении, в котором он отделяется. В таком случае, угол, расположенный между продольной осью наноспутника и основного спутника, должен обеспечивать безаварийное отделение наноспутника. В виду того, что панель солнечной батареи выполнена поворотной, это дает возможность оптимизировать расположение наноспутника по параметрам для безаварийности его отделения от носителя и действенной площади солнечной батареи для установленного запуска.

На системе отделения наноспутник располагается так, чтобы панель солнечной батареи была расположена около зоны полезного груза. Это обеспечивает самый большой угол наклона наноспутника от штанги гравитационного устройства головного спутника.

Для того, чтобы адаптировать наноспутник к основному спутнику при его разработке, нужно проделать следующие основные работы:

— определить зоны монтажа наноспутника и граничных условий;

— определить угол безаварийного отделения наноспутника (угол можно изменить варьированием усилий толкателей системы отделения) ;

— определить угол наклона панелей, чтобы обеспечить безаварийное отделение наноспутника;

— определить зависимость угла наклона панели солнечной батареи от действенной площади солнечной батареи для определенных и рассчитываемых параметров орбиты запуска;

— найти проектно—конструкторские параметры наноспутника опираясь на решение задачи по уменьшению суммарной массы наноспутника и средств его адаптации (в этом случае, системы отделения) при существующих ограничениях по зоне монтажа;

— ввести в процесс уменьшения массы дополнительные к спутнику—прототипу переменные и нижнее местоположение камеры на корпусе , что позволяет уменьшить суммарную массу наноспутника и средств его адаптации.

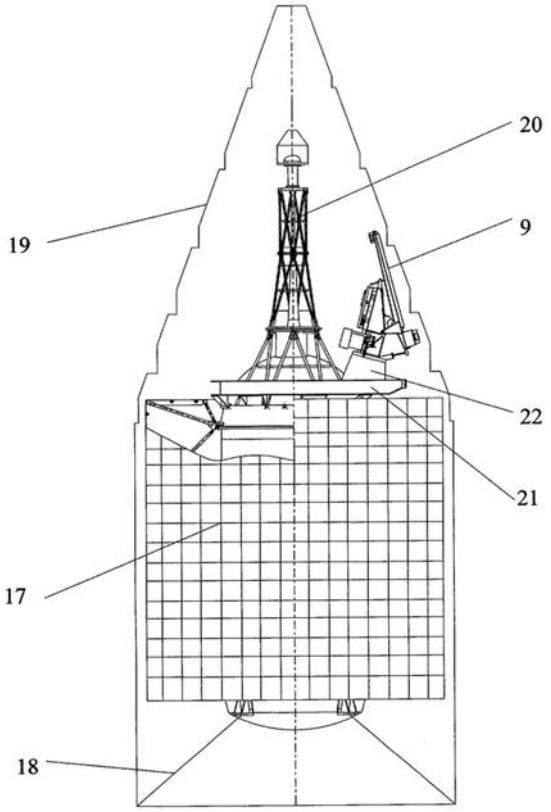

Рисунок 1.4 — Расположение наноспутника на основном спутнике при попутном запуске

Наноспутники применяют для выполнение следующих задач:

—астрономическое наблюдение;

—наблюдение за поверхностью Земли;

—изучение ближнего космоса;

—поисково—спасательные операции;

—правоохранительная деятельность (антитеррористическая, антикриминальная);

—эксперименты в области нано—технологий.

Первый наноспутник был выведен на орбиту в феврале 1958 года вместе с основным спутником Explorer-1. После того, как основной спутник отдалился на безопасное расстояние, от ракетоносителя в определенной последовательности отделились наноспутники с помощью пружинных толкателей.

Изучение малых космических аппаратов развивается в быстрых темпах, добавляются свежие подходы к проектированию новых малых космических аппаратов, появляются новые возможности попутного запуска аппаратов, уменьшают размеры спутников, усовершенствуют их возможности и т.п. Общая структура наноспутников, имеющихся в наши дни, изображена на рисунке 1.5.

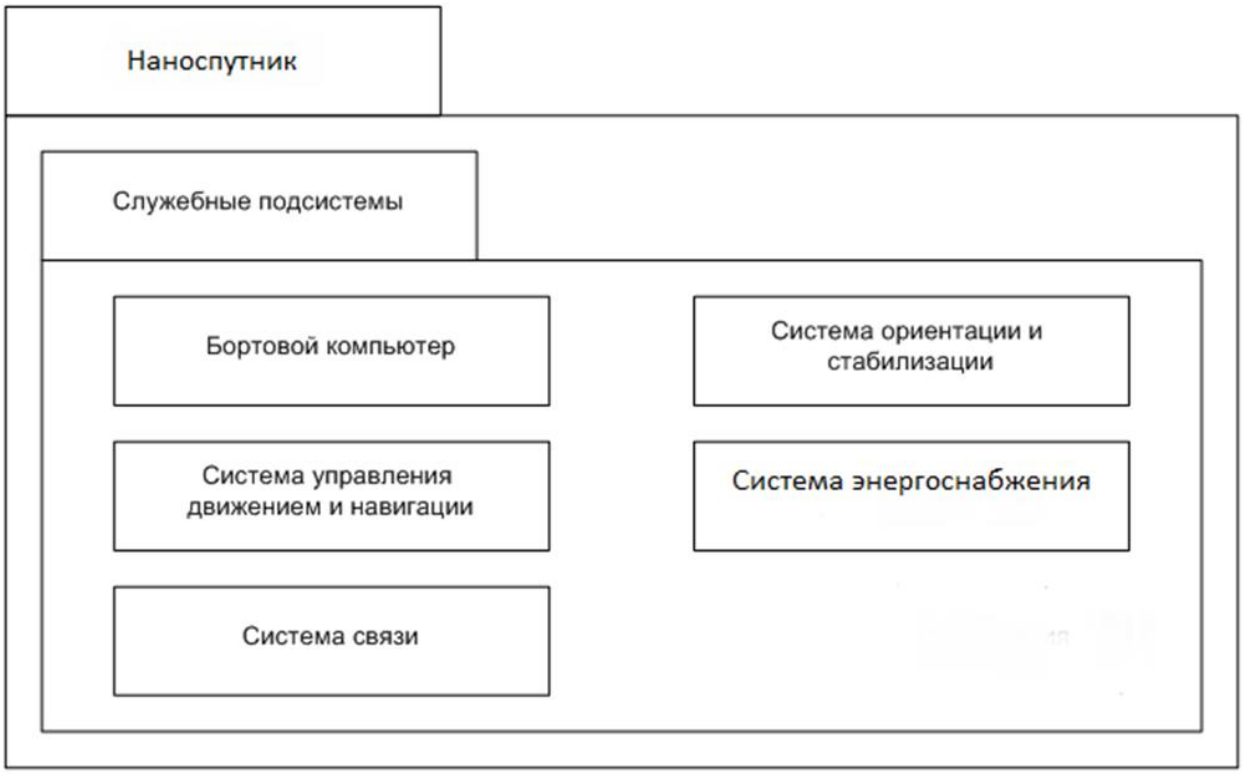

Рисунок 1.5 — Структурная схема аппаратного обеспечения наноспутника

На рисунке 1.6 можно увидеть количество наноспутников, которые вывели или собираются вывести на орбиту по данным NASA. Самое большое количество запусков ожидается в 2016 году. Зеленым цветом отмечены удачные запуски, оранжевым цветом отмечены запуски с неудачей, синим и голубым цветом на графике отмечены будущие планируемые запуски.

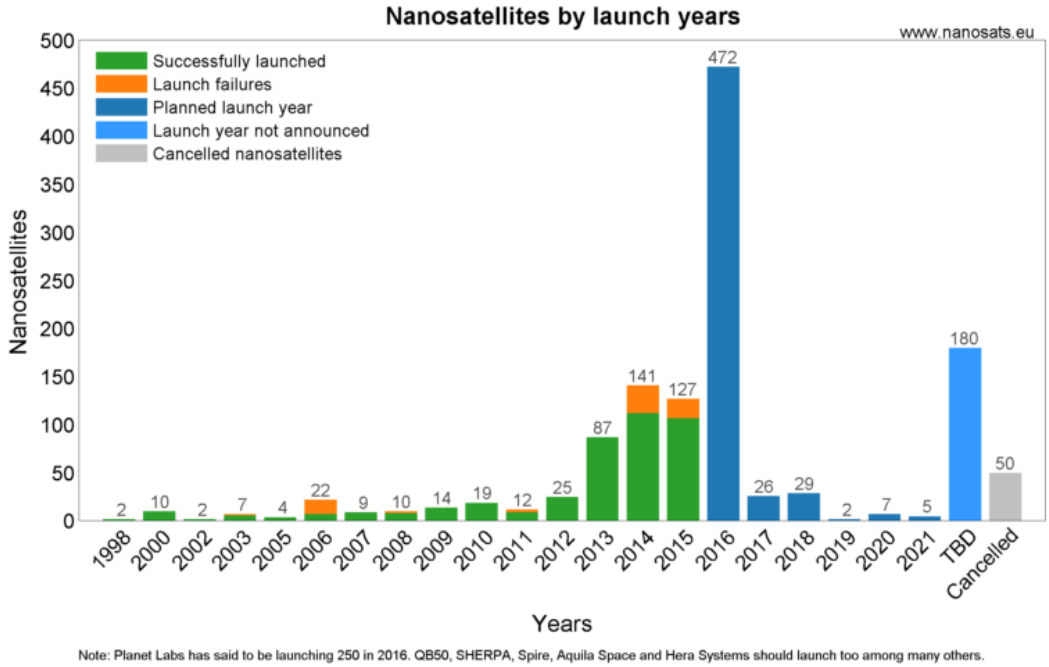

Рисунок 1.6 — Количество запусков наноспутников в разные годы

1.3 Разработка программного обеспечения наноспутника

Как говорилось выше, для того, чтобы создать полное программное обеспечение для наноспутника, нужно создать множество различных ПО. В данной части работы будет рассмотрено программное обеспечение для приема и передачи данных с наноспутника.

Качество космической радиосвязи зависит от множества факторов, таких как дальности, размеров антенн, мощности передатчика, длины волны, поглощения сигнала, качества техники приема-передачи, шумов, помех и скорости движения КА. Принцип работы радиосвязи состоит из того, что колебание тока в антенне передатчика образовывает в пространстве электромагнитные волны, эти волны, двигаясь со скоростью света, доходят до антенны в приемнике и возбуждают в антенне переменный ток; этот ток очень слаб, но тем не менее, если выполнить настройку приемника точно в резонанс с частотой радиоволны, то даже небольшое ее воздействие может раскачать в антенне вполне различимые колебания. Затем эти колебания усиливают, их анализируют и извлекают из них переданную информацию.

В данной части работы будет рассмотрено программное обеспечение для приема-передачи информации с наноспутника по радиоканалу. Для данной работы возьмем отдельное устройство, которое будет прикреплено к конструкции космического аппарата. Это может быть использовано, например, в такой ситуации, когда судно, находится в рейсе и порт приписки передает информацию с наноспутника о приближающейся буре.

Модем в подобном случае будет отличаться по функциям от модема, который работает с телефонной линией, так как тут нельзя дозвониться до

абонента, так же, здесь не допустима дуплексная связь. В этом случае, функции дозвона и другие осуществляются через блок приёма-передачи данных по радиоканалу. В таком случае, модем, только ждет получение сигнала, после, производит его демодуляцию, образовывая при этом цифровой код, далее, он передает его в компьютер. Если говорить о передаче информации, то модем выполняет прием цифрового кода, потом модулирует его, преобразуя в аналоговый сигнал, после чего, производит его передачу в блок передачи данных по радиоканалу.

На данный момент, технология, используемая для выпуска интегральных микросхем и микроконтроллеров, находится на высочайшем уровне. Технологии постоянно совершенствуются и изобретаются новые усовершенствованные виды микрочипов. Так, одним из видов микрочипов, применяемых в наше время, является DSP — digital signal processor (англ.) (цифровой сигнальный процессор). Этот процессор является идеальным для обрабатывания сигналов. Он имеет встроенный язык программирования, благодаря чему, процессор дает возможность настраивать его на любую работу, которая необходима. Почти во всех современных модемах, несмотря на его назначение, устанавливаются DSP.

В данной части дипломного проекта, мы будем разрабатывать программное обеспечение для подобного устройства, которое будет выполнять прием-передачу данных по радиоканалу, при этом выполняя кодирование-декодирование информации, используя для этих целей — цифровой сигнальный процессор (DSP).

1.3.1 Способы кодирования и декодирования информации

Для выбора нужного нам пути разработки ПО, сначала рассмотрим современные способы и средства для кодирования-декодирования данных.

Все модемы, независимо от их назначения, модулируют сигналы для передачи по телефонным и радиоканалам, но сигнал может быть промодулирован различными способами.

Модуляция сигнала – это модификация одного или сразу нескольких параметров несущего синусоидального колебания (фазы, амплитуды, частоты) отталкиваясь от значений двоичной информации, которая передается источником. Рассмотрим, какие существуют виды модуляции.

Фазовая модуляция – это такая модуляция, при которой фаза несущей получает только постоянные (не меняющиеся) значения из ряда возможных значений (к примеру: 0 , 90 , 180 град.), а информация складывается в перемены фазы несущего колебания. При этом наборе допустимых фаз, каждой перемене фазы соответствует определенное значение дибита — двух последовательных битов информации.

Для увеличения пропускной способности в амплитудно-фазовой модуляции, используются одновременно манипуляции сразу двух из параметров несущего колебания — фазы и амплитуды. Каждый из возможных

элементов модулированного сигнала, описывается значением амплитуды и фазы. Чтобы в дальнейшем увеличить скорость передачи, количество точек пространства модулированного сигнала повышается в число раз, кратное двум. Один из видов амплитудно-фазовой модуляции — 16-ти позиционная квадратурная АМ, подобная модуляция используется в дуплексных модемах.

В модемах так же применяется частотная манипуляция. При подобной манипуляции, каждому из значений бита информации («1» — «0») соответствует специальная частота синусоидального сигнала.

Модуляция с минимальным сдвигом (MSK) — это продолжение модуляции частотной. При MSK, разница между частотами «1» и «0» (по модулю) всегда будет равна половине от скорости передачи данных.

GMSK (Gaussian Minimum Shift Keying) — это гауссовская манипуляция, существующая с минимальным частотным сдвигом. Манипуляцию называют «гауссовской», в виду того, что порядок информационных битов до модулятора осуществляется через фильтр нижних частот (ФНЧ) с характеристикой Гаусса. Из-за этого значительно уменьшаются полосы частот.

В данной работе будем использовать модуляцию частотную.

1.3.2 Структурная схема устройства приема-передачи данных с наноспутника

Структурная схема устройства для приема-передачи данных с нано-спутника представлена на рисунке 1.7.

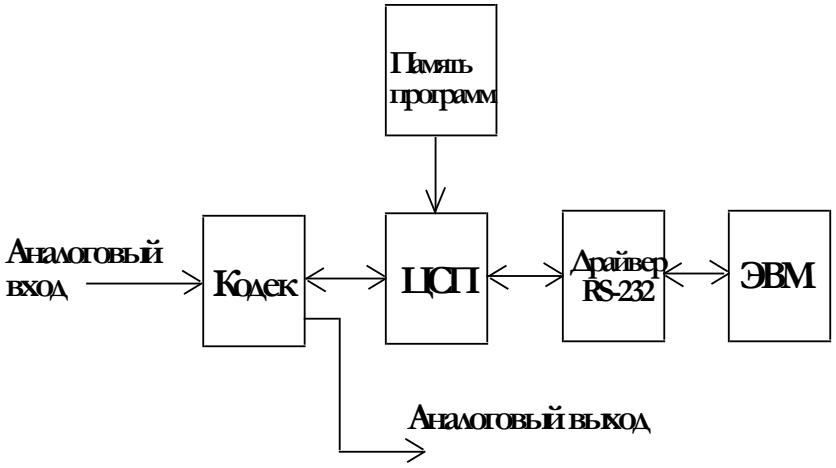

Рисунок 1.7. – структурная схема устройства приема-передачи информации

Принятый по радиоканалу сигнал подается на вход устройства. После того, как сигнал попадает на выход устройства, он посылается в аналогово—цифровой кодек. Кодек — это высокоскоростной чип, в который последовательным портом встроены аналого-цифровые и цифро-аналоговые

преобразователи; последовательный порт программируется интервальным таймером, регистрами и др.

Так как в данной работе мы будем использовать частотную модуляцию и частоту «1» = 1615 Hz и «0» = 1785 Hz. то, будет достаточно 16—ти разрядного кодека, с частотой дискретизации 8 kHz.

После того, как аналоговый сигнал поступит, кодек переделывает его в обычный цифровой код, передающийся по последовательному порту в цифровой сигнальный процессор — DSP, после чего, по определенному алгоритму он изменяется и поступает на ЭВМ. Когда цифровой код производит прием информации от DSP по своему порту, кодек преображает его в аналоговый сигнал, далее, он передает его на выход, после сигнал попадает на передатчик по радиоканалу и перенаправляется на другое подобное устройство.

Главная часть нашего устройства кодирования-декодирования данных — digital signal processor (DSP). Существует множество разных фирм, занимающихся производством DSP: Texas Instruments, Motorola, Analog Devices, Philips и др. В данной работе мы будем работать с цифровым сигнальным процессором, разрабатываемом фирмой — Analog Devices (а именно с моделью ADSP – 2181).

Краткая характеристика ADSP - 2181:

• 16Кх24 бит Program Memory;
• 16Кх16 бит Data Memory;
• два программируемых последовательных скоростных порта;
• 1 таймер интервала;
• 16-битный порт для прямого доступа к IDMA;
• 8-битный порт для прямого доступа к BDMA (объем до 4 Мб);
• адресное пространство устройства ввода-вывода (2048 адресов);
• 4 сегмента по 8К слов внешней оверлейной памяти;
• внешние прерывания, а так же, программируемые выводы флагов;
• режим сниженного энергопотребления (Рпот.< 1 мВт/час);
• производительность процессора — 33.33 MIPS;
• имеется отдельный порт внутрисхемной эмуляции.

На рисунке 1.8 графически изображен цифровой сигнальный процессор— ADSP–2181.

Память программ ADSP-2181 заключает в себе 16Кх24 ОЗУ программ на кристалле; возможность выполнять до двух обращений в каждом цикле дает возможность память программ, при этом, все операции могут быть завершены за 1 цикл.

Память данных ADSP-2181 содержит в себе 16,352 16-разрядных слова во внутренней памяти данных.

Пространство байтовой памяти является двунаправленным, 8-разрядным, внешним пространством памяти, которое используется для того, чтобы хранить программы и данные. Доступ к байтовой памяти реализовывается через BDMA. Пространство байтовой памяти включает в себя 256 страниц. Все эти страницы

имеют размер 16Кх8, это дает возможность использовать до 4Мх8 (т.е. 32Мбит) ПЗУ или ОЗУ без добавочной логики. Все обращения к байтовой памяти содержат временные параметры, которые определяются регистром BMWAIT.

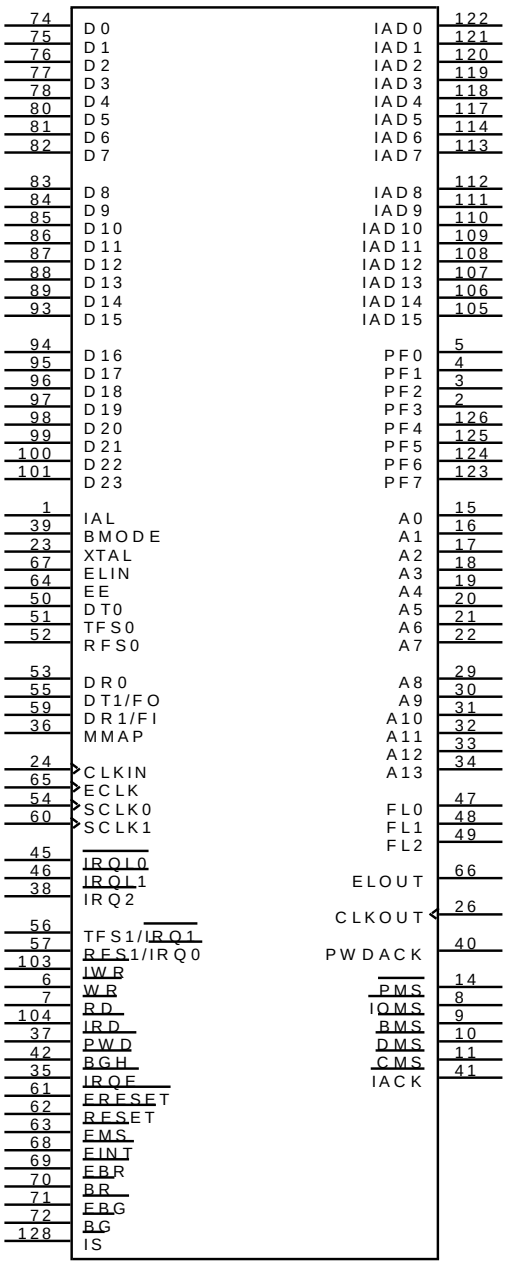

Рисунок 1.8 — Графическое изображение ADSP – 2181 процессора

Контроллер для прямого доступа в байтовую память называется – BDMA. Он дает возможность загружать и сохранять команды программы и данных, и использовать при этом пространство байтовой памяти. Схема BDMA имеет возможность обращения к пространству байтовой памяти, тогда как процессор функционирует и обхватывает всего один цикл DSP для того, чтобы переместить 8-, 16- или 24-разрядные слова.

IAD0—IAD15 представляют из себя 16-ти разрядную мультиплексированную шину данных и адреса порта IDMA (порт прямого доступа к внутренней памяти).

IDMA процессора ADSP-2181 на сегодняшний день одно из новых устройств, глобально упрощающих создание интерфейса с HOST-процессором. Интерфейс работы IDMA порта с HOST-процессором ихображен на рисунке 1.9.

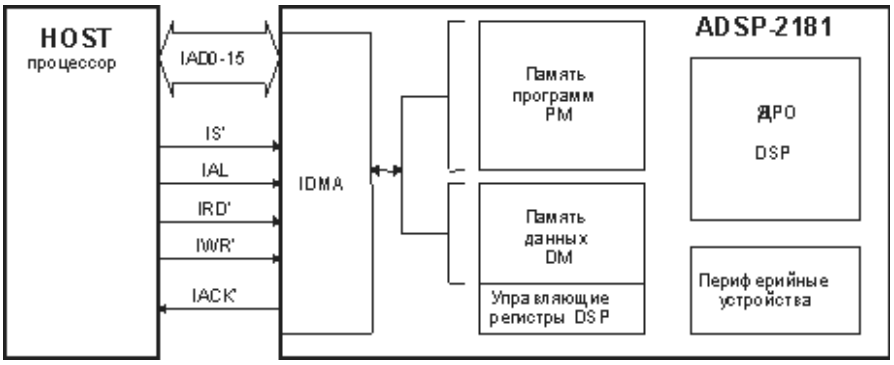

Рисунок 1.9 — Интерфейс работы IDMA порта с HOST-процессором

4 входа управления IDMA нужны для:

IS – нахождения и выбора порта;

IAL — для записывания адреса ячейки памяти;

IRD — для чтения данных через порт;

IWR – для записывания информации ;

IACK – сигнал для того, чтобы подтверждать доступ. Он обусловливает окончание операций чтения или записи, а так же, готовность IDMA к следующим операциям.

Выводы процессора BMODE и MMAP обусловливают порядок загрузки, и отвечают за распределение памяти DSP. Для произведения загрузки через внешнюю память — BMODE должно быть равно 0 и MMAP так же равно 0. Загрузка же состоит из:

• Сброса процессора сигналом RESET

• Загрузки в Data Memory и Programm Memory кодов программы, а так же, данных, исключая ячейку PM(0x0000).

• Записи слова в ячейку PM(0x0000), для того, чтобы запустить загруженную ранее программу.

Аппаратными входами прерываний являются: IRQ2, IRQL1, IRQL0 и IRQE. TFS1/IRQ1, RFS1/IRQ0. Когда на них подается низкий уровень сигнала, управление переходит к соответствующей подпрограмме. Таблица 1.1 является таблицей прерываний.

Рассмотрим подробнее источники прерываний:

RESET – если будет получен низкий уровень сигнала, управление передастся подпрограмме, которая производит инициализацию DSP. Тут же совершается вторичная загрузка программы в DSP из внешней памяти.

PWD — кнопка power down.

Таблица 1.1 – Таблица прерываний

Источник прерывания	**Адрес вектора прерывания (HEX)**
Reset (or Power-Up with PUCR = 1)	0000 (*Highest Priority*)
Power Down (Nonmaskable)	002C
$\overline{IRQ2}$	0004
$\overline{IRQL1}$	0008
$\overline{IRQL0}$	000C
SPORT0 Transmit	0010
SPORT0 Receive	0014
$\overline{IRQE}$	0018
BDMA Interrupt	001C
SPORT1 Transmit or $\overline{IRQ1}$	0020
SPORT1 Receive or $\overline{IRQ0}$	0024
Timer	0028 (*Lowest Priority*)

XTAL, CLKIN. На данные источники производится подача тактовой частоты от кварца. В нашем случае эта величина составляет 16,67 MHz.

BMS, PMS, DMS, IOMS, CMS — это выводы, необходимые для подключения оверлейной памяти и управления ею.

Имеется возможность для использования оверлейной памяти, как памяти данных.

Шина адреса ADSP-2181 состоит из 14 разрядов, это является достаточно маленькой величиной, потому, для расширения адресного пространства оверлейной памяти, используют такие флаги, как — FL0, FL1, и FL2 или PMS, в зависимости от требуемых конфигураций.

Системный интерфейс ADSP 2181 изображен на рисунке 1.10.

Для выполнения последующих действий, выбираем кодак, исходя из требований. Остановим выбор так же, на микросхемах, изготавливаемых фирмой — Analog Devices. Отталкиваясь от того, что в роли цифрового сигнального процессора был избран ADSP 2181, то выбираем звуковой кодек — AD1847, с последовательным цифровым интерфейсом. Этот кодек совместимый с ADSP 21xx. Кодек обладает 16-ти разрядным последовательным портом, его будет достаточно для того, чтобы выполнить наше устройство кодирования-декодирования информации.

Микросхема памяти с ультрафиолетовым стиранием предназначена для того, чтобы хранить в ней программы под цифровой сигнальный процессор. Эти программы должны будут осуществлять алгоритм кодирования-декодирования.

ADSP -2181 может работать не больше, чем с четырьмя Мбайтами внешней памяти. Выбранный цифровой сигнальный процессор — ADSP-2181 обладает расширенной системой команд, и сохраняет в памяти огромное

количество оперативной информации. Исходя из этого, только для его точной работы нужен большой объем памяти. В виду того, что мы разрабатываем большую программу по кодированию-декодированию информации, инициализации DSP, кодека, и по организации способа частотной модуляции, то добавочно к этому, нам нужно еще не меньше 500 Kb памяти. В общей сумме, необходимо не меньше 600 Kb памяти.

Выбераем микросхему с наибольшим объёмом памяти —1 Мб. Эта микросхема — EPROM AM27C080.

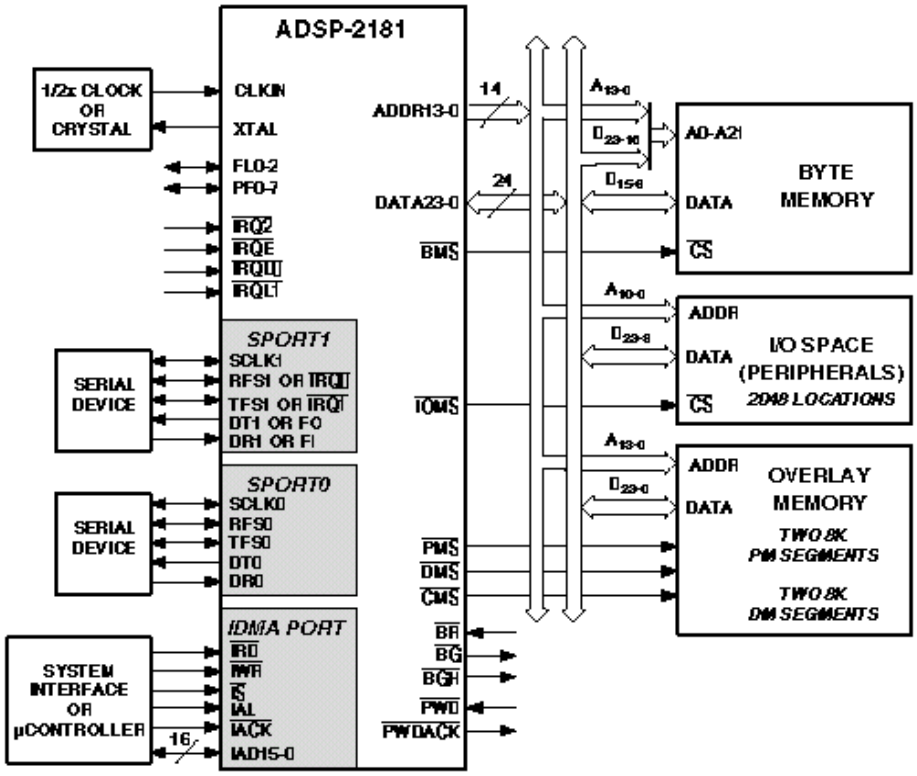

Рисунок 1.10 — Системный интерфейс ADSP 2181

1.3.3 Блок инициализации

В состав устройства приема-передачи данных с наноспутника входят программируемые микросхемы DSP и Codec. Для каждой из этих микросхем нужно выполнить инициализацию внутренних переменных, регистров и флагов. Кроме того , данное устройство работает с внешним host—компьютером-терминалом и для того, чтобы он начал работать, ему необходимо дать команду инициализации. Для того, чтобы начать работу с терминалом, так же нужно инициализировать переменные и регистры, используемые интерфейсом приема-передачи данных. Для кодирования информации используется структура судового телеграфа NBDP (narrowband printing) узкополосное буквопечатанье. Для того, чтобы инициализировать данную структуру, воспользуемся таблицей соответствия букв и цифр

определенной комбинации 1 и 0. На рисунке 1.11 представлен блок инициализации.

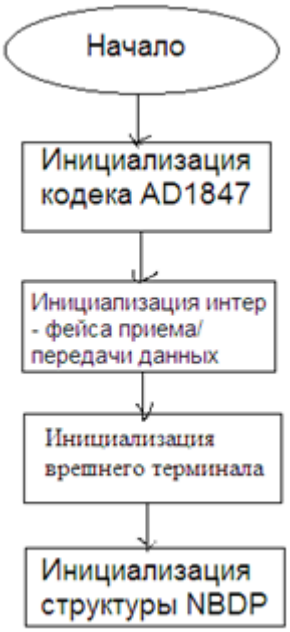

Рисунок 1.11 – Блок инициализации

1.3.4 Интерфейс приема / передачи данных

Для того, чтобы организовать работу с терминалом, используется набор команд, которые передаются по последовательному порту компьютера. Программа под терминал в данной работе рассматриваться не будет, но связь с ней с помощью нашего устройства будет показана и использована.

Когда в host-компьютере появляются данные, которые нужно принять, терминал дает запрос на передачу. Так же, когда появляются данные в нашем устройстве, которые необходимо передать, в терминал посылается запрос на прием им данных. Если терминал готов принять данные , устанавливается флаг готовности передачи в терминал. Если же устройство готово к приему данных от терминала, так же устанавливается флаг готовности приема от терминала. Блок—схема запроса на прием и передачу данных от терминала представлена на рисунке 1.12.

В случае установки флагов приема — передачи, происходит соответственно прием или передача данных.

При приеме данных от терминала выполняется кодирование по алгоритму NBDP и модулирование, используя для этого частотную модуляцию (1 — 1615 Hz , 0 — 1785 Hz), с последующей передачей на кодек и после цифро-аналогового преобразования на выход.

На рисунке 1.13 представлена блок-схема алгоритма приема данных от терминала.

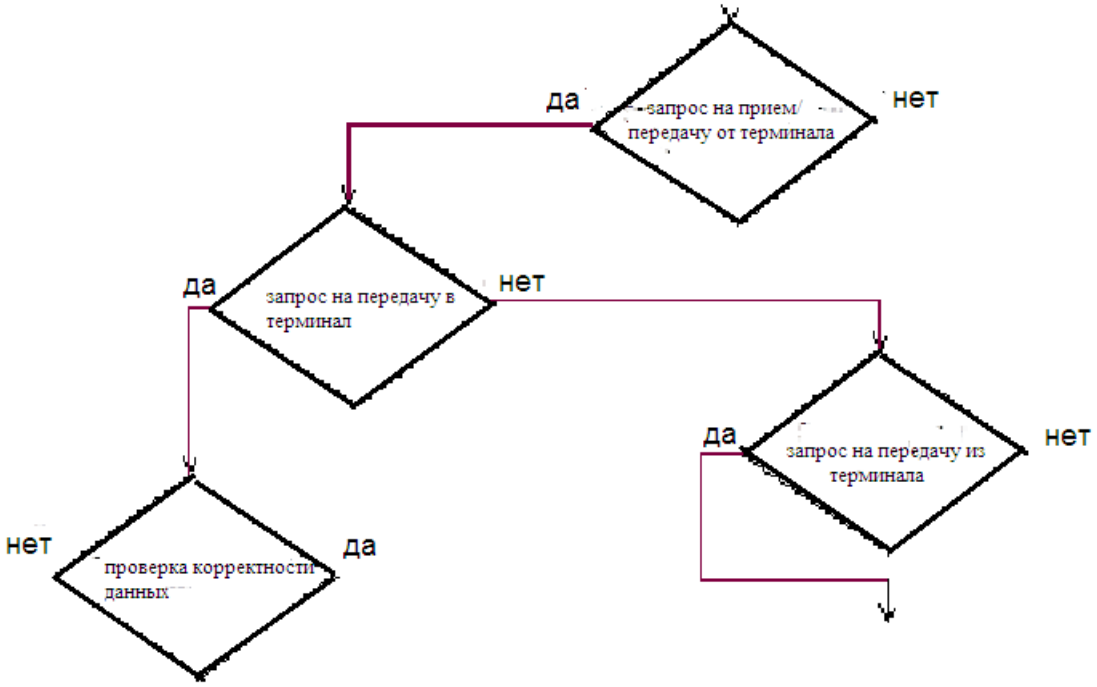

Рисунок 1.12 — Блок - схема запроса на прием - передачу данных от терминала

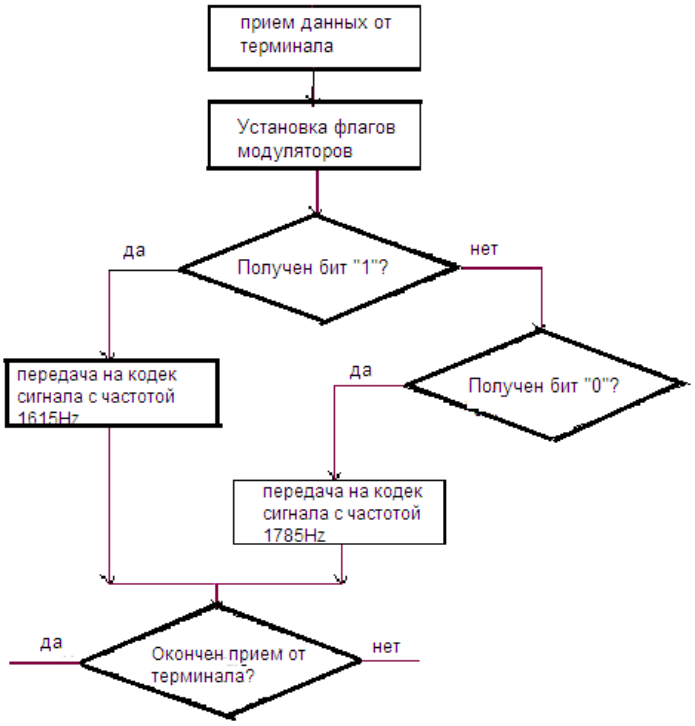

Рисунок 1.13 – прием данных от терминала

При передаче данных в терминал, выполняется демодуляция данных, которые поступили, они раскодировываются по алгоритму NBDP и переводятся в формат ASCII, после чего передаются по последовательному порту в

терминал. Блок-схема алгоритма передачи данных в терминал представлена на рисунке 1.14.

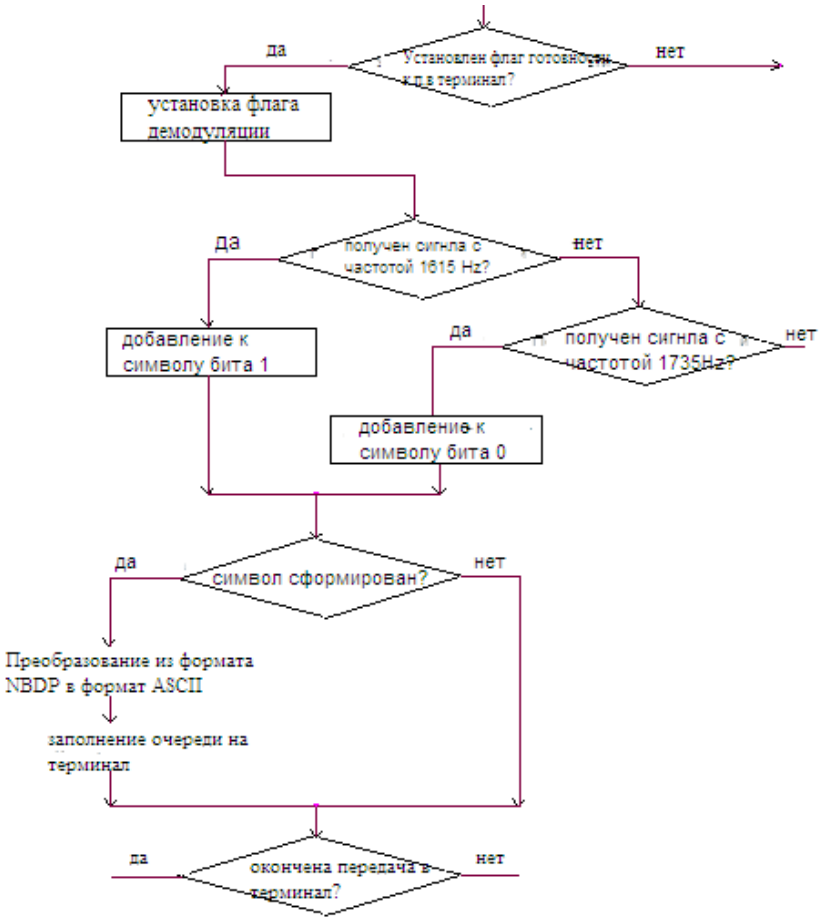

Рисунок 1.14 – передача данных в терминал

После того, как прием или передачи данных окончены, в терминал поступает команда — «устройство свободно». Это разрешает последующие запросы для обмена данными. При нажатии кнопки «RESET» программа перезагружается в память и происходит инициализация.

Алгоритм, представленный выше, является общим принципом работы программы; более подробная его реализация, с учетом особенностей выбранной технологической базы, будет рассмотрена далее.

1.3.5 Разработка программного обеспечения

Программное обеспечение будет разработано для цифрового сигнального процессора — Analog Devices «ADSP -2181». Для Разработки ПО будет использоваться программное обеспечение фирмы «Analog Devices», которое позволяет создавать и тестировать программы под DSP.

Разработка программного обеспечения будет составляться на языке Assambler под ADSP - 2181 и на языке Си. Язык Си напрямую не поддерживается DSP. Поэтому, в комплекте программ фирмы Analog Devices в

комплекте идет конвертер программ написанных на языке Си, в формат машинных кодов DSP. Так же, имеется возможность создавать проекты, включающие программы, написанные под DSP и под Си, с сылками на функции друг друга с помощью внешних процедур и переменных.

Для создания ПО и программ, будем использовать следующие инструменты:

Assembler — компоновщик программ Assambler *.dsp в *.obj
Linker — компилятор кода DSP
ADSP—2181 Simulator — эмулятор для ADSP - 2181
Minitor – программа, позволяющая совершать записи программы в DSP
C Compiler – компилятор для языка Си
C Runtime Libraries — библиотеки для языка Си
C Debugger — отладчик для программ, написанных на Си
Cconverter — конвертер программ, написанных на Си в код DSP
Разработанный программный код находится ниже.

1.3.6 Главный модуль программы

```
main.c
//Подключение заголовочных файлов и объявление локальных и
глобальных переменных//
#include <drivers.h>
#include <nbdp.h>
#include <string.h>
#include <ctype.h>
int PTTRecal;
int DebugLevel;
int cntr;
int CALLID[10];
//Объявляются внешние функции и переменные//
extern void FreeSignal(void);
extern int IsCharReady(void);
extern int FREFLAG;
extern int setISS,setIRS;
void main()
{
int savedsw;
// Сброс выводов FL2—FL0//
asm(«reset fl0; »);
asm(«reset fl1; »);
asm(«reset fl2; »);
init1847(); //Происходит инициализация кодека AD1847//
UARTInit(); //Происходит инициализация интерфейса UART//
//Даём команду host устройству на инициализацию://
```

```
SendCommandHost(«INIT»,«POWER ON»);
//Происходит инициализация структуры NBDP — narrowband direct printing//
NBDP_Init();
//Далее происходит процесс обмена данными//
while(1)
{ if(COMF & 1)
{ if(COMF & 0x02) arq(); //запускается протокол ARQ , описание в npdp_arq.c//
else if(COMF & 0x1C) fec(); //запускается протокол FEC , описание в npdp_fec.c//
else //Если другое состояние устройства COMF, то ошибка записывается в host//
{ SendHostError(5,COMF); /* COMF ERROR */
StandBy();}}
if(FREFLAG) //В случае, если модем свободен//
{ FREFLAG=0; // Команда о том, что модем свободен, отправляется в host //
SendCommandHost(«FRE»,FREFLAG == 1 ? «1»: «0»);}
/* Если символ готов к передаче, выбраются команды из порта*/
if(IsCharReady()) SerialDriver();
// Происходит передача символов в HOST//
if((COMF & 1) && (hocount))
{ char block[15],d;
int i=0;
while(hoRead(&d) != -1)
{block[i++] = d;
if(i>14) break;}
block[i]=0;
SendCommandHost(«TXT»,block); //блок отправляется в host//
}
} // заканчивается while//
} // заканчивается main()//

int CheckCommRequest(char *DATA) // Проверяется корректность номера объекта связи (например, плавательного судна), по которому происходит связь с данным объектом//
{int j=0,tmp;
for(j=0;DATA[j]!=';';)
{tmp = DATA[j];
if(!(tmp>='0' && tmp<='9')) // цифры не имеют возможности быть кодом объекта//
{SendHostError(11,tmp); /* ERR;01.1;ID WRONG DIGIT (%c),ID[j] */
return 1;}
```

```
CALLID[j] = tmp;
j++;
if(j>9) break;}
CALLID[j]=0;
if(j!=4 && j!=5 && j!=9) /* incorrect ID */ //символ ';' в коде должен располагаться третьим, четвертым или же восьмым//
{SendHostError(12,j);
return 1;}
return 0; //проверка прошла успешна, возврат//
}

//здесь указан массив названий команд для функции SendCommandHost(char *cmd, ...)//
char *cmds[] =
{«TXT», «ARQ»,«CFE»,«SFE»,«SET», /* 0... 4 */
«STA»,«INI»,«DEB»,«DBG»,«DIA», /* 5... 9 */
«FRE»,«STB»,«TST»,«STR»,«FCC», /* 10...14 */
NULL};
int CheckCommRequest(char *DATA);
```

1.3.7 Программа модуляции

За основу представленного ниже способа модуляции взята — частотная модуляция. В данной модуляции используется протокол морского телеграфа NBDP — узкополосное буквопечатанье. В основе лежит таблица кодовых значений символов, представленных в виде комбинации 1 и 0. После, эти символы модулируются с соответственно частотами 1615 Hz и 1785 Hz. (таблица символов в файле nbdptable.c).

Mod.c

```
//Подключение заголовочных файлов и объявление локальных и глобальных переменных//
#include <drivers.h>
#include <template.h>
#include <nbdp.h>
/*#define OUTKOEFF 0x6000*/
#ifdef RXTOTX_
extern int RXTOTX[2];
#endif
int PHACC,TMPPHACC;
int PHINC;
int modready=1;
int BITTIME=1;
```

```
int bitsleft;
int outdata;
volatile int lout,rout;
extern int FDIV,FSPTTOFF;
int dbgcntr;
/* MODULATOR */
extern void Timing(void);
void modulator(void)
{
// Так же, возможна работа в тестовом режиме//
if(SelfTest) goto testmodes;
FDIV++;
//через 10 мс отсчеты обнуляются, и посылаются в хост синхронизирующего сигнала цикла ARQ//
if(FDIV==80) {FDIV=0;NBDPTHRTX();}
#ifdef RXTOTX
txbuf[1] = RXTOTX[0];
txbuf[2] = RXTOTX[1];
return;
#endif
// Проверяется, нужна ли модуляция, если не нужна, то возврат//
if(!ModulatorON)
{PHACC = 0;
return;}
restart:
if(modready)
{txbuf[1] = txbuf[2] = 0;
if(FSPTTOFF) // выключается модуляция//
{FSPTTOFF=0;
PTTOFF();}
return;}
testmodes:
BITTIME——;
if(BITTIME==0)
{/*=== determine the new bit from out byte ===*/
// Включение режима реальной работы//
if(!SelfTest) outdata <<= 1; /* 7 bit mode */
if(outdata & 0x80) /* MARK */
{PHINC = MARKINC;
asm(«
#define PFDATA 0x3fe5
ar = b#0000000001000000; /* set 1 PF6/TLG OUT */
ay1 = dm(PFDATA);
ar = ar or ay1;
```

```
dm(PFDATA) = ar; »);}
else /* SPACE */
{PHINC = SPACEINC;
asm(«
ar = b#1111111110111111; /* reset 1 PF6/TLG OUT */
ay1 = dm(PFDATA);
ar = ar and ay1;
dm(PFDATA) = ar; »);}
// Включается тестовый режим, работа в 8—битном режиме//
if(SelfTest) out_data <<= 1;
BITTIME = BITLENGTH;
bitsleft——;
if(bits_left<0)
{mod_ready=1;
modulating();
goto restart;}
else
{/* blink by PF7 as syncro_out */
asm(«
ar = dm(PFDATA);
ar = tglbit 7 of ar;
dm(PFDATA) = ar; »);}}
/*asm («dis m_mode; »);*/
PHACC += PHINC;
lout = sini(PHACC);
txbuf[2] = lout; // выдаются результаты в порт (происходит связь с кодеком)//
}

void modulating(void)
{if(SelfTest)
{if(SelfTest==1) /* space */
{outdata = 0;}
else if(SelfTest==2) /* mark */
{outdata = 0xFF;}
else if(SelfTest==3) /* dot */
{/* 10101010 */
outdata = 0xAA;}
else if(SelfTest==4) /* big dot */
{/* 11110000 */
outdata = 0xF0;}
else
{SelfTest=0;}
BITTIME=1;
```

```
bitsleft=8; //8—битный режим работы//
modready=0;
return;}
// В тестовом режиме дальше не идем//
if(modready == 0) return;
BITTIME=1;
if(ModulatorGet(&outdata)==0) /* nothing to get */
{
//обнуляются флаги и возврат//
/* clear PF6 as TLG & PF7 as syncroout */
asm(«
ar = dm(PFDATA);
ar = clrbit 7 of ar;
ar = clrbit 6 of ar;
dm(PFDATA) = ar; »);
return;}
bits_left=7; //7—битный режим работы//
modready=0;}
```

1.3.8 Программа демодуляции

Принцип демодуляции входного сигнала устройства приема и передачи данных с наноспутника основывается на известных нам частотах, поступающих от отдельного устройства приема сигнала по радиоканалу которые уже отфильтрованные и поданные на вход устройства. Необходимо выделить соответственно частоты 1785 Hz = MARK = ' 0 ' и 1615 Hz = SPACE = ' 1 ' и получить из этого определенный код. После того, как код получен, происходит раскодирование по таблице NBDP и передача в ЭВМ, где программа TERMINAL реагирует на данные кода.

```
// Demod.c
// Подключаем заголовочные файлы и объявляем локальные и
глобальные переменные//
#include <drivers.h>
#include <stdlib.h>
#include <template.h>
#include <nbdp.h>
extern void outmcr(int data);
#define DMDKOEFF 0x019A
#define DMDLEVEL 0x1000 /* порог срабатывания */
#define OUTMARK 'M'
#define OUTSPACE 'S'
#define OUTERR 'E'
```

```
int myfir(int NewValue);
int mysqrt(int NewValue);

volatile int lin,rin;
int DemodulatorON=1; //включить/выключить флаги демодулятора//
int PHTONEACC[2]; //опорный MARK/SPACE PHASE ACC//
int PHTONEINC[2] = {MARKINC, SPACEINC};
int i,j; //счетчики времени//
int R [4]; //результаты времени//
int S [4]; //частичные суммы//
int DL[4*BITLENGTH]; //Удалить  линии для 4x//
int DLp = 0; //DL точки//
int countN=BITLENGTH; //количество циклов//
int REZ[2]; // Итоги//
int PRZLT[2]; // Приблизительные итоги (prev. Rez)//
int svMode; /* Save multiplier mode location */
int JitterOff; /* Bit syncro OFF flag */
int OutData[2];
void demodulator(void)
{
// Если демодулятор выключен, то происходит обнуление результатов
и возврат//
if(!DemodulatorON && (dipsw & DIPSW1))
{REZ[0] = 0;
outmcr(0);
goto CheckCycle;}
if(SelfTest) // Если включен тестовый режим, то возврат//
{return;}
//Отключение режима целочисленной арифметики//
asm(«dis mmode; »);
lin = rxbuf[INCHNL];
// ограничение входного сигнала для устранения переполнения
фильтра//
rin = FRACTMULTIPLY(rin,DMDKOEFF);
//Заполнение массива временных результатов для MARK и SPACE//
PHTONEACC[0] += PHTONEINC[0];
R[0] = FRACTMULTIPLY(rin,sini(PHTONEACC[0]));
R[1] = FRACTMULTIPLY(rin,cosi(PHTONEACC[0]));
PH_TONEACC[1] += PHTONEINC[1];
R[2] = FRACTMULTIPLY(rin,sini(PHTONEACC[1]));
R[3] = FRACTMULTIPLY(rin,cosi(PHTONEACC[1]));
//Извлечение старых данных и добавление новых//
for(i=0;i<4;i++)
{S[i] = S[i] — DL[DLp];
```

```
S[i] = S[i] + R[i];
DL[DLp++] = R[i];}
if(DLp >= (4*BITLENGTH)) DLp=0; /* wrap DL pointer */
// Получение результатов по каждому фильтру//
PRZLT[0] = REZ[0]; // Предварительные результаты//
REZ[0]=FRACTMULTIPLY(S[0],S[0]) + FRACTMULTIPLY(S[1],S[1]);
REZ[1]=FRACTMULTIPLY(S[2],S[2]) + FRACTMULTIPLY(S[3],S[3]);
if(dipsw & DIPSW2)
{REZ[0] = mysqrt(REZ[0]);
REZ[1] = mysqrt(REZ[1]);}
R[1] = (REZ[0]—REZ[1]);
R[2] = (REZ[0]+REZ[1]);
R[0] = FRACTMULTIPLY((FRACTDIVIDE(R[1],R[2])),0x6400);
REZ[0] = R[0];
/* debug solution output */
txbuf[1] = REZ[0];
```

Шаг 4.
Контроль времени, тактовый детектор, и т.д.

```
CheckCycle:
countN——;
if(!countN)
{if(abs(REZ[0]) > DMDLEVEL)
{if(REZ[0] > 0)
{OutData[0] = OUTMARK;
asm(«"reset fl0; set fl2; set fl1; »);}
else
{OutData[0] = OUTSPACE;
asm(« set fl0; reset fl2; set fl1; »);}}
else
{OutData[0] = OUTERR;
asm(« set fl0; set fl2; reset fl1; »);}
countN = BITLENGTH;
//Отправление на ЦАП (кодек)//
NBDPTHRRX(OutData[0]);}
// включается режим целочисленной арифметики//
asm(«ena mmode; »);}
```

1.4 Дополнительные программы и функции

1.4.1 Программа nbdptable.c

Эта программа содержит в себе таблицу соответствия кодов ASCII и NBDP.

```
#include <nbdp.h>
/*B — MARK FREQ:1785Hz
/*Y — SPACE FREQ:1615Hz
/* ¦ +————————————————————————————¦ 5—Unit ¦ 7—Unit ¦ */
/* ¦No ¦LET ¦FIG ¦RUS ¦ Code ¦ Code ¦ */
UCHAR  7bitcode[]=  {  /*+——+————————+————————+————————
+————————————————+————————————————¦ */
Ox71 ,/*¦01¦A 41 ¦— 2D ¦A 80 ¦ZZAAA ¦BBBYYYB 71 ¦ */
Ox27 ,/*¦02¦B 42 ¦? 3F ¦Б 81 ¦ZAAZZ ¦YBYYBBB 27 ¦ */
Ox5c ,/*¦03¦C 43 ¦: 3A ¦Ц 96 ¦AZZZA ¦BYBBBYY 5C ¦ */
Ox65 ,/*¦04¦D 44 ¦Wh?? ¦Д 84 ¦ZAAZA ¦BBYYBYB 65 ¦ */
Ox35 ,/*¦05¦E 45 ¦3 33 ¦E 85 ¦ZAAAA ¦YBBYBYB 35 ¦ */
Ox6c ,/*¦06¦F 46 ¦Э*9D ¦Ф 94 ¦ZAZZA ¦BBYBBYY 6C ¦ */
Ox56 ,/*¦07¦G 47 ¦Ш*98 ¦Г 83 ¦AZAZZ ¦BYBYBBY 56 ¦ */
Ox4b ,/*¦08¦H 48 ¦Щ*99 ¦X 95 ¦AAZAZ ¦BYYBYBB 4B ¦ */
Ox59 ,/*¦09¦I 49 ¦8 38 ¦И 88 ¦AZZAA ¦BYBBYYB 59 ¦ */
Ox74 ,/*¦10¦J 4A ¦Ю*07 ¦Й 89 ¦ZZAZA ¦BBBYBYY 74 ¦ */
Ox3c ,/*¦11¦K 4B ¦( 28 ¦K 8A ¦ZZZZA ¦YBBBBYY 3C ¦ */
Ox53 ,/*¦12¦L 4C ¦) 29 ¦Л 8B ¦AZAAZ ¦BYBYYBB 53 ¦ */
Ox4e ,/*¦13¦M 4D ¦. 2E ¦M 8C ¦AAZZZ ¦BYYBBBY 4E ¦ */
Ox4d ,/*¦14¦N 4E ¦, 2C ¦H 8D ¦AAZZA ¦BYYBBYB 4D ¦ */
Ox47 ,/*¦15¦O 4F ¦9 39 ¦O 8E ¦AAAZZ ¦BYYYBBB 47 ¦ */
Ox5a ,/*¦16¦P 50 ¦0 30 ¦П 8F ¦AZZAZ ¦BYBBYBY 5A ¦ */
Ox3a ,/*¦17¦Q 51 ¦1 31 ¦Я 9F ¦ZZZAZ ¦YBBBYBY 3A ¦ */
Ox55 ,/*¦18¦R 52 ¦4Ч34 ¦P 90 ¦AZAZA ¦BYBYBYB 55 ¦ */
Ox69 ,/*¦19¦S 53 ¦' 27 ¦C 91 ¦ZAZAA ¦BBYBYYB 69 ¦ */
Ox17 ,/*¦20¦T 54 ¦5 35 ¦T 92 ¦AAAAZ ¦YYBYBBB 17 ¦ */
Ox39 ,/*¦21¦U 55 ¦7 37 ¦У 93 ¦ZZZAA ¦YBBBYYB 39 ¦ */
Ox1e ,/*¦22¦V 56 ¦= 3D ¦Ж 86 ¦AZZZZ ¦YYBBBBY 1E ¦ */
Ox72 ,/*¦23¦W 57 ¦2 32 ¦B 82 ¦ZZAAZ ¦BBBYYBY 72 ¦ */
Ox2e ,/*¦24¦X 58 ¦/ 2F ¦Ь 9C ¦ZAZZZ ¦YBYBBBY 2E ¦ */
Ox6a ,/*¦25¦Y 59 ¦6 36 ¦Ы 9B ¦ZAZAZ ¦BBYBYBY 6A ¦ */
Ox63 ,/*¦26¦Z 5A ¦+ 2B ¦З 87 ¦ZAAAZ ¦BBYYYBB 63 ¦ */
Ox0f ,/*¦27¦CR 0D —— ———————— ¦AAAZA ¦YYYBBBB 0F ¦ */
Ox1b ,/*¦28¦LF 0A —— ———————— ¦AZAAA ¦YYBBYBB 1B ¦ */
Ox2d ,/*¦29¦LET 16 —— ———————— ¦ZZZZZ ¦YBYBBYB 2D ¦ */
Ox36 ,/*¦30¦FIG 0F —— ———————— ¦ZZAZZ ¦YBBYBBY 36 ¦ */
Ox1d ,/*¦31¦SP 20 —— ———————— ¦AAZAA ¦YYBBBYB 1D ¦ */
Ox2b ,/*¦32¦RUS 00 —— ———————— ¦AAAAA ¦YBYBYBB 2B ¦ */
```

```
#ifdef RUSSIAN // структура данных, отлаженная под русский алфавит//
struct CVT CVT_TABLE[] =
{/*¦ +————————————————————————————¦ 5—Unit ¦ 7—Unit ¦ */
/*¦ No ¦LET ¦ FIG ¦RUS ¦ Code ¦ Code ¦ */
/*+————+————————+————————+————————————————
+————————————————————————————————¦ */
{0x41,0x2D,'a' },/*¦01¦A 41 ¦— 2D ¦A 80 ¦ZZAAA ¦BBBYYYB 71 ¦ */
{0x42,0x3F,'b' },/*¦02¦B 42 ¦? 3F ¦Б 81 ¦ZAAZZ ¦YBYYBBB 27 ¦ */
{0x43,0x3A,'c' },/*¦03¦C 43 ¦: 3A ¦Ц 96 ¦AZZZA ¦BYBBBYY 5C ¦ */
{0x44,0x03,'d' },/*¦04¦D 44 ¦Wh?? ¦Д 84 ¦ZAAZA ¦BBYYBYB 65 ¦ */
{0x45,0x33,'e' },/*¦05¦E 45 ¦3 33 ¦E 85 ¦ZAAAA ¦YBBYBYB 35 ¦ */
{0x46,'|' ,'f' },/*¦06¦F 46 ¦Э*9D ¦Ф 94 ¦ZAZZA ¦BBYBBYY 6C ¦ */
{0x47,'{' ,'g' },/*¦07¦G 47 ¦Ш*98 ¦Г 83 ¦AZAZZ ¦BYBYBBY 56 ¦ */
{0x48,'}' ,'h' },/*¦08¦H 48 ¦Щ*99 ¦X 95 ¦AAZAZ ¦BYYBYBB 4B ¦ */
{0x49,0x38,'i' },/*¦09¦I 49 ¦8 38 ¦И 88 ¦AZZAA ¦BYBBYYB 59 ¦ */
{0x4A,'`' ,'j' },/*¦10¦J 4A ¦Ю*07 ¦Й 89 ¦ZZAZA ¦BBBYBYY 74 ¦ */
{0x4B,0x28,'k' },/*¦11¦K 4B ¦( 28 ¦К 8A ¦ZZZZA ¦YBBBBYY 3C ¦ */
{0x4C,0x29,'l' },/*¦12¦L 4C ¦) 29 ¦Л 8B ¦AZAAZ ¦BYBYYBB 53 ¦ */
{0x4D,0x2E,'m' },/*¦13¦M 4D ¦. 2E ¦M 8C ¦AAZZZ ¦BYYBBBY 4E ¦ */
{0x4E,0x2C,'n' },/*¦14¦N 4E ¦, 2C ¦H 8D ¦AAZZA ¦BYYBBYB 4D ¦ */
{0x4F,0x39,'o' },/*¦15¦O 4F ¦9 39 ¦O 8E ¦AAAZZ ¦BYYYBBB 47 ¦ */
{0x50,0x30,'p' },/*¦16¦P 50 ¦0 30 ¦П 8F ¦AZZAZ ¦BYBBYBY 5A ¦ */
{0x51,0x31,'q' },/*¦17¦Q 51 ¦1 31 ¦Я 9F ¦ZZZAZ ¦YBBBYBY 3A ¦ */
{0x52,0x34,'r' },/*¦18¦R 52 ¦4Ч34 ¦P 90 ¦AZAZA ¦BYBYBYB 55 ¦ */
{0x53,0x27,'s' },/*¦19¦S 53 ¦' 27 ¦C 91 ¦ZAZAA ¦BBYBYYB 69 ¦ */
{0x54,0x35,'t' },/*¦20¦T 54 ¦5 35 ¦T 92 ¦AAAAZ ¦YYBYBBB 17 ¦ */
{0x55,0x37,'u' },/*¦21¦U 55 ¦7 37 ¦У 93 ¦ZZZAA ¦YBBBYYB 39 ¦ */
{0x56,0x3D,'v' },/*¦22¦V 56 ¦= 3D ¦Ж 86 ¦AZZZZ ¦YYBBBBY 1E ¦ */
{0x57,0x32,'w' },/*¦23¦W 57 ¦2 32 ¦B 82 ¦ZZAAZ ¦BBBYYBY 72 ¦ */
{0x58,0x2F,'x' },/*¦24¦X 58 ¦/ 2F ¦Ь 9C ¦ZAZZZ ¦YBYBBBY 2E ¦ */
{0x59,0x36,'y' },/*¦25¦Y 59 ¦6 36 ¦Ы 9B ¦ZAZAZ ¦BBYBYBY 6A ¦ */
{0x5A,0x2B,'z' },/*¦26¦Z 5A ¦+ 2B ¦З 87 ¦ZAAAZ ¦BBYYYBB 63 ¦ */
{0x0D,0x0D,0x0D},/*¦27¦CR 0D —— ———————— ¦AAAZA ¦YYYBBBB 0F ¦ */
{0x0A,0x0A,0x0A},/*¦28¦LF 0A —— ———————— ¦AZAAA ¦YYBBYBB 1B ¦ */
{0x01,0x01,0x01},/*¦29¦LET 16 —— ———————— ¦ZZZZZ ¦YBYBBYB 2D ¦ */
{0x02,0x02,0x02},/*¦30¦FIG 0F —— ———————— ¦ZZAZZ ¦YBBYBBY 36 ¦ */
{0x20,0x20,0x20},/*¦31¦SP 20 —— ———————— ¦AAZAA ¦YYBBBYB 1D ¦ */
```

{OxO3,OxO3,OxO3},/*¦32¦RUS OO —— ———— ¦AAAAA ¦YBYBYBB 2B ¦ */

1.4.2 Программа nbdp.c

Эта программа содержит в себе функции для работы с протоколом NBDP и объявление всех глобальных переменных, которые используются в обмене данными по протоколу NBDP.

```
#include <nbdp.h>
#include <drivers.h>
#include <string.h>
#include <ctype.h>
#include <template.h>

UINT No; /* Channel No */
UCHAR COMF;
/* b0 : O=stanby 1=communication */
/* b1 : 1=ARQ Mode */
/* b2 : 1=CFEC Mode */
/* b3 : 1=SFEC Mode */
/* b4 : 1=GFEC Mode */
/* b5 : O=send 1=recv */
/* b6 : 1 = SLAVE wait PHASING, O = SLAVE PHASED */
/* b7 : */
UCHAR STATE;
UCHAR LAST_STATE;
/* FEC ARQ */
/* 0 stand—by stand—by */
/* 1 set receive mode send before process */
/* 2 syncro signal rx receive before process */
/* 3 FEC msg rcv 7id—phasing—master */
/* 4 Synchronous signal send 7id—rephasing—master */
/* 5 ID send 4id—phasing—master */
/* 6 FEC msg send 4id—rephasing—master */
/* 7 stop signal send 7id—phasing—slave */
/* 8 7id—rephasing—slave */
/* 9 4id—phasing—slave */
/* 10 4id—rephasing—slave */
/* 11 iss */
/* 12 irs */
UCHAR arqf; /* ARQ flag */
/* b0 : 0 = slave, 1=master */
/* b1 : 1 = WRU */
```

```
/* b2 : 1 = OVER FLAG */
/* b3 : */
/* b4 : */
UCHAR SelfTest;
UCHAR FirstIdleBlock;
UCHAR SendFCCInfo;
UCHAR IDLEAcnt; /* IDLE ALFA для FEC */
UINT RxErr,RxOK; /* Полученные Err и OK символы */
UCHAR SHIFT; /* LET=0,FIG=1,RUS=2 */
/*UCHAR PTT;*/ /* —BK intenal bit */
UCHAR WRU_SUPPORT=1;
UCHAR DX[5]; /* ввод DX LINE для FEC */
UCHAR RX[5]; /* ввод RX LINE для FEC */
UCHAR IDIN[7]; /* ввод SFEC ID SHIFT REG */
UCHAR SFSIS4[4]; /* SFEC идентификация сигнала для 4—digit ID */
UCHAR SFSIS5[4]; /* SFEC идентификация сигнала для 5—digit ID */
UCHAR SFSIS9[7]; /* SFEC идентификация сигнала для 9—digit ID */
UCHAR SCID[7]; /* SFEC название станции  ID */
UINT n[10]; /* ARQ(и другие) наблюдение */
UCHAR connector;
UCHAR CB[3][3]; /* запрос блока во время загрузки */
/* и для пользователя ARQ */
UCHAR CIB[3][3]; /* блок идентификации ID для 7—digit
синхронизирующих следующие форматы */
UCHAR CCS[3]; /*во время связи ARQ */
/* и во время поэтапно осуществляющейся работы */
UCHAR CKERR; /* ошибки CK во время фазировки*/
/*UCHAR RCB[3][3];*/ /* REPHASING: запросить загрузку блоков*/
UCHAR RDATA[3]; /* получить блок данных */
UCHAR SLICEBUF[2];
UCHAR LASTCS;
UCHAR LASTBLOCK[3];/* последний загруженный блок */
UCHAR LASTBLNO;
UCHAR LASTID[10]; /* последние попытки коммуникации ID */
UCHAR RQ; /* повторить запрос */
UCHAR RPTF; /* повторить флаг */
UCHAR IDTYPE; /* 4— или 7—DIGIT ID тип. 0—4,1—7 */
UCHAR WRU_CNT;
UCHAR strAAB[21]; /* не преобразованный  AAB */
UCHAR AAB[AABLEN];/* преобразовать AAB */
UCHAR SIS4[6]; /* идентификационный сигнал для 4—digit ID */
UCHAR SIS5[6]; /* идентификационный сигнал для 5—digit ID */
UCHAR SIS9[9]; /* идентификационный сигнал для 9—digit ID */
UCHAR SCS9[3]; /* проверка суммы для 9—digit ID */
```

```
char oshift=4; /* измениение */
char RQ3SIG [3] = «\x33\x33\x33»; /* rq—rq—rq */
char BAB [3] = «\x66\x78\x66»; /* b—a—b */
char EOCSIG[3] = «\x78\x78\x78»; /* a—a—a */
/*UCHAR PREKEY; */ /* PRE ключ в MS (0..100) */
/*UCHAR POSTKEY;*/ /* POST ключ в MS (0..100) */
UCHAR CFECRX=1; /* CFEC получение ON/OFF, по умолчанию ON
*/
UCHAR FECRATE=50; /* FEC коэффициент ошибки, по умолчанию
50% */
UCHAR INCHNL =2; /* ввод канала L/R, по умолчанию RIGHT */
UCHAR OUTCHNL=1; /* выход канала L/R, по умолчанию LEFT */
UCHAR LASTRXCS = 0;
UCHAR LASTRXBLOCK = 0;
UCHAR afterrephase = 0;
extern int SlaveAnswerTime; /* Данный способ отвечает на задержку */

void StandBy(void) // В алгоритме эта подпрограмма вызывается при
ошибке//
{MODEMSTATE=5; /* state = 5 —RESET */
COMF = STATE = SHIFT = RxErr = RxOK = /*RQ = RPTF = */ 0;
LASTSTATE = 0;
IDLEAcnt = 0;
connector = 0;
/*memset(DX,0,5);memset(RX,0,5);*/
FirstIdleBlock = 0;
SendFCCInfo = 0;
arqf = 0;
LASTCS = 0;
oshift = 4; /* изменение на неопределенное число */
WRUSUPPORT=1; /* по умолчанию ON */
hiClear();
hoClear();
Clear();
doClear();
if(dipsw & DIPSW3)
{if(SlaveAnswerTime != (21+2))
{SlaveAnswerTime = 21+2; /* принятый ответ удалить через 20 мс */
goto informhost;}}
else
{if(SlaveAnswerTime != (21+5))
{SlaveAnswerTime = 21+5; /* принятый ответ удалить через 50 мс */
informhost:
```

```
SendCommandHostDBG(1, «изменить время ответа [%X0 мсек] »,время загрузки ответа—21);}}
send_stat();
/*FS[0] = 1;*/ /* перезапустить свободный сигнал */
SendCommandHostDBG(1, «STAND BY»);}

void sendstat(void) /* Статус устройства отправляется в host—компьютер */
{
/*
SendCommandHost(«STAT»,«%02X;%02X;%02X;%03X;%02X;%02X»,
COMF,STATE,arqf,hiIFree,hicount,connector);
*/
SendCommandHost(«STAT»,«%02X;%02X;%02X;%03X;%02X;%X;%X»,
COMF,STATE,arqf,hiIFree,hicount,MODEM_STATE,connector);}

/* Mode */
/* 0 — ARQ */
/* 1 — CFEC */
/* 2 — SFEC */

UCHAR convert(char ch) // Это конвертер в NBDP формат//
{if(ch == ERRSMBL) return ERRSMBL;
{int index = table2[(int) ch];
if(index)
{struct CVT *cvt;
cvt = &CVT_TABLE[index—1];
switch(SHIFT)
{case 0: return cvt—>_LET;
case 1: return cvt—>_FIG;
case 2: return cvt—>_RUS;}}}
return ERRSMBL;}

void NBDPInit(void) // Инициализация NBDP вызывается из main.c
{SetID(1, «32610»,SIS5,NULL,1);
SetID(2, «123456789»,SIS9,SCS9,1);
strcpy(SIS9, «\x2e\x33\x65\x33\x27\x39\x2e\x71\x69»);
strcpy(SCS9, «\x17\x4e\x69»);

StandBy();}
```

1.4.3 Программа serial.c

Эта программа содержит в себе функции для работы с последовательным портом.

```
#include <stdarg.h>
#include <string.h>
#include <drivers.h>

#define MAXCMDLEN 127
unsigned int CMDS, /* command rx state */
CMDL; /* command rx pointer */
char CMD[MAX_CMD_LEN+1]; /*команда буферу rx */

int chr;
int SerialDriver(void) // проверяется работа и инициализация последовательности порта//
{/*int chr;*/
repeat:
if(getchar(&chr))
{if(CMDS) /* получение команды */
{if(CMDL>=MAXCMDLEN)
{SendHostError(0,0);
resetstate:
CMDL=CMDS=0;
return 0;}
if((CMD[CMDL—1] == ';') && (chr == '@')) /* EOC */
{CMD[CMDL++] = chr;
CMD[CMDL] = 0;
HostCommandParser(&CMD[0]);
goto resetstate;}
else
{CMD[CMDL++] = chr;
goto repeat;
return 0;}}
else /* ожидание команды */
{if(chr=='$')
{CMDL=0;
CMD[CMDL++] = chr;
CMDS=1;
goto repeat;}}}
return 0;}
extern unsigned int No;

char BUFFER[MAXCMDLEN+1];
```

```
int SendCommandHost(char *cmd, char *fmt,...) // функция, созданная для передачи команд в терминал//
{
/* делают данные строковыми */
/*int len;*/
valist argptr;
/* заголовок пакета: $CMD;channelno; */
outchar('$');
outstring(cmd/*,strlen(cmd)*/);
outchar(';');
outchar(No + 0x30);
outchar(';');
/* переменная часть */
vastart(argptr, fmt);
/*len =*/ vprinter(&BUFFER[0], fmt, argptr);
va_end(argptr);
outstring(BUFFER/*,len*/);
/* Out the end part of packet */
outstring(«;@\r\n»/*,4*/);
return 0;}

int SendCommandHostDBG(int level, char *fmt,...)
{/* сделать данные строковыми */
/*int len;*/
valist argptr;
if(!(dipsw & DIPSW4)) return 0; /* не выходной отладочный */
if(level > DebugLevel) return 0;
/* заголовок пакета: $CMD;channelno; */
outstring(«$DBG; »); outchar(No + 0x30); outchar(';');
/* переменная часть */
vastart(argptr, fmt);
vprinter(&BUFFER[0], fmt, argptr);
vaend(argptr);
outstring(BUFFER);
/* часть окончания пакета */
outstring(«;@\r\n»);
return 0;}
```

Критичные по быстродействию функции, выполняются на языке ассемблер под ADSP 2181.

Sin.dsp

Функции разложения на синус и косинус, которые используются для модуляции и демодуляции для сверхбыстрых вычислений.

```
.MODULE/RAM SINCOSINTEGER;
{Sine/Cosine approximation for 1.15 format
int Y = sini(int X)
int Y = cosi(int X)

Calling parameters
AR = X in scaled 1.15 format
M1 = 1
L1 = 0
Return values
AR = X in 1.15 format
Computation time
sin : 30 + (3) cycles
cosine: 32 + (3) cycles}
.VAR/DM sincoeff[5];
.INIT sincoeff : 0x3240, 0x0053, 0xAACC, 0x08B7, 0x1CCE;
.ENTRY sini;
.ENTRY cosi;
cosi:
AY1 = 0x4000; { AY0 = PI/2 }
AR = AR + AY1; { AR = X+PI/2 }
sini:
SI = AR; { save AR }
I1 = ^sincoeff; {ptr to coeff buffer }
AY1=0x4000;
AF=AR AND AY1; {check 2nd or 4th quad.}
IF NE AR=—AR; {If yes negate input }
AY1=0x7FFF;
AR = AR AND AY1; {remove sign bit }
MY1=AR;
/*#ifndef GLOBAL_F*/
SR1 = MSTAT; {save MSTAT}
DIS M_MODE; {set fractional}
/*#endif*/
SR0 = MX1; {save MX1}
MF=AR*MY1 (RND), MX1=DM(I1,M1); {MF = X^2 }
MR=MX1*MY1 (SS), MX1=DM(I1,M1); {MR = C1*X }
CNTR=3;
DO approx UNTIL CE;
MR=MR+MX1*MF (SS);
approx: MF=AR*MF (RND), MX1=DM(I1,M1);
MR=MR+MX1*MF (SS);
MX1 = SR0; { restore MX1 }
```

```
/*#ifndef GLOBAL_F*/
MSTAT = SR1; { restore MSTAT }
/*#endif*/
SR=ASHIFT MR1 BY 3 (HI);
SR=SR OR LSHIFT MR0 BY 3 (LO); {convert to 1.15 format}
AR=PASS SR1;
IF LT AR=PASS AY1; {saturate if needed }
AY1=SI;
AF=PASS AY1;
IF LT AR=—AR;
RTS;
.ENDMOD;
```

Программа 2181hdr.dsp содержит в себе код инициализации ADSP 2181.

```
.MODULE/ABS=0 ADSP2181Runtime_Header;
#define MYHANDLER 1
// объявление внешних функций Си для возможности использовать их в ассемблере//
.EXTERNAL libsetupeverything;
.EXTERNAL main;
#ifndef MYHANDLER
.EXTERNAL libsp0rctrl;
#endif
.EXTERNAL statflag_;
.EXTERNAL nextcmd_;
.EXTERNAL processabit; { uart }
.EXTERNAL IRQEFlag;
//Устанавливаются векторы прерываний//
// По сбросу загружается функция Си main//
Resetvector: CALL lib_setupeverything;
CALL main_; RTS; NOP; {Begin C program}
__Interrupt2: rti;NOP;NOP;NOP;
__InterruptL1: rti;NOP;NOP;NOP;
__InterruptL0: rti;NOP;NOP;NOP;
__Sport0_trans: jump _sp0tx;nop;nop;nop;
#ifndef _MY_HANDLER
__Sport0_recv: JUMP libsp0rctrl;NOP;NOP;NOP;
#else
_Sport0_recv: JUMP sp0rx;NOP;NOP;NOP;
#endif
_InterruptE: ena secreg; ar = 1; dm(IRQEFlag)=ar; rti;
_BDMA_interrupt: rti;NOP;NOP;NOP;
```

```
#ifndef HW_UART
__Interrupt1: pop sts; /* 20: SPORT1 tx or IRQ1 */
ena timer; rts; rti;
#else
_Interrupt1: rti; /* 20: SPORT1 tx or IRQ1 */
rti; rti; rti;
#endif
_Interrupt0: rti;NOP;NOP;NOP;
_Timerinterrupt: jump processabit; /* 28: timer */
rti; rti; rti;
_Powerdown_interrupt: rti;NOP;NOP;NOP;
#ifdef _MY_HANDLER

1) Enble sec reg bank
2) Save all unaltered registers
3) Setup stack, Run C—handler
4) Restore unaltered registers
5) disable sec reg bank

.var/dm REG_SAVE_SP0RX[11];
.external modulator;
.external demodulator;
sp0rx:
//сохранение регистров//
dm(REGSAVESP0RX + 0) = PX; /* 1 */
dm(REGSAVESP0RX + 1) = L0; /* 2 */
dm(REGSAVESP0RX + 2) = I1; /* 3 */
dm(REGSAVESP0RX + 3) = L1; /* 4 */
dm(REGSAVESP0RX + 4) = M2; /* 5 */
dm(REGSAVESP0RX + 5) = M3; /* 6 */
dm(REGSAVESP0RX + 6) = M5; /* 7 */
dm(REGSAVESP0RX + 7) = L5; /* 8 */
dm(REGSAVESP0RX + 8) = I6; /* 9 */
dm(REGSAVESP0RX + 9) = M6; /* 10 */
dm(REGSAVESP0RX + 10) = L6; /* 11 */
/* enable second register bank */
ena secreg;
/* set predefined INTR modes */
DIS BITREV, DIS ARSAT, DIS AVLATCH, ENA MMODE;
LO=O;
L1=O;
L5=O;
L6=O;
M2=O;
```

```
M6=O;
/* restore unaltered registers */
PX = dm(REGSAVESP0RX + O)/* = PX*/; /* 1 */
LO = dm(REGSAVESP0RX + 1)/* = L0*/; /* 2 */
I1 = dm(REGSAVESP0RX + 2)/* = I1*/; /* 3 */
L1 = dm(REGSAVESP0RX + 3)/* = L1*/; /* 4 */
M2 = dm(REGSAVESP0RX + 4)/* = M2*/; /* 5 */
M3 = dm(REGSAVESP0RX + 5)/* = M3*/; /* 6 */
M5 = dm(REGSAVESP0RX + 6)/* = M5*/; /* 7 */
L5 = dm(REGSAVESP0RX + 7)/* = L5*/; /* 8 */
I6 = dm(REGSAVESP0RX + 8)/* = I6*/; /* 9 */
M6 = dm(REGSAVESP0RX + 9)/* = M6*/; /* 1O */
L6 = dm(REGSAVESP0RX + 10)/* = L6*/; /* 11 */
rti;
#endif
.var/dm REGSAVESP0TX[11];
sp0tx:
/* enable second register bank */
ena secreg;
ar = dm(statflag_);
ar = pass ar;
if ne jump nextcmd_;
/* save unaltered registers */
dm(REGSAVESP0TX + O) = PX; /* 1 */
dm(REGSAVESP0TX + 1) = L0; /* 2 */
dm(REGSAVESP0TX + 2) = I1; /* 3 */
dm(REGSAVESP0TX + 3) = L1; /* 4 */
dm(REGSAVESP0TX + 4) = M2; /* 5 */
dm(REGSAVESP0TX + 5) = M3; /* 6 */
dm(REGSAVESP0TX + 6) = M5; /* 7 */
dm(REGSAVESP0TX + 7) = L5; /* 8 */
dm(REGSAVESP0TX + 8) = I6; /* 9 */
dm(REGSAVESP0TX + 9) = M6; /* 1O */
dm(REGSAVESP0TX + 1O) = L6; /* 11 */
/* enable second register bank */
/*ena secreg;*/
/* set predefined INTR modes */
DIS BIT_REV, DIS AR_SAT, DIS AV_LATCH, ENA M_MODE;
LO=O;
L1=O;
L5=O;
L6=O;
M2=O;
M6=O;
```

```
call modulator;
/* restore unaltered registers */
PX = dm(REGSAVESP0TX + O)/* = PX*/; /* 1 */
LO = dm(REGSAVESP0TX + 1)/* = LO*/; /* 2 */
I1 = dm(REGSAVESP0TX + 2)/* = I1*/; /* 3 */
L1 = dm(REGSAVESP0TX + 3)/* = L1*/; /* 4 */
M2 = dm(REGSAVESP0TX + 4)/* = M2*/; /* 5 */
M3 = dm(REGSAVESP0TX + 5)/* = M3*/; /* 6 */
M5 = dm(REGSAVESP0TX + 6)/* = M5*/; /* 7 */
L5 = dm(REGSAVESP0TX + 7)/* = L5*/; /* 8 */
I6 = dm(REGSAVESP0TX + 8)/* = I6*/; /* 9 */
M6 = dm(REGSAVESP0TX + 9)/* = M6*/; /* 1O */
L6 = dm(REGSAVESP0TX + 1O)/* = L6*/; /* 11 */
rti;
.ENDMOD;
```

2 Экономическая часть

2.1 Цели и задачи проекта

Цель технико—экономического обоснования заключается в доказательстве того, что разработка наноспутника и программного обеспечения для него является не убыточным проектом с экономической точки зрения.

Технико—экономическое обоснование исследований данной работы содержит следующие части:

— трудоемкость выполнения научно—исследовательской работы (НИР);
— расчет затрат на выполнение НИР;
— определение вероятной цены НИР;
— оценку научно—технической результативности и эффективности НИР.

2.2 Определение трудоемкости выполнения НИР

Для определения трудоемкости выполнения НИР сначала нужно составить перечень всех основных этапов работ, которые должны быть сделаны.

Особое внимание стоит уделить логическому упорядочению последовательности видов работ и выявлению возможностей их одновременного выполнения, что позволит сократить общую длительность проведения НИР.

Таблица 2.1 – Распределение работ по этапам и видам и оценка их трудоемкости

Этап разработки	Вид работы на данном этапе	Трудоемкость выполнения НИР, чел.*ч.
1. Подготовительный	1. Постановка задачи	8
	2. Сбор материала и анализ существующих разработок	20
2. Постановочный	1. Проверка прочности Al 5005	6
	2. Проверка прочности Al 6061	6
	3. Проверка прочности Al 7075	10
	4. Проверка прочности Al Amr — 2	12
3. Корректирующий	Подготовка предварительных выводов	16
4. Заключение	1. Подведение итогов	7
	2.Оформление результатов	14
ИТОГО трудоемкость выполнения проекта		93

Количество часов активной работы для исследования составляет 93 часа. Рабочее время, которое выделялось на исследования, в сутки равно 8 часов. Из этого следует, что срок на выполнение данного проекта составляет 12 суток.

2.3 Расчет затрат на выполнение НИР

Данный проект предполагает создание наноспутника стандарта «CubeSat» с использованием алюминия высокопрочных сплавов. Из этого следует, что для того, чтобы достичь поставленных целей, нужно просчитать затраты на тестирование материала, стоимость итоговой конструкции, закупку соответствующего оборудования, а так же окупаемость затрат.

Таблица 2.2 – Затраты на техническое оснащение

Наименование материального ресурса	Единица измерения	Количество израсходованного материала	Цена за единицу, тг	Итого, тг
Лист алюминия (amr—2)	Шт.	1	12000	12000
Труба из нержавеющего материала 40х13	Шт.	1	1830	1830
Компьютер (Neo Office)	Шт.	1	86000	86000
Монитор (Samsung LS22D300NYI/CI)	Шт.	1	37300	37300
Принтер (Canon PixmaG1400)	Шт.	1	52000	52000
Программа Matlab Simulink	Шт.	1	20000	20000
Программа Assambler	Шт	1	15000	15000
Итоговые затраты				223830

На первом же этапе исследований работе используются следующие технические средства:

– Компьютер;
– Установочная среда Matlab Simulink и Assambler;
– Монитор;
– Принтер.

Итоговая сумма затрат на материальные ресурсы (Зм) рассчитывалась по формуле (2.1):

$$З_м = \sum_{i=1}^{n} P_i * Ц_i, \quad (2.1)$$

где Pi — расход i—го вида материального ресурса, натуральные единицы;

Цi — стоимость i—го вида материального ресурса за единицу, тг;

i — вид материального ресурса;

n — количество материальных ресурсов (видов).

При выполнении научно—исследовательской работы используется электрооборудование, из этого следует, что необходимо произвести расчет затрат на электроэнергию. С помощью формулы (2.2) произведем расчет общей суммы затрат на электроэнергию.

$$З_э = \sum_{i=1}^{n} M_i * K_i * T_i * Ц, \quad (2.2)$$

где Mi — мощность i—го электрооборудования по паспорту, кВт;

Ki — коэффициент использования мощности i—го электрооборудования;

Ti — время работы i—го оборудования за весь период работы, час;

Ц — цена за кВт электроэнергии, тг/кВт/ч;

i — вид электрооборудования;

n — количество электрооборудования (шт).

Расчет затрат на электроэнергию представлен в таблице 2.3

Таблица 4.3 – Затраты на электроэнергию

Неименование	W, кВт	Коэф. использования W	Время работы оборудования для ПП	Цена тг/кВт*ч	∑, тенге
Компьютер	0,4	0,9	93	25	837,0
Монитор	0,06	0,8	93	25	111,6
Принтер	0,08	0,8	40	25	64,0
Модем	0,08	0,8	93	25	148,8
ИТОГО					1160,6

Заработная плата за выполнения научно—исследовательской работы расчитывается в зависимости от тарифной ставки разработчика за час работы и времени, потраченного на каждый этап работы.

Итоговая сумма, затраченная на оплату труда (Зтр) определяется по формуле:

$$З_{тр} = \sum_{i=1}^{n} ЧС_i * T_i, \quad (2.3)$$

где $ЧС_i$ – ставка за час работы i—го работника, тг;

T_i — трудоемкость разработки ПП, чел.×ч;

i — категория работника;

n — общее количество работников, занимающихся разработкой САД.

Расчёт заработной платы работников по данной НИР представлен в таблице 2.4.

Таблица 2.4 – Затраты на оплату труда НИР

Категория работника	Кваллификация	Трудоемкость разработки ПП	Часовая ставка, тенге	∑, тенге
Научный руководитель	Руководитель проекта	15	2000	30000
Разработка документации	Руководитель проекта	15	2000	30000
Проектировщик	Разработчик проекта	31	700	21700
Программист	Разработчик проекта	32	700	22400
ИТОГО		93		104100

Социальный налог – согласно Налоговому кодексу РК он составляет 11% от ФОТ (фактическая оплата труда), пенсионные отчисления не облагаются налогом. Расчет выполняется с помощью формулы (2.4):

$$ОС = (ФОТ - ПО) * 11\%, \quad (2.4)$$

где ПО – отчисления в пенсионный фонд, которые составляет 10% от ФОТ:

$$ПО = ФОТ * 10\% = 104100 * 0{,}1 = 10410 \text{ тг}, \quad (2.5)$$

Подставим данные, полученные в ходе расчетов, в формулу (2.4):

$$ОС = (104100 - 10410) * 0{,}11 = 10306 \text{ тг}.$$

В амортизацию основных фондов включается сумма амортизационных отчислений, расчитанных от стоимости оборудования и программного обеспечения, используемых при выполнении НИР.

Итоговая сумма амортизационных отчислений рассчитывается по формуле (2.6):

$$З_{АМ} = \sum_{i=1}^{n} \frac{Ф_i * Н_{A_i} * Т_{НИР_i}}{100 * Т_{ЭФ_i}}, \quad (2.6)$$

где $Ф_i$ — стоимость i—го ОФ, тг;

$Н_{Ai}$ — годовая норма амортизации i—го ОФ, %;

$Т_{НИРi}$ = время работы за весь период выполнения НИР i—го ОФ, час;

$Т_{ЭФi}$ — эффективный фонд времени работы i—го ОФ за год, ч/год;

i — вид ОФ;

n — общее количество ОФ.

Расчет амортизационных отчислений за время выполнения научно—исследовательской работы приведен в таблице 2.5.

Таблица 2.5 – Амортизация основных фондов

Оборудование	Первоначальная стоимость, тг	Годовая норма амортизации, %	Эффективный фонд времени работы об—я, ч/год	Время работы оборуд—я для выполнения НИР, ∑, тенге	Сумма, тг
Компьютер	86000	20%	2920	93	547,80
Монитор	37300	20%	2920	93	237,59
Принтер	52000	15%	2920	40	248,42
Программа Си	20000	25%	2920	93	95,55
Программа Assambler	15000	25%	2920	93	71,66
ИТОГ					1201,02

Рассчитаем амортизационные отчисления:

а) Компьютер:

$$З_{АМ} = \frac{86000 * 20 * 93}{100 * 2920} = 547{,}80тг$$

b) Монитор:

$$З_{АМ} = \frac{37300 * 20 * 93}{100 * 2920} = 237{,}59тг$$

c) Принтер:

$$З_{АМ} = \frac{52000 * 15 * 40}{100 * 2920} = 106{,}85тг$$

d) Программа Си:

$$З_{АМ} = \frac{20000 * 25 * 93}{100 * 2920} = 159{,}25тг$$

e) Программа Assambler:

$$З_{АМ} = \frac{15000 * 25 * 93}{100 * 2920} = 119{,}43тг$$

Сумму, необходимую на аренду помещения рассчитаем по формуле:

$$Ц = S_п * Ц_{кв}, \quad (2.7)$$

где $S_п$ – площадь помещения (=20 м^2);
$Ц_{кв}$ – цена за 1м^3 помещения (=6000).

Ц= 20 * 6000 = 120000тг

Стоимость интернета рассчитаем по формуле (2.8):

$$Ц = S_1 * N, \quad (2.8)$$

где S1 – стоимость 1 Гб интернета (730 тг);
N1 – необходимое кол—во Гб (5 Гб).

Ц = 730 * 5 = 3650 тг

Таблица 2.6 – Смета затрат на выполнение НИР

Статьи затрат	**Сумма, тг**
Затраты на оплату труда	104100
Социальный налог	10410

Накладные расходы	62754,57
Амортизация основных фондов	1201,02
Затраты на электроэнергию	4441,22
Затраты на интернет	3560
Затраты на аренду помещения (месяц)	120000
ИТОГО по смете	387006,81

На рисунке 2.1 показана диаграмма распределения средств на создание наноспутника.

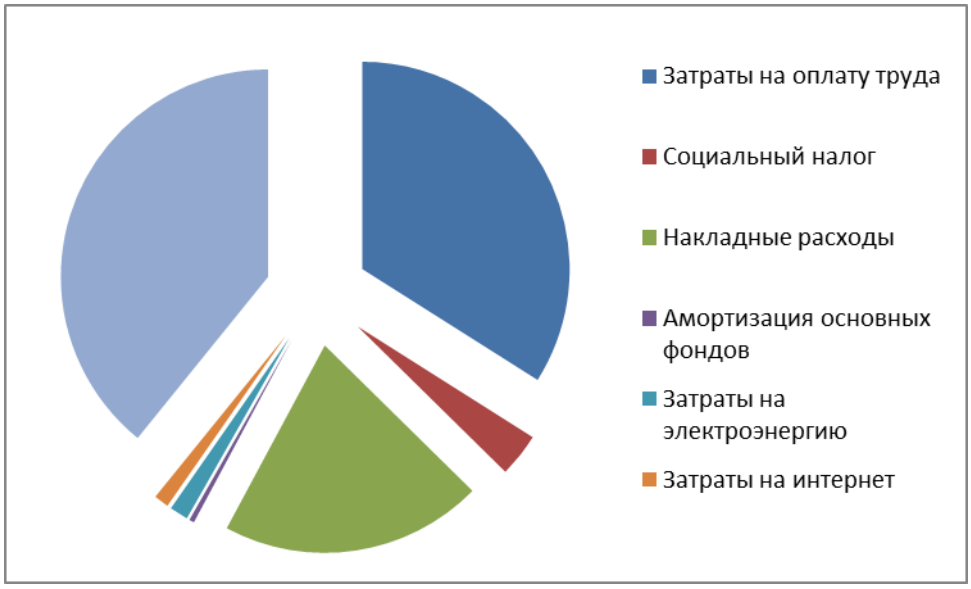

Рисунок 2.1 – диаграмма распределения средств на создания наноспутника

2.4 Определение вероятной (договорной) цены НИР

Величина вероятной (договорной) цены НИР должна быть установлена с учетом эффективности, качества и сроков выполнения проекта на уровне, отвечающем экономическим интересам заказчика и исполнителя.

Договорная цена (Цд) для прикладных НИР находится по формуле (2.9):

$$Ц_д = З_{НИР} * \left(1 + \frac{Р}{100}\right), \quad (2.9)$$

Где Знир— затраты на выполнение НИР (данные из таблицы 4.6), тг;
Р— средний уровень рентабельности НИР, % (примем 25%).
Подставим данные в формулу (2.9) , получим:

$$Ц_д = 387006{,}81 * \left(1 + \frac{25}{100}\right) = 483758{,}51 \text{ тг}$$

После рассчитывается цена реализации проекта с учетом налога на добавленную стоимость (НДС), ставка НДС устанавливается Налоговым Кодексом РК. На 2016 год ставка НДС установлена в размере 12%.

Цена реализации с учетом НДС рассчитывается по формуле(2.10):

$$Ц_р = Ц_Д + Ц_Д * НДС, \quad (2.10)$$

Подставив данные в формулу (2.10), получим:

$$Ц_р = 387006{,}81 + 483758{,}51 * 0{,}12 = 445057{,}83 \text{ тг}$$

2.5 Оценка социально — экономических результатов выполнения НИР

На создание данного проекта приходится 12 суток. По истечению 12 суток срока исследование будет завершено, будут подготовлены все материалы для дальнейшего исследования и внедрения технологий в работу.

Уже множество спутников вращаются вокруг Земли и отправляют данные о других планетах на Землю. Космические аппараты решают множество задач – военные, исследовательские, социально—экономические, научные, прием и передача информации. Благодаря наноспутникам, значительно уменьшены расходы на создание и запуск космических аппаратов, в данной части расчета, рассчитана стоимость создания корпуса спутника и программного обеспечения к нему. Сегодня наноспутники являются помощниками инженерам и ученым, они обладают большой функциональностью, несмотря на небольшой размер. С их помощью можно проводить тестирование микроэлектроники, изучать Земную атмосферу, следить за передвижением каких—либо объектов (например, морских судов в море).

Данное исследование позволит упросить запуск малых космических аппаратов в космос, так как у них низкие требования к созданию конструкции. В процессе работы были рассчитаны экономические характеристики разработки наноспутника. По данной работе было рассчитано, что себестоимость данного НИР составляет $387006{,}81$ тг. В результате цена разработки данной конструкции составляет $445057{,}83$ тг.

3 Безопасность жизнедеятельности

Административное здание центра управления полетами панельного типа. В нем не установлены автоматические системы сигнализации и оповещения о пожаре. Здание имеет два этажа, его размеры в плане здания12х32 м, в его коридорах шириной 3 м имеются схемы эвакуации людей при пожаре. На каждом этаже располагается по 8 кабинетов. На первом этаже располагается 76 рабочих мест, из них 50 человек занимаются разработкой программного обеспечения для управления летательными аппаратами, 18 человек занимаются

отчетами по полетам и 8 человек руководят отделом. На втором этаже находится 98 рабочих мест, из них 14 человек занимаются сбором и обработкой информации, 25 человек наблюдают за данными, полученными с приборов КА и за правильностью работы аппаратов, 52 человека занимаются устранением ошибок во время полета и 7 человек следят за правильностью отчетов и действий персонала. План второго этажа здания представлен на рисунке 3.1. Первый этаж аналогичен второму.

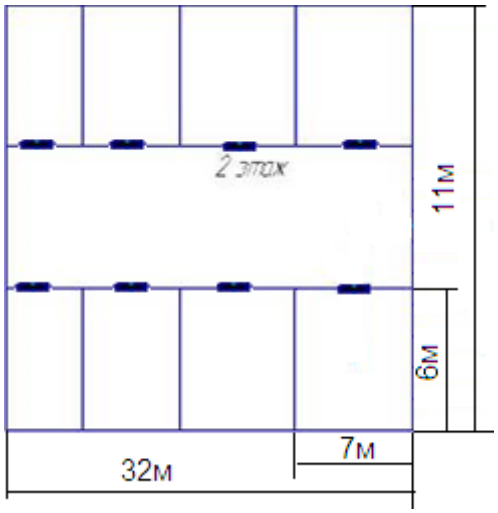

Рисунок 3.1 – План второго этажа здания ЦУП

В кабинете, расположенном на втором этаже, слева от лестницы, находится наибольшее количество электронной аппаратуры. Перечень аппаратуры представлен в таблице 3.1.

Таблица 3.1 – Рабочее оборудование помещения:

Рабочее оборудование	**Количество (шт)**
Системный блок	20
Монитор	20
Телевизор	3
Принтер	5
Модем	1
Кондиционер	2
Настольная лампа	25
Диспенсер с водой	1

В этом кабинете наибольшая вероятность возникновения пожара в случае короткого замыкания. Поражение током — опасный фактор для сотрудников ЦУП. На участке применяется напряжение 380 В, частота тока составляет 50 Гц. В данной части дипломного проекта будет приведен расчет времени эвакуации людей из данного кабинета при пожаре, будет сделан вывод, нужны

ли в этом кабинете датчики пожара, а так же будет проведен расчет защитного заземления электроприборов.

В этом кабинете наибольшая вероятность возникновения пожара в случае короткого замыкания. Поражение током — опасный фактор для сотрудников ЦУП. На участке применяется напряжение 380 В, частота тока составляет 50 Гц. В данной части дипломного проекта будет приведен расчет времени эвакуации людей из данного кабинета при пожаре, будет сделан вывод, нужны ли в этом кабинете датчики пожара, а так же будет проведен расчет защитного заземления электроприборов.

3.1 Эвакуация при пожаре

Необходимо определить время эвакуации из кабинета сотрудников Центра управления полетом при возникновении пожара в здании. Кабинет сотрудников, наблюдающих за данными, полученными с космического аппарата, имеет объем 126 м3 (высота потолков равна 3 м) и расположен на втором этаже в непосредственной близости от лестничной клетки, которая ведет на первый этаж. Лестничные клетки шириной 1,5 м и длиной 10 м. В кабинете работает 25 инженеров, отвечающих за данные, полученные с КА. Всего на втором этаже располагаются 98 человек. На первом этаже располагается 76 рабочих мест. Схема эвакуации из здания представлена на рисунке 3.2

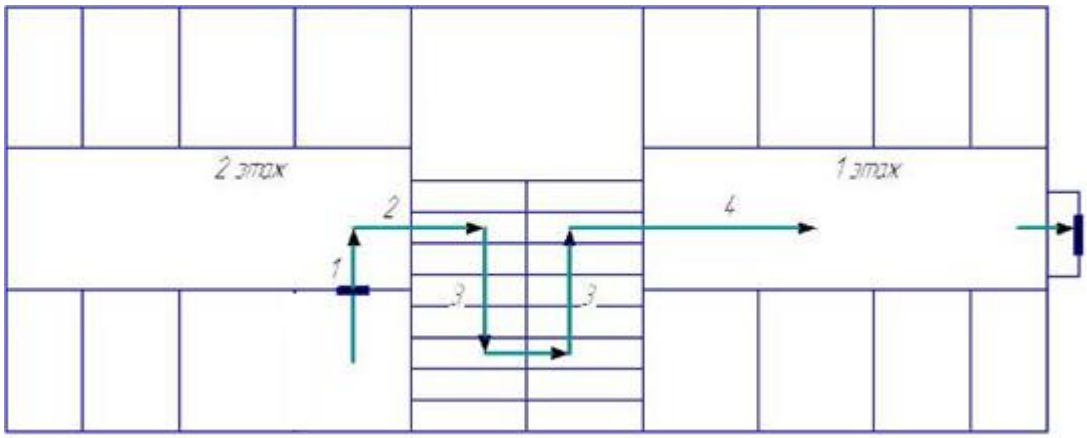

Рисунок 3.2 – План эвакуации сотрудников ЦУП на участках 1,2,3,4

3.2 Расчет времени эвакуации

По категории помещение относится к группе Д и II степени огнестойкости. Эти значения устанавливаются нормативными документами.

Критическая продолжительность пожара по температуре рассчитывается по формуле (3.1) с учетом мебели в помещении:

$$\tau_{n.k.} = \frac{W_{пом} \cdot c \cdot (t_{кр} - t_n)}{(1-\varphi) \cdot Q \cdot f \cdot n}, \quad (3.1)$$

где $W_{пом}$ – объем воздуха в данном помещении, м3;

с – удельная изобарная теплоемкость газа, кДж/кг—град;

$t_{Кр}$ – критическая для человека температура (равна 70°С);

$t_Н$ – изначальная температура воздуха, °С;

φ– коэффициент, характеризующий потерю тепла на нагрев конструкций и окружающих предметов (принимается равным 0,5);

Q – теплота сгорания веществ, кДж/кг;

f – площадь поверхности горения, $м^2$;

п – весовая скорость горения, кг/$м^2$—мин;

V – линейная скорость распространения огня по поверхности горючих веществ, м/мин.

$$\tau_{n.k} = \sqrt[3]{\frac{100{,}8 \cdot 1009 \cdot (70-20)}{(1-0{,}5) \cdot 3{,}14 \cdot 13800 \cdot 14 \cdot (0{,}36)^2}} = \sqrt[3]{129{,}36} = 5{,}05 мин$$

Критическая продолжительность пожара по концентрации кислорода рассчитывается по формуле (3.2):

$$\tau_{nk}^{O_2} = \sqrt[3]{\frac{100 * W_{пом}}{\pi * n * W_{O_2} * \sqrt{2}}}, \quad (3.2)$$

где W_{02} – расход кислорода на сгорание 1 кг горючих веществ, м /кг (составляет 4,76 м/кг)

$$\tau_{n.k.}^{O_2} = \sqrt[3]{\frac{100 \cdot 100{,}8}{3{,}14 \cdot 14 \cdot 4{,}76 \cdot (0{,}36)^2}} = \sqrt[3]{371{,}69} = 7{,}19 мин$$

Минимальная длительность пожара по температуре составляет 5,05 мин. Допустимая продолжительность эвакуации для помещения ЦУП находим по формуле (3.3):

$$\tau_{доп}^{1} = m * \tau^{1}_{n.k}, \quad (3.3)$$

где $\tau_{n.k.}$ и $\tau_{n.k.}^{11}$ – допустимая продолжительность эвакуации при пожаре.

$$\tau^{1}_{доп} = 1 * 5{,}05 = 5{,}05 \text{ мин}$$

Время задержки операции эвакуации принимается 4,1 мин по таблице 3.2 с учетом того, что здание не имеет автоматической системы сигнализации и оповещения о пожаре.

Таблица 3.2 – Время задержи эвакуации

Тип здания	Время задержи начала эвакуации, мин, при типах систем оповещения			
	W1	W2	W3	W4
Административные, торговые и производственные здания, если посетители знакомы с планировкой здания и процедурой эвакуации	<1	3	>4	<4
Магазины, выставки, музеи, досуговые центры и другие здания массового назначения, посетители не знакомы с планировкой здания и процедурой эвакуации	<2	3	>6	<6
Общежития и интернаты, посетители могут находиться в состоянии сна, но знакомы с планировкой здания и процедурой эвакуации	<2	4	>5	<5
Отели и пансионаты, посетители могут находиться в состоянии сна, они не знакомым с планировкой здания и процедурой эвакуации	<2	4	>6	<5
Госпитали, дома престарелых и др., где значительное число посетителей может нуждаться в помощи	<3	5	>8	<8
Примечание: Характеристика системы оповещения W1 – оповещение и управление эвакуацией оператором; W2 – использование записанных заранее типовых фраз и информационных табло; W3 – сирена пожарной сигнализации; W4 – без оповещения.				

Чтобы определить время движения людей по первому участку, с учетом габаритных размеров кабинета 6х7 м, определим плотность движения потока людей на первом участке по формуле (1.4):

$$D_1 = \frac{N_1 \cdot f}{L_1 \cdot b_1}, \qquad (3.4)$$

где N_1 – число людей на участке 1, чел.;

f – средняя площадь горизонтальной проекции человека, $м^2$/чел.;

L_1 и b_1 – длина и ширина первого участка пути, м.

$$D_1 = \frac{7 \cdot 0{,}1}{6 \cdot 7} = 0{,}01 \ м^2/м^2.$$

По таблице 3.3 определяем скорость движения потока, она составляет 100 м/мин, интенсивность движения 1 м/мин, тогда время движения по первому участку определим по формуле (3.5):

$$t_n = \frac{L_n}{V_n}, \qquad (3.5)$$

где L_n – длина участка n—го пути, м;

V_n – скорость движения людского потока по горизонтальному пути на n—м участке, она определяется в зависимости от относительной плотности D, $м/м^2$.

$$t_1 = \frac{7}{100} = 0{,}07 \text{ мин}$$

Таблица 3.3 Зависимость скорости и интенсивности движения от плотности потока людей

Плотность потока D,	Горизонтальный путь		Дверной проем	Лестница вниз		Лестница вверх	
	V, м/мин	q> м/мин	q, м/мин	V, м/мин	q, м/мин	V, м/мин	q, м/мин
0,01	100	1,0	1,0	100	1,0	60	0,6
0,05	100	5,0	5,0	100	5,0	60	3,0
0,1	80	8,0	8,7	95	9,5	53	5,3
0,2	60	12,0	13,4	68	13,6	40	8,0
0,3	47	14,1	15,6	52	16,6	32	9,6
0,4	40	16,0	18,4	40	16,0	26	10,4
0,5	33	16,5	19,6	31	15,6	22	11,0
0,6	27	16,2	19,0	24	14,4	18	10,6

0,7	23	16,1	18,5	18	12,6	15	10,5
0,8	19	15,2	17,3	13	10,4	10	10,0
0,9 и более	15	13,5	8,5	10	7,2	8	9,9
Примечание. Табличное значение интенсивности движения в дверном проеме при плотности потока 0,9 и более, равное 8,5 м/мин, установлено для дверного проема шириной 1,6 м и более.							

Длина дверного проема равна нулю. Наибольшая интенсивность движения в проеме в нормальных условиях, которая возможна, gmffic=19,6 м/мин, интенсивность движения в проеме шириной 1,1 м рассчитывается по формуле (3.6):

$$q_d = 2,5 + 3,75 \cdot b, \qquad (3.6)$$

где b – ширина проема.

$$q_d = 2,5 + 3,75 \cdot b = 2,5 + 3,75 \cdot 1,1 = 6,62 \text{ м/мин},$$

q_d $q_d \leq q_{max}$, поэтому движение через проем будет проходить беспрепятственно. Время движения в проеме определим по формуле (3.7):

$$t_{dL} = \frac{N*f}{q*b}, \qquad (3.7)$$

$$t_{dL} = \frac{7 \cdot 0,1}{6,62 \cdot 1,1} = 0,09 \text{мин}$$

На втором этаже работает 98 человек, рассчитаем плотность людского потока второго этажа рассчитаем, подставив данные в формулу (3.4) :

$$D_2 = \frac{98 \cdot 0,1}{28 \cdot 3} = 0,11 \text{м}^2 / \text{м}^2$$

По таблице 1.3 скорость движения составляет 80 м/мин, интенсивность движения 8 м/мин, рассчитаем время движения по второму участку (из коридора на лестницу), подставив значения в формулу (3.5):

$$t_2 \frac{L_2}{V_2} = \frac{28}{80} = 0,35 \text{мин}$$

Для того, чтобы определить скорость движения людей по лестнице, проведем расчёт интенсивности движения на третьем участке по формуле (3.8):

$$q_i = \frac{q_{i-1} \cdot b_{i-1}}{b_i}, \qquad (3.8)$$

где q_{i-1} — интенсивность движения людских потоков, соединяющихся вначале участка , м/мин;

b_{i-1} i – ширина участков пути соединения, м;

b_i – ширина данного участка пути, м.

$$q_i = \frac{8 \cdot 3}{1{,}5} = 16 \text{м/мин}$$

Следовательно, на лестнице скорость потока людей снижается до 40 м/мин. Время движения вниз по лестнице (3—й участок) найдем, подставив значения в формулу (3.5):

$$t_3 = \frac{L_3}{V_3} = \frac{10}{40} = 0.25 \text{мин}$$

При переходе на первый этаж, люди со второго этажа смешиваются с людьми с первого этажа. Плотность людского потока для первого этажа рассчитаем, подставив значения в формулу (3.4):

$$D_4 = \frac{N_4 \cdot f}{L_4 \cdot b_4} = \frac{76 \cdot 0{,}1}{28 \cdot 3} = 0{,}09 \text{м / мин}$$

При этом интенсивность движения составит около 8 м/мин.

При переходе на 4—й участок, потоки людей смешиваются, поэтому интенсивность движения определим по формуле (3.9):

$$q_i = \frac{\sum q_{i-1} \cdot b_{i-1}}{b_i}, \qquad (3.8)$$

$$qi = \frac{(16 * 1{,}5) + (8 * 3)}{3} = 16 \text{ м/мин}$$

По таблице 3.3 скорость движения — 40 м/мин, поэтому скорость движения по коридору первого этажа, по формуле (3.5):

$$t_4 \frac{L_4}{V_4} = \frac{28}{40} = 0{,}7 \text{мин}$$

Коридор при выходе на улицу длиной 5 метров, на этом участке создастся максимальная плотность людского потока, поэтому, согласно данным таблицы 3.3 скорость падает до 15 м/мин, а время движения по коридору рассчитаем, подставив данные в формулу (3.5):

$$t_5 \frac{L_5}{V_5} = \frac{5}{15} = 0{,}3\text{мин}$$

При максимальной плотности потока людей, интенсивность движения через дверной проем на улицу шириной более 1,6 м будет равна 8,5 м/мин, время движения через него рассчитаем, подставив данные в формулу (3.7):

$$t_{д2} = \frac{N \cdot f}{q \cdot b} = \frac{174 \cdot 0{,}1}{8{,}5 \cdot 2} = 1{,}02\text{мин}$$

Расчетное время эвакуации рассчитывается по формуле (1.9):

$$t_р = t_{н.э.} + t_1 + t_2 + t_3 + \ldots + t_i \qquad (3.9)$$

$$t_р = 4{,}1 + 0{,}07 + 0{,}09 + 0{,}35 + 0{,}25 + 0{,}7 + 0{,}3 + 1{,}02 = 6{,}88 \text{ МИН}.$$

Таким образом, расчетное время эвакуации из кабинетов здания ЦУП больше допустимого. Поэтому, здание, в котором располагается центр управления полетами, нужно оборудовать системой оповещения о пожаре и средствами автоматической сигнализации.

3.3 Защитное заземление электрооборудования

Так как в помещении располагается множество электрической техники, необходимо ее заземление. Заземленные объекты к магистральному заземлению соединяются параллельно. Магистраль заземления проходит по территории здания и соединена в двух местах с контурами заземления на первом этаже.

Расчет заземления проводится в соответствии с нормами СНиП 21—01—97. Напряжение на используемых линиях электропередач составляет 35 кВатт.

Типы заземлителей, которые применяются в данном здании: вертикальный стержневой, трубообразный заземлитель , его длина — L равна 5 м, его диаметр — d равен 0.1 м, t—0,5 м, t—0.5L+t=0.5*3+0.5=2 м. Горизонтальный полосовой имеет длину — L, равную 10 м, f=1 м, в=0,5 м. Линейное напряжете в сети (U) составляет 4 кВатт. Протяженность линий электропередач $L_{КЛ}$—3 км, а $L_{ВК}$—12 км.

Данный расчет производится с учетом того, что в наличие четыре заземлителя из вертикальных стержней размеров указанных выше, верхние

концы которых соединяются с помощью горизонтального электрода — стальной полосы с площадью сечения 1х40 мм, которая уложена в землю на глубину 1 м.

В качестве естественного заземлителя применяется металлическая технологическая конструкция, состоящая из десяти дюймовых стальных труб. Ее сопротивление растеканию тока с учетом сезонных изменений проводимости грунта примем равной 150 Ом.

Естественные заземлители — металлические конструкции, арматуры железобетонных конструкций (в случаях, которые допускает ПУЭ—96), трубопроводы и оборудование, имеющие хорошее соединение с землей.

В качестве искусственных заземлителей используют вертикально расположенные стальные трубы, угловую сталь, металлические стержни, горизонтально расположенные стальные полосы и т.д.

Отталкиваясь от имеющихся данных, определим расчетный ток замыкания на землю по формуле (3.10):

$$I=\frac{U}{350}*(35*L\text{кл}+L\text{вк}) \qquad (3.10)$$

Где U – линейное напряжение сети;

$L_{КЛ}$, $L_{ВК}$ — протяженность линий электропередач.

$$I=4/350*(35*3+12)=1{,}34\text{A}$$

Определим требуемое сопротивление заземляющего устройства:

Для вертикального заземлителя по формуле (3.11):

$$R_1=\frac{\rho_{РАСЧ}}{2*\pi*L}*(In\frac{2*L}{d}+\frac{1}{2}*In\frac{4t+L}{4t-L}), \qquad (3.11)$$

Для горизонтального заземлителя по формуле (3.12):

$$R_1=\frac{\rho_{РАСЧ}}{2*\pi*L}*In\frac{2*L^2}{b*f}, \qquad (3.12)$$

где $\rho_{РАСЧ}$ — расчетное удельное сопротивление грунта. Оно зависит от сезона года, состояния грунта во время измерения; для того, чтобы получить расчетное значение удельного сопротивления грунта вводится коэффициент сезонности. Этот коэффициент определяется с учетом характеристики климатических зон:

$$\rho_{РАСЧ}=\rho_{изм}*\psi \qquad (3.13)$$

где $\rho_{изм}$ — измеренное удельное сопротивление земли в теплое время года;

ψ — коэффициент сезонности.

В данном случае, измерение $\rho_{изм}$ проводилось в мае в третей климатической зоне. Удельное сопротивление составило: для вертикального заземлителя (L=5 м) $\rho_в$=90Ом*м, для горизонтального заземлителя $\rho_г$=80 Ом*м.

Учитывая коэффициент сезонности (для вертикального и горизонтального заземлителей), расчетные значения удельного сопротивления грунта для этих заземлителей составляют:

$$\rho_{ВРАСЧ}=\rho_{ВИЗМ}*\psi_{В}=90*1,3 = 117Ом*\ м;$$
$$\rho_{ГРАСЧ}=\rho_{ГИЗМ}*\psi_{Г}=80*4,2 = 336Ом*\ м.$$

Определим расчетные сопротивления растеканию заземлителей вертикального R_1 и горизонтального R_2:

$$R_1=\frac{117}{2*\pi*5}*(In\frac{2*5}{0,1}+\frac{1}{2}*In\frac{4*3+5}{4*3-5}) = 23,76Ом$$
$$R_2=\frac{336}{2*\pi*10}*In\frac{(2*10)^2}{0,5*1} = 35,75Ом$$

При этом уточним параметры заземлителя, сделав проверочный расчет. Примем горизонтальный заземлитель L=10 м; количество вертикальных заземлителей n=4 штуки. Тип заземлителя выберем контурный, размещенный по всему периметру установки. Вертикальные электроды располагаются на расстоянии a=5 м друг от друга. Так как принятый заземлитель контурный, n=4 штуки, а отношение a/L=5/5=l, определим коэффициенты использования заземлителей: вертикальных = 0,69, горизонтальных =0,45.

Теперь вычислим сопротивление растеканию принятого группового заземлителя:

$$R_e=\frac{R_1*R_2}{(R_1*\eta_Г+R_2*\eta_В*n)}=\frac{23.7666*35.75}{23,76*0,45+35,75*0,69*0,4}=7,8Ом$$

Если мощность генераторов и трансформаторов от 100 кВатт и менее допускалось увеличение сопротивления до 10 Ом. Если напряжение в электроустановках выше 1000В, сопротивление заземляющего устройства указывается в зависимости от величины тока замыкания на корпус; R_3 не должно превышать 10 Ом.

Если используются естественные заземлители, сопротивление искусственного заземления определяется из выражения:

$$R_и = \frac{R_E * R_З}{R_E - R_З} = \frac{150*10}{150-10} \approx 10 \text{ Ом}.$$

где, $R_И$ — сопротивление искусственного заземлителя;

$R_З$ — наибольшее допустимое значение сопротивления заземляющего устройства;

R_e — сопротивление растеканию естественного заземлителя.

Тогда требуемое сопротивление искусственного заземлителя будет примерно равно $R_И \cong 10$ 0м.

Сравнив результаты расчета сопротивления растеканию принятого нами группового заземлителя – R_e, с требуемым сопротивлением искусственного заземлителя $R_И$, можно увидеть, что $Re < R_И = 7{,}8 < 10$, это повышает условия безопасности. Этот результат примем, как окончательный.

Вывод: проектируемый групповой заземлитель является контурным, он состоит из четырех вертикальных трубообразных заземлителей, длиной 5 м и диаметром 100 мм и горизонтального заземлителя, который представляет из себя стальную полосу, длиной 10м, с площадью сечения 4х40 мм, заглубленных в землю на 1 м.

Заключение

Грамотно составленное программное обеспечение является главным атрибутом правильной работы космического аппарата в космосе.

В ходе работы были рассмотрены виды программного обеспечения, классификация космических аппаратов. В данном дипломном проекте была представлена разработка программного обеспечения устройства для приема и передачи данных с наноспутника. Так же рассмотрены вопросы о правильном выборе плат для создания программного обеспечения, разработаны схемы алгоритма программы.

В части экономического обоснования рассчитаны экономические затраты на создание корпуса наноспутника и программного обеспечения к нему.

В части безопасности жизнедеятельности просчитано время, затраченное на эвакуацию из здания центра управления полетами и соответственно, нужны ли в этом здании системы противопожарной безопасности, а так же, произведен расчет защитного заземления электроприборов.

Список литературы

1. http://www.vesti.ru/doc.html?id=2627012
2. http://www.kondor—tour.kz/kaz_kosm_era_kaz
3. Журнал «Новости комонавтики», №6, июнь 1999 г., стр.3, 4
4. 6. Гущин В.Н. Основы устройства космических аппаратов: Учебник для вузов. – М.: Машиностроение, 2003. – 272 с.: ил.
5. Штернфельд А.А. Введение в космонавтику. – М.: Наука, 1974. – 240 с.
6. ru.wikipedia.org
7. Фетисов, П.А. Справочник по пожарной безопасности. – М.: Энергоиздат, 1984. – 262 с.
8. Таблица физических величин: Справочник./ И.К. Кикоин [и др.]
9. ГОСТ 12.1.004–91. ССБТ. Пожарная безопасность. Общие требования. — Введ. с 01.07.1992. – М.: Изд—во стандартов, 1992. —78 с.
10. Амортизация основных средств: бухгалтерская и налоговая / Учебное пособие / Г. Ю. Касьянова (4—е издание, перераб. и доп.). – М.: АБАК, 2011.
11. Спутниковая связь и вещание: Справочник. — 3—е изд., перераб. и доп. / В.А. Бартенев, Г.В. Болотов, В.Л. Быков и др.; под ред. Л.Я. Кантора. — М.: Радио и связь, 1997.
12. Системы спутниковой связи / А.М. Бонч—Бруевич, В.Л. Быков, Л.Я. Кантор и др.; под ред. Л.Я. Кантора: Учебное пособие для вузов. — М.: Радио исвязь, 1992. —224 с.
13. Меныников В.А., Чернов В.В., Феоктистов Н.Н., Александров И.Е. Космос и связь // Электросвязь. — 1995. — №6. — С. 10—12.
14. Иванов Е.Н. Расчет и проектирование систем противопожарной защиты. М.: Химия,1990.
15. Кодекс Республики Казахстан «О налогах и других обязательных платежах в бюджет» (Налоговый кодекс) (с изменениями и дополнениями по состоянию на 14.01.2016 г.)
16. Панасенко С.П. Алгоритмы шифрования. Специальный справочник. – Сп.: 2009. —63с.
17. http://lordn.narod.ru/download/books/walla/programming/Spr_po_C/main.htm
18. http://c—book—help.narod.ru
19. http://altcode.ru/assembler/ssm/asm31.php
20. http://profxaker.at.ua/load/literatura/assembler/spravochnik_po_assembleru/8—1—0—8