Содержание

Введение

1. Аналитический обзор

1.1 Обзор способов кодирования-декодирования информации

1.2 Сравнительный анализ способов кодирования декодирования информации

1.3 Анализ аппаратной реализации

1.4 Сравнительный анализ способов аппаратной реализации

1.5 Выводы по аналитическому обзору

2. Разработка структурной схемы

3. Синтез схемы электрической принципиальной

3.1 Выбор цифрового сигнального процессора

3.2 Выбор кодека

3.3 Выбор драйвера интерфейса RS - 232

3.4 Выбор памяти с ультрафиолетовым стиранием

3.5 Выбор вспомагательных элементов схемы

4. Разработка алгоритма программы

4.1 Блок инициализации

4.2 Интерфейс приема / передачи

5. Разработка программного обеспечения

6. Технико-экономический расчет

7. Охрана труда

Приложение

Введение

Необходимость приёма и передачи информации беспокоило человечество всегда. В современном , насыщеном компьютерной техникой мире, это получило наиболее широкое распространение. Возможность соединения нескольких компьютеров, находящихся на расстоянии позволяющим соединить их эл. проводом, и доступ к их данным, добавила качественно новую ступень к использованию возможностей современных ЭВМ. Такое соединение называется локальной сетью. Также после этого появилось и понятие глобальной сети, при этом компьютеры могут не находиться рядом, а допустим в разных городах. При таком соединении используется специальное устройство называемое "модем". Связь при этом обеспечивается по телефонной линии.

Модем - это сокращение от МОдулятор - ДЕМодулятор.

Также еще существует способ приёма и передачи информации между компьютерами по радиоканалу. В этом случае также используется устройство модуляции/демодуляции (модем). При этом с компьютером и модемом используется также отдельное устройство - блок приёма и передачи информации по радиоканалу. Это достаточно громоздкое устройство и каждый пользователь ЭВМ, конечно, не может себе его приобрести. Но такая комбинация технических средств очень эффективна при связи двух объектов находящихся на очень большом расстоянии и не обладающими доступом к телефонной линии. Например, это могут быть судно, находящееся в рейсе и порт приписки передающий информацию со спутника о надвигающейся буре.

Конечно, модем в этом случае будет по функциям отличаться от модема работающего с телефонной линией. Т.к. тут нет понятия дозвона до абонента, дуплексная связь также здесь не допустима. В принципе функции дозвона и другие берет на себе блок приёма и передачи информации по радиоканалу. Модем же только ожидает получение сигнала , производит его демодуляцию, образуя цифровой код, и передает его в компьютер. При передаче, модем принимает цифровой код, модулирует его, преобразует в аналоговый сигнал и передает в блок передачи информации по радиоканалу.

В наше время технология производства интегральных микросхем, микроконтроллеров и т.п. находится на очень высоком уровне, постоянно совершенствуется и изобретает все новые виды микрочипов. Одним из таких микрочипов является DSP - digital signal processor ( цифровой сигнальный процессор). Это идеальное средство для обработки сигналов. Имея встроенный язык программирования, он позволяет его настраивать на любую работу необходимую электронщику. Практически во всех современных модемах независимо от назначения установленны DSP.

В данном дипломном проекте мы будем проектировать устройство, которое будет принимать и передавать данные по радиоканалу, при этом выполняя кодирование и декодирование информации, используя цифровой сигнальный процессор (DSP).

1. Аналитический обзор

1.1 Обзор способов кодирования - декодирования информации

Для выбора необходимого пути проектирования устройства требуется провести анализ современных способов и средств кодирования-декодирования информации.

С начала рассмотрим способы решения кодирования-декодирования информации. Для этого рассмотрим современные способы модуляции - демодуляции сигнала.

Как сказано выше модемы модулируют сигнал для передачи по телефонным или радио каналам, но сигнал может быть промодулирован разными способами.

Модуляция - изменение одного или нескольких параметров несущего синусоидального колебания (амплитуды, частоты, фазы) в соответствии со значениями двоичной информации, передаваемой источником.

В модемах используется разновидность модуляции, так называемая "манипуляция", при которой указанные модулируемые параметры могут иметь только фиксированные значения из некоторого определенного набора.

Модуляция позволяет согласовать спектр передаваемого информационного сигнала с полосой пропускания телефонного или радио канала. При малых скоростях передачи (до 1200 бит/с) в модемах применяется частотная модуляция, реализация которой на таких скоростях наиболее проста. При средних скоростях передачи (1200 - 4800 бит/с) используется дифференциальная разностная модуляция с числом возможных изменений фазовых положений от двух (1200 бит/с) до восьми (4800 бит/с) (фазовая модуляция). Передаваемые значения цифровой информации содержатся в приращениях фазы между данным и предыдущим элементом модулированного сигнала. При больших скоростях передачи (>4800 бит/с) и при передаче по коммутируемым каналам с частотным разделением направлений передачи, начиная с 2400 бит/с, используется комбинированная амплитудно-фазовая модуляция). При использовании этого вида модуляции цифровая информация содержится как в значении амплитуды, так и в приращениях фазы несущей частоты. При амплитудно-фазовой и многопозиционной фазовой модуляциях количество возможных позиций модулированного сигнала (или количество векторов сигнала) более двух. В этом случае один элемент модулированного сигнала содержит несколько битов цифровой информации (это число равно двоичному логарифму от количества возможных векторов модулированного сигнала).

Фазовая модуляция:

При использовании так называемой относительная фазовая манипуляция (phase shift keying, PSK), т.е. модуляция, при которой фаза несущей принимает только фиксированные значения из ряда допустимых значений (например, 0 , 90 , 180 и 270 град.), а информация закладывается в изменения фазы несущего колебания. При указанном выше наборе возможных фаз каждому изменению фазы соответствует определенное значение дибита, т.е. двух последовательных битов информации. Фазовая манипуляция относится к двухполосным методам модуляции, т.е. спектр модулированного сигнала располагается симметрично относительно несущей частоты, а ширина спектра в Гц на уровне 0,5 от его значения на несущей частоте равна модуляционной линейной скорости , выраженной в Бодах. Наиболее часто используются в модемах такие разновидности фазовой манипуляции, как относительная фазовая манипуляция (ОФМ) /скорость 1200 бит/с, два положения фазы/, четырехпозиционная (или квадратурная фазовая манипуляция /2400 бит/с, четыре положения фазы/) и восьмипозиционная (4800 бит/с, восемь положений фазы). Иногда в литературе указанные виды манипуляции называются соответственно ФРМ (фазоразностная модуляция), ДОФМ (двухкратная фазовая модуляция) и ТОФМ (трехкратная фазовая модуляция). Дальнейшее увеличение числа позиций с целью повышения скорости приводит к резкому снижению помехоустойчивости, поэтому на более высоких скоростях стали применяться комбинированные амплитудно-фазовые методы модуляции .

Амплитудно - фазовая модуляция :

В данном виде модуляции для повышения пропускной способности используется одновременная манипуляция двух параметров несущего колебания: амплитуды и фазы. Каждый возможный элемент модулированного сигнала (вектор сигнала или точка сигнального пространства) характеризуется значением амплитуды и фазы.

Для дальнейшего повышения скорости передачи количество "точек" пространства модулированного сигнала увеличивается в число раз, кратное двум. В настоящее время в модемах используются методы амплитудно-фазовой модуляции с числом возможных позиций сигнала до 256. Это означает, что скорость передачи информации превышает модуляционную линейную скорость до 7 раз.

Для обеспечения максимальной помехоустойчивости точки сигнального пространства размещаются на равном расстоянии с огибающей всех точек в форме квадрата (16-ти позиционная квадратурная АМ), восьмиугольника и т.п. Увеличение числа позиций сигнала приводит к быстрому снижению помехо-устойчивости приема.

Радикальным средством обеспечения помехоустойчивой передачи стало применение сочетания модуляции с "решетчатым" кодированием. При использовании этого метода вводится некоторая избыточность в пространство сигналов и за счет этого создаются корреляционные связи между передаваемыми сим-волами. Благодаря этому на приеме на основе анализа последо-вательности принятых элементов модулированного сигнала возможно выявление и исправление ошибок. Практически это дает значительное повышение помехоустойчивости приема.

Разновидность амплитудно-фазовой модуляции - 16-ти позиционная квадратурная АМ (сигнальное пространство 4х4 точек в форме квадрата, точки равноудалены одна от другой, и по 4 точки в каждом квадрате) используется в дуплексных модемах.

Частотная модуляция: ( Frequency shift keying, FSK )

В модемах используется так называемая частотная манипуляция, при которой каждому значению бита информации ("1" и "0") соответствует определенная частота синусоидального сигнала.

Спектральные характеристики сигналов с частотной манипуляцией допускают относительно простую реализацию модемов до скоростей 1200 бит/с.

Модуляция с минимальным сдвигом (MSK)

MSK - это продолжение частотной модуляции, при которой разница частот "1" и "0" по модулю всегда равны половине скорости передачи данных. Т.е. индекс модуляции m=0.5, определяется следующим образом:

Например: если скорость передачи данных 1200 бит/сек, то частота "1" 1200Hz, а частота "0" 1800Hz. См. рис. 1.1.1

Рис. 1.1.1 а) Цифровые данные b) MSK сигнал

Рис. 1.1.2 MSK модулятор

Гауссова модуляция с минимальным сдвигом (GMSK )

GMSK (Gaussian Minimum Shift Keying) - гауссовская манипуляция с минимальным частотным сдвигом.

В стандарте GSM применяется спектрально-эффективная гауссовская частотная манипуляция с минимальным частотным сдвигом (GMSK). Манипуляция называется "гауссовской" потому, что последовательность информационных бит до модулятора проходит через фильтр нижних частот (ФНЧ) с характеристикой Гаусса, что дает значительное уменьшение полосы частот

Рисунок 1.3. - Структурная схема GMSK - модуляции

Формирование GMSK радиосигнала осуществляется таким образом, что на интервале одного информационного бита фаза несущей изменяется на 90°. Это наименьшее возможное изменение фазы, распознаваемое при данном типе модуляции. Непрерывное изменение фазы синусоидального сигнала дает в результате частотную модуляцию с дискретным изменением частоты. Применение фильтра Гаусса позволяет при дискретном изменении частоты получить "гладкие переходы". В стандарте GSM применяется GMSK-модуляция с величиной нормированной полосы ВТ - 0,3, где В - ширина полосы фильтра по уровню минус 3 дБ, Т - длительность одного бита цифрового сообщения.

Модуляцию GMSK отличают следующие свойства, которые предпочтительны для подвижной связи:

• постоянная по уровню огибающая, которая позволяет использовать эффективные передающие устройства с усилителями мощности в режиме класса С;

• компактный спектр на выходе усилителя мощности передающего устройства, обеспечивающий низкий уровень внеполосного излучения;

• хорошие характеристики помехоустойчивости канала связи.

Импульсно - кодовая модуляция:

Наряду с использованием аналоговых (АМ) можно использовать импульсные методы модуляции, в частности, амплитудно-импульсную модуляцию (АИМ), что позволяет улучшить энергетические характеристики процесса передачи в целом, если учесть, что длительность излучаемого импульса может быть мала по сравнению с периодом несущей. Импульсные методы модуляции основаны на процессе дискретизации передаваемого аналогового сигнала, т.е. использовании последовательности выборок (выборочных значений) аналогового сигнала, взятых периодически с частотой дискретизации fд. Она выбирается из условия возможности последующего восстановления аналогового сигнала без искажений из дискретизированного сигнала с помощью фильтра нижних частот.

Следующим логичным шагом может быть квантование амплитуд импульсных выборок - процесс определения для каждой выборки эквивалентного ей численного (цифрового значения). Указанные два шага (дискретизация и квантование) определяют процессы, осуществляемые при импульсно-кодовой модуляции. Они позволяют перейти от аналогового представления речевого сигнала к цифровому.

Численное значение каждой выборки в этой схеме может быть далее представлено (закодировано) в виде 7 или 8 битного двоичного кода (на практике при использовании аналого-цифровых преобразователей (АЦП двоичное кодирование осуществляется непосредственно при квантовании). Такое кодирование (часто называемое кодификацией дает возможность передать 128 или 256 дискретных уровней амплитуды речевого сигнала, обеспечивая качественную передачу речи формально с динамическим диапазоном порядка 42 или 48 дБ. Учитывая, что выборки должны передаваться последовательно, получаем двоичный цифровой поток со скоростью 56 кбит/с (8 кгц х 7 бит) в случае 7 битного кодирования или 64 кбит/с (8 кГц х 8 бит) в случае 8 битного кодирования.

Использование ИКМ в качестве метода передачи данных позволяет:

Для систем цифровой телефонии - ликвидировать недостатки присущие аналоговым методам передачи, а именно:

•убрать существенное затухание сигнала и его изменение в сеансе связи и от сеанса к сеансу;

•практически убрать посторонние шумы;

•улучшить разборчивость речи и увеличить динамический диапазон речи.

Перспективные способы модуляции:

OCDM-модуляция :

В широкополосных сигналах, получаемых мультиплексированием нескольких широкополосных сигналов с ортогональным кодовым уплотнением (Orthogonal Code Division Multiplex -- OCDM), используется одновременно несколько широкополосных каналов на одной частоте. Каналы разделяются за счет применения ортогональных PN-кодов. Фирма Sharp анонсировала 10-мегабитный модем, построенный по этой технологии. Фактически одновременно передаются 16 каналов с 16-чиповыми ортогональными кодами. В каждом канале применяется BPSK, затем каналы суммируются аналоговым методом. Вследствие суммирования независимых каналов возникает довольно сильная паразитная AM, что требует использования линейного усилителя мощности и вызывает большее потребление энергии.

Data Mux -- мультиплексор входных данных

BPSK -- блок фазовой модуляции

Spread -- блок расширения спектра методом прямой после-довательности ; Sum -- выходной сумматор

Рисунок 1.4 - Схема OCDM модуляции (16 Spread-каналов по 0,68 кбит/с = 10 Мбит/с)

OFDM-модуляция

Широкополосные сигналы, получаемые мультиплексированием нескольких широкополосных сигналов с ортогональным частотным уплотнением (Оrthogonal Frequency Division Multiplex -- OFDM), представляют собой одновременную передачу на разных несущих частотах сигналов с фазовой модуляцией. Одним из ее преимуществ является высокая устойчивость к провалам в спектре, возникающим вследствие многолучевого затухания. Модуляция обеспечивает наименьшую интерференцию и спектр, весьма похожий на спектр "белого шума". Обработка OFDM обычно происходит с использованием быстрого преобразования Фурье (БПФ) и инверсного БПФ, что приводит к усложнению схем декодирования по сравнению с более простым типом модуляции.

Применяется OFDM в современных системах наземного цифрового ТВ-вещания (DTTV) в диапазонах ОВЧ/УВЧ.

Data mux - мультиплексор входных данных

Channel - частотный канал

BPSK -- блок фазовой модуляции

Sum -- сумматор частотных каналов

Рисунок 1.5 - Схема OFDM-модуляции (16 независимых каналов по 0,68 кбит/с = 10 Мбит/с)

1.2 Сравнительный анализ способов кодирования - декодирования информации

Сделаем сравнительный анализ рассмотренных выше способов модуляции и выберем подходящий.

Прежде всего определимся с требованиями к способу приёма/передачи сигнала. В нашем случае сигнал передается по радиоканалу. При этом требования к высоким скоростям , как в случае передачи по телефонной линии, отсутствуют .

Также , в отличие от передачи по телефонной линии, отсутствует понятие дуплексной связи. Исходя из этого будем выбирать наиболее простой и проверенный способ. Так называемые , переспективные способы модуляции, OCDM и OFDM не подходят, т.к. работают с широкополосными сигналами и слишком сложны в исполнении. Импульсно-кодовая модуляция также не подходит т.к. напра-влена на работу с цифровыми линиями. Среди таких видов модуляции как, фазовая , амплитудно-фазовая и частотная, наиболее подходящей является частотная модуляция. Частотная модуляция (FSK) обычно используется при приёме-передаче не требующей высоких скоростей, проста в реализации. Очень часто FSK используется в судовом телеграфе , а т.к наше устройство в принципе предназначено для установки на судах дальнего плавания, то при согласовании с телеграфом можно будет принимать сообщения прямо на ЭВМ. Вывод: выбираем частотную модуляцию.

1.3 Анализ аппаратной реализации

Аппаратная реализация устройства кодирования-декодирования информации может быть очень различной.

1) Приведем структурную схему аппаратного способа реализации устройства кодирования-декодирования использующего частотную модуляцию.

Рисунок 1.6 - Структурная схема передатчика.

Рисунок 1.7 - Структурная схема приёмника.

Описание работы.

При передаче от ЭВМ цифрового кода генераторы G1 и G2, заполняют сигнал, синусоидами с разными частотами, соответ-ствующими "1" и "0", т.е. первоначальный сигнал моделируется.

При приёме, полосовые фильтры Ф1 и Ф2 настроенные на полосы частот "1" и "0" формируют соответствующий цифровой код , крутые фронты которого создаются с помощью пороговых элементов ПЭ1 и ПЭ2. После чего происходит определение какой из кодов пришел и передача его в ЭВМ.

2) Рассмотрим следующую блок - схему устройства кодирования-декодирования информации.

В наше время все большую популярность получают устройства кодирования-декодирования информации с использованием цифровых сигнальных процессоров.

Приведем в пример устройство выпускаемое фирмой Analog Devices называемое SHARC EZ-Kit. Это устройство является универсальным комплексом по решению различных задач обработки сигналов, в том числе и задач кодирования и декодирования информации. На рис. 1.8 представленна блок-схема данного комплекса.

Рисунок 1.8 - Блок - схема комплекса SHARC EZ-Kit

Аналогов таких устройств существует большое количество, отличающихся типом DSP, кодека, элементной базой и т.п., но принцип у всех один - универсальность. Основной алгоритм работы устройства обеспечивается программным образом без вмешательства в аппаратную часть.

SHARC EZ-KIT Lite состоит из небольшой демонстрационной платы на основе процессора ADSP-21061, работающего на тактовой частоте 160 МГц. Кроме процессора плата содержит 16-битный звуковой стереокодек AD1847, контроллер интерфейса RS-232 и микросхему EPROM, установленную в кроватку. Внешние порты и различные управляющие сигналы процессора не использованные на самой плате выведены на контакты разъемов расширения (сами разъемы не установлены для упрощения платы). Таким образом, плата может использоваться как прототип устройства, если подключить ее к необходимым внешним модулям. Для удобства отладки на плате установлены три кнопки — сброс, прерывание, флаг и четыре раздельно управляемых светодиода. Кодек связан с процессором через один из последовательных портов, второй порт оставлен свободным. Предусмотрен также JTEG порт, предназначенный для подключения переходника внутрисхемного эмулятора EZ-ICE. Сам эмулятор в комплект поставки не входит и может быть приобретен отдельно.

Такие устройства достаточно дорогие в финансовом плане, но там где необходима высокая точность и быстродействие, они себя окупают. Например, данное уствойство SHARC EZ-Kit стоит на фирме Analog Devices - 220$.

1.4 Сравнительный анализ способов аппаратной реализации

В ходе анализа аппаратной реализации было рассмотренно два способа: первый без использования цифрового сигнального процессора, второй с его использованием.

Т.к. проектируемое устройство планируется использовать на судах дальнего плавания, то оно должно обладать высокой степенью точности, надежности и быстродействия. Все это можно реализовать только с использованием современных процессоров цифровой обработки сигналов. Исходя из этого, мы выбираем второй пример аппаратной реализации рассмотренный в предыдущем разделе. Конечно, наше устройство не будет похоже SHARC EZ-Kit, но принцип и технология останутся теми же.

1.5 Выводы по аналитическому обзору

После проведения аналитического обзора и сравнительного анализа было решено, что :

устройство кодирования-декодирования информации будем выполнять с использованием цифрового сигнального процессора;

весь алгоритм модуляции-демодуляции, передачи в порт ЭВМ, приёма из порта и т.п. выполняется программно под соответствующий DSP;

в качестве способа модуляции выбран метод частотной модуляции.

2. Разработка структурной схемы устройства

Составлять структурную схему устройства будем из основных функциональных блоков входящих в устройство.

Рисунок 2.1 - Структурная схема устройства

Описание структурной схемы:

Аналоговый вход

На вход устройства подается принятый по радиоканалу сигнал. Приемник сигнала из радиоканала нами рассматриваться не будет, но о наличие такового мы будем помнить. После попадания на вход устройства сигнал направляется в аналогово-цифровой кодер-декодер (кодек).

Кодек:

После фильтра низких частот сигнал поступает на кодер-декодер.

Кодек - это высокоскоростной чип, со встроенными аналого-цифровым и цифро-аналоговым преобразователями, последовательным портом, программируемым интервальным таймером, регистрами и т.п.

Кодек имеет определенную разрядность цифровых данных передаваемых по его последовательному порту. Разрядность должна совпадать с разрядностью DSP , а также частота дискретизации по теореме Котельникова, должна быть в два раза больше частоты сигнала.

Т.к. используется частотная модуляция и частота "1" = 1615 Hz и "0" = 1785 Hz. то, вполне достаточно выбрать 16-ти разрядный кодек, с частатой дискретизации 8 kHz.

Рассмотрим принципы АЦП и ЦАП преоразования, которые также используются в кодеке.

Физические сигналы являются непрерывными функциями времени. Чтобы преобразовать непрерывный, в частности, аналоговый сигнал в цифровую форму используются аналого-цифровые пребразователи (АЦП). Процедуру аналого-цифрового преобразования сигнала обычно представляют в виде последовательности трех операций: дискретизации, квантования и кодирования. Однако, если придерживаться терминологии принятой в литературе по системам цифровой связи, то первая операция, дискретизация, соответствует модуляции сигнала, а вторая операция, квантование, есть ни что иное, как один из cпособов кодирования. Поэтому процедуру аналого-цифрового преобразования сигнала можно также представить в виде последовательности двух операций - модуляции и кодирования.

Операция дискретизации заключается в определении выборки моментов времени измерения сигнала. Операция квантования состоит в считывании значений координаты сигнала в выбранные моменты измерения с заданным уровнем точности, а операция кодирования - в преобразовании полученных измерений сигнала в соответствующие значения некоторого цифрового кода или кодовой комбинации, которые затем передаются по каналам связи.

Процедуру восстановления непрерывного сигнала из цифрового представления также можно представить в виде двух операций: декодирования и демодуляции. Операция декодирования выполняет операцию обратную операции кодирования, т.е. преобразует последовательность заданных значений кодовой комбинации (кодовых слов) в последовательность измерений, следующих друг за другом через заданные интервалы времени дискретизации. Операция демодуляции выполняет интерполяцию или восстановление непрерывного сигнала по его измерениям. Преобразование сигнала из цифровой формы в непрерывный сигнал осуществляется цифро-аналоговыми пребразователями (ЦАП).

Итак, после поступления аналогового сигнала , кодек преобразует его в цифровой код, который по последовательному порту передается в цифровой сигнальный процессор (DSP), где по определенному алгоритму преобразуется и передается в ЭВМ.

При приеме цифрового кода от DSP, по своему последовательному порту, кодек преобразует его в аналоговый сигнал и передает на выход, где сигнал поступает на передатчик по радиоканалу (который не входит в наше устройство) и передается на другое такое же устройство.

Инициализация и настройка работы кодека, также как в DSP, осуществляется программно, что намного упрощает его точную настройку.

Цифровой сигнальный процессор (ЦСП или DSP):

Главной частью нашего устройства кодирования - декодирования информации является цифровой сигнальный процессор ( digital signal processor , DSP).

Цифровые сигнальные процессоры обладают огромными возможностями по цифровой обработке сигналов : спектральный анализ, цифровая фильтрация, преобразование частоты дискретизации, подстройка в реальном времени скорости воспроизведения и т.п.

3. Разработка схемы электрической принципиальной

3.1 Выбор цифрового сигнального процессора

Цифровой сигнальный процессор (DSP) - является основой нашего устройства. Алгоритм кодирования - декодирования информации, прием и передача сигнала из ЭВМ и обратно, выполняются с помощью DSP.

Существует много различных фирм - производителей DSP: Texas Instruments, Analog Devices, Motorola , Philips и т.п. Реализацию нашего устройства можно выполнить практически с помощью DSP любой из перечисленных фирм. Рассмотрим цифровые сигнальные процессоры фирмы Analog Devices, и в частности сигнальный процессор ADSP - 2181.

Процессор ADSP-2181, является развитием семейства ADSP-21xx, 16-разрядных сигнальных процессоров DSP фирмы Analog Devices с фикси-рованной точкой. В ADSP-2181 используется базовая архитектура ядра этого семейства. ADSP-2181 обладает полным объемом внутренней памяти адресуемой на кристалле, расширенным набором внутренних периферийных устройств, высокой производительностью.

Краткая характеристика ADSP - 2181:

• 16 К x 24 бит внутренней памяти программ PM (Program Memory)

• 16 К x 16 бит внутренней памяти данных DM (Data Memory)

• 2 программируемых скоростных последовательных порта

• интервальный таймер

• 16-битный порт прямого доступа к внутренней памяти IDMA (Internal Direct Memory Access)

• 8-битный порт прямого доступа к внешней байтовой памяти BDMA (Byte Direct Memory Access) объемом до 4 Мбайт

• адресное пространство устройств ввода/вывода (2048 адресов)

• 4 сегмента по 8 К слов внешней оверлейной памяти

• внешние прерывания и программируемые выводы флагов

• режим пониженного энергопотребления (Pпот.< 1 мВт)

• расширенный набор инструкций

• производительность - 33.33 MIPS

• отдельный порт внутрисхемной эмуляции

Рис. 3.1.1. Графическое изображение цифрового сигнального процессора ADSP - 2181

Дополнительные сведения о процессоре ADSP - 2181.

Тип обратываемых данных - с фиксированной точкой.

Тактовая частота - 16,67 MHz.

Время одного командного цикла - 30 nS.

Последовательный порт - 2

Параллельный порт - 2 ( 1(IDMA) - 8-битный или 16-битный, 1 - 24-битный)

Напряжение питания - 3V, 5V.

Рассмотрим назначение выводов и принцип работы процессора:

D0 ... D23 - 24-разрядная шина данных. Используется совместно с

шиной адреса для загрузки программы из внешней памяти в DSP.

A0 ... A13 - 14-разрядная шина адреса.

Память программ : ADSP-2181 содержит 16Kx24 ОЗУ программ на кристалле. Память программ позволяет выполнять до двух обращений в каждом цикле, тогда все операции могут завершаться за один цикл.

Память данных : ADSP-2181 имеет 16,352 16-разрядных слова внутренней памяти данных.

Пространство байтовой памяти - двунаправленное, 8-разрядное, внешнее пространство памяти, используемое для хранения программ и данных. Доступ к байтовой памяти осуществляется через BDMA. Пространство байтовой памяти состоит из 256 страниц, каждая из которых имеет размер 16К x 8.Это позволяет использовать до 4М x 8 (32 мегабит) ПЗУ или ОЗУ без дополнительной логики. Все обращения к байтовой памяти имеют временные параметры, определяемые регистром BMWAIT.

Контроллер прямого доступа в байтовую память (BDMA) позволяет осуществлять загрузку и сохранение команд программы и данных, используя пространство байтовой памяти. Схема BDMA способна обращаться к пространству байтовой памяти в то время, как процессор работает и захватывает только один цикл DSP для перемещения 8-, 16- или 24-разрядного слова.

IAD0 ... IAD15 - 16-ти разрядная мультиплексированная шина данных/адреса порта IDMA.

Порт прямого доступа к внутренней памяти (IDMA) процессора ADSP-2181 является одним из новых устройств, существенно упрощающих построение интерфейса с HOST-процессором.

Рис. 3.1.2. Интерфейс работы порта IDMA с HOST-процессором.

Четыре входа управления IDMA предназначены для:

IS - выбор порта;

IAL - запись адреса ячейки памяти;

IRD - чтения данных через порт;

IWR - запись данных ;

IACK - Сигнал подтверждения доступа. Определяет завершение операций чтения/записи и готовность IDMA к следующей операции.

BMODE и MMAP

Выводы процессора BMODE и MMAP определяют режим загрузки и распределение (карту) памяти DSP. Для загрузки через внешнюю память BMODE=0 и MMAP=0. Загрузка состоит из следующих операций:

• Сброс процессора сигналом RESET

• Загрузка в Programm Memory и Data Memory кодов программы и данных, исключая ячейку PM(0x0000).

• Запись слова в ячейку PM(0x0000) для запуска загруженной программы.

IRQ2, IRQL1, IRQL0 и IRQE. TFS1/IRQ1, RFS1/IRQ0.

Аппаратные входы прерываний. При подаче на них низкого уровня сигнала управление передается соответствующей подпрограмме

Таблица 3.1 - Таблица прерываний.

RESET - при получении низкого уровня сигнала передается управление подпрограмме инициализации DSP. При этом происходит повторная загрузка программы из внешней памяти в DSP.

PWD - (power down) отключение питания.

XTAL, CLKIN - на них подается тактовая частота от кварца. В нашем случае 16,67 MHz.

PMS, IOMS,

BMS, DMS , CMS - Данные выводы служат для подключения и управления оверлейной памятью.

Конфигурация оверлейной памяти задается установкой управляющего сигнала CMS в регистре программируемых флагов и составного сигнала управления (Programmable Flag and Composite Select Control). Также можно использовать оверлейную память, как память данных.

Так как шина адреса ADSP-2181 имеет только четырнадцать разрядов, то для расширения адресного пространства оверлейной памяти используются флаги FL0, FL1, а также FL2 или PMS в зависимости от требуемой конфигурации.

Рисунок 3.3 Системный интерфейс ADSP 2181

Выбранный нами цифровой сигнальный процессор ADSP - 2181 способен выполнять следующие действия:

|  |  |
| --- | --- |
| За один цикл процессор ADSP-2181 может: | Это происходит в то время как процессор продолжает: |
| - генерировать следующий адрес программы | - получать и передавать данные через два последовательных порта |
| - Выбирать следующую команду | - получать и/или передавать данные через внутренний порт прямого доступа в память |
| - выполнить одно или два перемещения данных | - получать и/или передавать данные через порт прямого доступа в байтовую память |
| - модифицировать один или два указателя адреса данных | - Декрементировать таймер |
| - выполнить вычислительную операцию |  |

Это полностью удовлетворяет нашим требованиям, для обеспечения требуемой модуляции и реализации метода кодирования NBDP. А также фирма Analog Devices поставляет со своими процессорами мощные программные продукты для отладки и записи программ в DSP, что делает данный цифровой сигнальный процессор еще более приемлемым для нас.

3.2 Выбор кодека

Как было видно из пункта 3.1 , цифровой сигнальный процессор не занимается преобразованием аналогового сигнала в цифровой и наоборот, это делают АЦП и ЦАП. Вот таким комбинированным АЦП/ЦАП являются микросхемы CODEC.

Их как и DSP существует большое количество, но мы также остановимся на микросхемах фирмы Analog Devices. Т.к. в роли цифрового сигнального процессора нами выбран ADSP 2181, то выбираем звуковой кодек AD1847 с последовательным цифровым интерфейсом совместимым с ADSP 21xx.

Рисунок 3.4 - Графическое изображение Codec AD1847

Параметры AD1847:

Тип сигнала - моно/стерео

Преобразование- АЦП / ЦАП

Напряжение питания- + 5 V

Диапазон выходных частот- 20 Hz ... 20 kHz

Наличие фильтров: цифровой фильтр;

аналоговый фильтр НЧ;

Максимальная тактовая частота- 27 MHz

Аналоговый вход- 2

Вспомагательный аналоговый вход - 1

Аналоговый выход - 1

Рассмотрим назначение выводов и принцип работы кодека:

VCC - питание + 5 V. Источник питания тот же, что и ADSP - 2181.

GND - земля.

GNA - земля аналогового сигнала

SCLK - тактовый генератор последовательной передачи данных. (значение зависит от XTAL1,XTAL2 ) при установленном XTAL1 значение будет 12,288 MHz, при XTAL2 11,2896 MHz.

SDFS - синхронизация последовательных данных.

SDI, SDO - прием и передача данных из последовательного порта DSP. ( Serial Data Input и Serial Data Output ). Обмен данными может осуществляться как с DSP, так и любым HOST - процессором. Размер даных - 16 бит.

RST - при установке низкого уровня , происходит инициализация всех регистров начальными значениями. (RESET)

PWD - также установке низкого уровня , происходит инициализация всех регистров начальными значениями и перевод чипа в режим пониженного энергопотребления, при котором Vref и аналоговые выводы земли - отключены.

BM - при наличии на этот выводе высокого уровня сигнала , на шине устанавливается сигнал MASTER, и происходит передача данных в DSP по порту RXD0. В случае низкого уровня сигнала на шине устанавливается сигнал SLAVE, и происходит прием данных от DSP по порту TXD0.

CLKO - (clock output) выход тактового генератора. Значение зависит от XTAL1, XTAL2: при установленном XTAL1 значение будет 12,288 MHz, при XTAL2 16,9344 MHz.

Аналоговые выводы приема/передачи

LI1L - линейный вход 1 для левого канала

LI1R - линейный вход 1 для правого канала

LI2L - линейный вход 2 для левого канала

LI2R - линейный вход 2 для правого канала

AI1L - вспомагательный вход 1 для левого канала

AI1R - вспомагательный вход 1 для правого канала

AI2L - вспомагательный вход 2 для левого канала

AI2R - вспомагательный вход 2 для правого канала

LOL - линейный выход для левого канала

LOR - линейный выход для правого канала

X2O, X2I - от кварцевого резонатора 16,9344 MHz.

X1O, X1I - от кварцевого резонатора 24,576 MHz.

Исходя из этого выбираем кварцевый резонатор X2 с частатой 16, 9344 MHz, а X3 с частатой 24, 576 MHz.

Выбранные кварцевые резонаторы включаются по стандартной схеме с двумя параллельными конденсаторами по 18 pF.

Конденсаторы C31, C32, C36, C37 выбираем емкостью 18 pF.

FLTL - левый канал фильтра. Используется для подключения стандартного конденсатора 1 мкФ.

FLTR - правый канал фильтра. Используется для подключения стандартного конденсатора 1 мкФ.

Исходя из этого выбираем конденсаторы C40 и C41 емкостью 1 мкФ.

VRO - внешнее опорное напряжение. Величина 2,25 V. Запрещается подключение к данному выводу какой - либо нагрузки

VRI - внутренее опорное напряжение.

Рисунок 3.5 - Схема включения опорного напряжения

Из схемы включения видно, что выходное опорное напряжение используется для аналоговых сигналов. Исходя из данной стандартной схемы включения конденсаторы C46 и C47 выбираем емкостью 10 мкФ, а конденсатор C45 = 0.1 мкФ.

Рисунок 3.6 - Функциональная блок - схема кодека AD1847

В данном разделе были рассмотрены функции кодека ADSP 2181 , назначения выводов и принцип работы. Также были выбраны все необходимые элементы для стандартной схемы включения.

Выбраный кодек поностью совместим с ADSP 2181 , имеет 16-ти разрядный последовательный порт, чего вполне достаточно для реализации нашего устройства кодирования - декодирования информации.

Практически все рассмотренные функции данного кодека необходимы для реализации нашего устройства и будут задействованы.

3.3 Выбор драйвера интерфейса RS - 232

Драйверы интерфейса RS - 232 служат для согласования с host - компьютером использующим уровни сигнала RS - 232 и асинхронный последовательный порт.

Выбираем драйвер интерфейса RS - 232 фирмы Analog Devices ADM232A.

Рисунок 3.7 - Графическое изображение драйвера RS - 232 ADM232A

Основные параметры:

Скорость передачи - 200 kB/s.

Напряжение питания - 5 V.

2 TTL/CMOS входа (выхода) ( от DSP )

2 RS - 232 входа (выхода) ( от ЭВМ )

Выходной уровень сигнала - + 9 V ( на ЭВМ )

Входной уровень сигнала - + 30 V (от ЭВМ )

Корпус DIP 16.

Совпадение по выводам с MAX232A.

Диапазон рабочих температур: -40 0C ... +85 0C

Назначение выводов AD232A:

V+ - внутренняя генерация позитивного напряжения ( +10 V )

V- - внутренняя генерация негативного напряжения ( -10 V )

С1+ - подключение внешнего конденсатора 1 номиналом 0.1 мкФ ( плюсом к этому выводу)

С1- - подключение внешнего конденсатора 1 номиналом 0.1 мкФ ( минусом к этому выводу)

С2+ - подключение внешнего конденсатора 2 номиналом 0.1 мкФ ( плюсом к этому выводу)

С2- - подключение внешнего конденсатора 2 номиналом 0.1 мкФ ( минусом к этому выводу)

TR1IN - первый вход сигнала с TTL/CMOS уровнем от DSP. Внутри микросхемы находится резистор номиналом 400 кОм подключаемый к данному выводу и соединенный с Ucc = + 5 V.

TR2IN - второй вход сигнала с TTL/CMOS уровнем от DSP. Внутри микросхемы находится резистор номиналом 400 кОм подключаемый к данному выводу и соединенный с Ucc = + 5 V.

TR1OUT - первый выход сигнала с уровнем для интерфейса RS - 232. (уровень + 9 V).

TR2OUT - второй выход сигнала с уровнем для интерфейса RS - 232. (уровень + 9 V).

RC1IN - первый вход сигнала с уровнем от интерфейса RS - 232 из host - компьютера (ЭВМ). Внутри микросхемы находится резистор номиналом 5 кОм подключаемый к данному выводу и соединенный с землей.

RC2IN - второй вход сигнала с уровнем от интерфейса RS - 232 из host - компьютера (ЭВМ). Внутри микросхемы находится резистор номиналом 5 кОм подключаемый к данному выводу и соединенный с землей.

RC1OUT - первый выход сигнала с уровнем TTL/CMOS образованным из сигнала поступившего по интерфейсу RS - 232 из host - компьютера (ЭВМ).

RC2OUT - второй выход сигнала с уровнем TTL/CMOS образованным из сигнала поступившего по интерфейсу RS - 232 из host - компьютера (ЭВМ).

Рисунок 3.8 Функциональная блок - схема драйвера ADM232A

Рисунок 3.9 - Схема включения драйвера интерфейса RS - 232 AD232A

Исходя из данной стандартной схемы включения выбираем конденсаторы C25,C26,C27,C28 номиналом 0.1 мкФ.

Рассмотренный драйвер интерфейса RS-232 способен обеспечивать высокое быстродействие при работе с последовательным портом, что в нашем устройстве, которое оснащено скоростным DSP, очень важно, а также способен работать с сигналами ТТЛ-уровня, которые используются нашим DSP. Следовательно данный драйвер полностью подходит для обеспечения связи по последовательному порту с ЭВМ.

3.4 Выбор памяти с ультрафиолетовым стиранием (EPROM)

Микросхема памяти с ультрафиолетовым стиранием необходима в нашем устройстве для хранения в ней программы под цифровой сигнальный процессор, которая и будет реализовывать алгоритм кодирования - декодирования.

ADSP -2181 способен работать не более, чем с 4 Мб внешней памяти.

Рассмотрим микросхемы памяти серии AM27Cxxx:

Таблица 3.1 - Объем памяти EPROM серии AM27Cxxx.

|  |  |  |
| --- | --- | --- |
|  | Тип микросхемы | Объём памяти |
| 1 | AM27С256 | 32Кб |
| 2 | AM27С512 | 64Кб |
| 3 | AM27С010 | 128Кб |
| 4 | AM27С020 | 256Кб |
| 5 | AM27С040 | 512Кб |
| 6 | AM27С080 | 1Мб |

Т.к. выбранный нами цифровой сигнальный процессор

ADSP-2181 имеет расширенную систему команду, по сравнению со своими предшественниками, и хранит в памяти большое количество оперативной информации (стек, регистры, указатели и т.п.), то только для его нормальной работы необходим большой объем памяти.

А так как кроме того, мы проектируем большую программу по кодированию-декодированию информации, инициализации DSP и кодека, а также организации способа частотной модуляции, то дополнительно к этому нам еще требуется не менее 500 Kb.

Итого вобщем необходимо не менее 600 Kb.

Выбираем микросхему с максимальным объёмом памяти 1 Мб AM27C080.

Рисунок 3.10 - Графическое изображение EPROM AM27C080.

3.5 Выбор вспомагательных элементов схемы

Схема питания устройства:

На наше устройство должно подаваться питание от 8 до 10 вольт ( берется из набора стандартных питаний на судах дальнего плавания).

Микросхемы нашего устройства питаются от +5 V. Для этого выбираем

регулятор напряжения LM7805, на вход которого подается напряжение от 8 до 10 V, а на выходе получается +5V. Ниже представлена схема подачи питания и стандартная схема включения микросхемы LM7805.

Рисунок 3.11 - Схема подачи питания.

Для обеспечения защиты от перемены полярности на входе, устанавливаем диод D1 . В качестве его выбираем КД522А.

Конденсаторы С14 и С15 выбираем изходя из стандартной схемы включения равными 0.33 мкФ и 10 мкФ соответственно.

Резистор R18 = 100 kOm.

Рассмотрим характеристики выбранного регулятора напряжения LM7805:

Выходные напряжения: 5, 6, 8, 12, 15, 18, 24 V

Входные напряжения: 8-10, 17-19, 21-23 V

Рабочие температуры: 0 ... +125 0C

Рисунок 3.12 - График зависимости выходного напряжения от входного микросхемы LM7805.

Прочие элементы:

Для обеспечения фильтра по питанию каждой микросхемы выбираем соответствующее количеству выводов питания количество конденсаторов емкостью 0.1 мкФ: C4, C5, C6, C7, C8, C22, C23, C24, C25, C26, C27, C30, C33.

Также выбираем резисторы подтяжки от питания к выводам микросхем равными 10 кОм: R4, R6, R7, R8, R9.

Прочие элементы входят в стандартные схемами включения микросхем и их номиналы указаны в перечне элементов.

4. Разработка алгоритма программы

4.1 Блок инициализации

В состав нашего устройства входят программируемые микросхемы DSP и Codec. Для каждой из них необходимо выполнить инициализацию внутренних переменных, регистров и флагов. Также , как рассматривалось в главе 2, наше устройство работает с внешних host-компьютером - терминалом и для начала работы необходимо дать ему команду инициализации. Для начала работы с терминалом , также необходимо инициализировать переменные и регистры используемые интерфейсом приема/передачи данных. Как было рассмотренно в главе 2, для кодирования информации используется структура судового телеграфа NBDP ( narrowband printing) узкополосное буквопечатанье. Для инициализации данной структуры используется таблица соответствия букв и цифр определенной комбинации 1 и 0.

Рисунок 4.1. - Блок инициализации.

4.2 Интерфейс приема / передачи

Для организации работы с терминалом используется набор команд передаваемых по последовательному порту компьютера. Программа под терминал нами рассматриваться не будет, но связь с ней с помощью нашего устройства будет показана и использованна.

При появлении в host-компьютере данных которые требуется принять, терминал дает запрос на передачу. Также при появлении данных в нашем устройстве, которые требуется передать , в терминал посылается запрос на прием терминалом данных. В случае если терминал готов принять наши данные , устанавливается флаг готовности передачи в терминал. Если же наше устройство готово принять данные от терминала, то также устанавливаеися флаг готовности приема от терминала.

Рисунок 4.2. - Блок - схема запроса на прием / передачу от терминала

Далее, в случае установки флагов приема / передачи, выполняется соответственно прием или передача данных.

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

Рисунок 4.3. - Блок - схема приема данных от терминала

При передаче данных в терминала происходит демодуляция поступивших данных , раскодирование их по алгоритму NBDP и перевод в формат ASCII с последующей передачей по последовательному порту в терминал.

Рисунок 4.4. - Блок - схема передачи данных в терминал.

Далее после окончания приема или передачи данных , в терминал передается команда "устройство свободно", что разрешает дальнейшие запросы на обмен данными.

Перезагрузка программы в память и инициализация происходят при нажатии кнопки RESET.

Полная блок-схема алгоритма предоставленна в приложении.

Данный алгоритм представляет собой общий принцип работы программы. Более конкретная его реализация с учетом особенностей выбранной технологической базы будет рассмотренна в следующей главе.

5. Разработка программного обеспечения

Программное обеспечение будет разрабатываться для цифровго сигнального процессора фирмы Analog Devices ADSP -2181. Для этого будем использовать программное обеспечение фирмы Analog Devices, позволяющее создавать и тестировать программы под DSP.

Разработку программного обеспечения будем производить на языке ассемблер под ADSP - 2181 и Си. Язык Си напрямую не поддерживается DSP, но в комплекте программ фирмы Analog Devices имеется конвертер программ написанных на языке Си, в формат машинных кодов DSP. Также есть возможность создания проектов включающих программы написанные под DSP и под Си, и ссылаться на функции друг друга с помощью EXTERNAL (внешних) процедур и переменных.

Будем использовать следующее программное обеспечение для создания нашей программы:

Assembler - компоновщик ассемблерных программ \*.dsp в \*.obj

Linker - компилятор в код DSP

ADSP-2181 Simulator - эмулятор ADSP - 2181

Minitor - программа для записи программы в DSP

C Compiler - компилятор языка Си

C Runtime Libraries - библиотеки языка Си

C Debugger - отладчик для программ на Си

Cconverter - конвертер программ на Си в код DSP

Разработка программного кода находится в приложении 1.

## 6. Технико-экономический расчет

В данном дипломном проекте разработано устройство кодирования - декодирования информации на цифровом сигнальном процессоре. Наше устройство выполняет функции радиомодема и использует частотную модуляцию по протоколу обмена данными морского телеграфа. Эта система является конструктивно законченным устройством, поэтому целесообразно рассчитать затраты на расходные материалы и производство устройства , а также рассчитать срок окупаемости предложенной системы.

6.1 Расчёт трудоёмкости выполнения опытно-конструкторской работы

В технологии ОКР сочетаются работы по разработке необходимого комплекта конструкторской документации и работы по изготовлению, отладке и испытаниям опытного образца.

Для определения трудоёмкости разработки комплекта КД используем перечень документов, разрабатываемых в курсовой работе и типовые нормы времени на разработку конструкторской документации. Кроме этого, учитываем затраты времени на участие в испытании и отладке опытного образца и корректировку технической документации после испытания опытного образца.

Трудоёмкость разработки конструкторских документов рассчитываем по формулам:

Где t pi , t kki t тki , t нki - соответственно нормы времени на разработку, конструкторский контроль, технологический контроль и нормоконтроль КД i-го наименования.

t нрi , 0.7, 0.2, 0.15(0.2) – типовые нормы времени на разработку, конструкторский и технологический контроли, нормоконтроль единицы объёма КД i-го наименования;

Кфi – поправочный коэффициент к норме времени в зависимости от фактического формата КД i-го наименования;

Кс – поправочный коэффициент к норме времени в зависимости от типа производства (1,0 – единичное; 1,1 – серийное; 1,2 – массовое);

Кмi – поправочный коэффициент к норме времени в зависимости от масштаба исполнения КД i-го наименования;

Каi – поправочный коэффициент к норме времени на разработку КД ЭА, не имеющей аналога.

Трудоёмкость разработки печатной платы рассчитываем по формуле:

 ,

Где N- количество функциональных узлов на печатной плате.

Расчёт трудоёмкости разработки конструкторской документации выполняется по форме, приведенной в таблице 6.1.1.

Таблица 6.1.1-1 — Расчёт трудоёмкости КПП

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Наименование и  | Tpi | Ni | Kфi | Kc | Kmi | Kai | Расчет |
| Характеристики КД |
| Схем электрическая структурная МПСУ СО | 36,96 | 1А1 | 1, 6 | 1, 1 | 1 | 1 | Тр=65,05 |
| 0,7 | 1А1 | 6, 4 | 1, 1 | 1 | 1 | Тк=1,23 |
| 0,15 | 1А1 | 6, 4 | 1, 1 | 1 | 1 | Тн=0,26 |
| Схема электрическая принципиальная устройства | 46,2 | 1А1 | 1, 6 | 1, 1 | 1 | 1 | Тр=81,31 |
| 0,7 | 1А1 | 6, 4 | 1, 1 | 1 | 1 | Тк=1,23 |
| 0,15 | 1А1 | 6, 4 | 1, 1 | 1 | 1 | Тн=0,26 |
| Структурна схем устройства  | 5,3 | 1А1 | 1,6 | 1, 1 | 1, 1 | 1 | Тр=9,4 |
| 0, 7 | 1А1 | 6, 4 | 1, 1 | 1, 1 | 1 | Тк=5 |
| 0, 15 | 1А1 | 6, 4 | 1, 1 | 1, 1 | 1 | Тн=1,1 |
| Чертеж ПП узла схемы | 122 | 1А1 | 1 | 1 | 1 | 1 | Тр=147,6 |
| 0, 7 | 1А1 | 6, 4 | 1, 1 | 1, 1 | 1 | Тк=0,85 |
| 0, 2 | 1А1 | 6, 4 | 1, 1 | 1, 1 | 1 | Ттк=0,24 |
| 0, 15 | 1А1 | 6, 4 | 1, 1 | 1, 1 | 1 | Тнк=0,18 |
| Сбор. чертеж ПП узла схемы | 36,2 | 1А1 | 1 | 1,1 | 1,1 | 1 | Тр=43,8 |
| 0,7 | 1А1 | 6,4 | 1,1 | 1,1 | 1 | Тк=0,85 |
| 0,2 | 1А1 | 6,4 | 1,1 | 1,1 | 1 | Ттк=0,24 |
| 0,15 | 1А1 | 6,4 | 1,1 | 1,1 | 1 | Тнк=0,18 |

Таблица 6.1.1 - 2 — Расчёт трудоёмкости КПП

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Наименование и характеристики КД | Tpi | Ni | Kфi | Kc | Kmi | Kai | Расчет |
| Алгоритм и временные диаграммы | 3,6 | 1А1 | 6,4 | 1,1 | 1 | 1 | Тр=25,34 |
| 0,2 | 1А1 | 6,4 | 1,1 | 1 | 1 | Тк=4,93 |
| 0,15 | 1А1 | 6,4 | 1,1 | 1 | 1 | Тнк=1,06 |
| Вед-ть специфик. | 3,6 | 3А4 | 1 | 1,1 | 1 | 1 | Тр=11,88 |
| 0,2 | 3А4 | 1 | 1,1 | 1 | 1 | Тнк=0,5 |
| Техническое Задание | 3 | 3А4 | 1 | 1,1 | 1 | 1 | Тр=9,9 |
| 0,2 | 3А4 | 1 | 1,1 | 1 | 1 | Тнк=0,5 |
| Пояснительная  | 3,1 | 90А4 | 1 | 1,1 | 1 | 1 | Тр=148,5 |
|  записка | 0,2 | 90А4 | 1 | 1,1 | 1 | 1 | Тнк=14,85 |
| Экономическая таблица | 2 | 1А1 | 6,4 | 1,1 | 1 | 1 | Тр=14,08 |
| 0,2 | 1А1 | 6,4 | 1,1 | 1 | 1 | Тнк=1,06 |

Трудоёмкость разработки комплекта КД и загрузка исполнителей в рабочих днях определяется по формуле:

Тр.д.=Σti / 8 , Тк.д.=Тр.д./0,73

Где Σti – суммарная трудоёмкость в часах;

– продолжительность рабочего дня в часах;

0,73 – коэффициент перевода в календарные дни.

По каждому виду конструкторской документации определяется квалификационный уровень исполнителей.

Распределение работ и расчёт загрузки исполнителей производится по форме, приведенной в таблице 6.1.2.

Таблица 6.1.2 — Расчёт загрузки исполнителей

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование КД | ОбъемКД | Трудоемкость(час) | Исполнители |
| Разработка | КК | ТК | НК |
| К III | K I | Технолог | Инж. Стандарт |
| V | t | V | t | V | t | V | t |
| Схема структурная МПСУ СО | А1 | 66,55 | А1 | 65,05 | А1 | 1,23 |  |  | А1 | 0,26 |
| Схема структурная СДК | А1 | 15,5 | А1 | 9,4 | А1 | 5 |  |  | А1 | 1,1 |
| Схема принципиальная | А1 | 82,81 | А1 | 81,31 | А1 | 1,23 |  |  | А1 | 0,26 |
| Чертеж ПП | А1 | 148,89 | А1 | 147,6 | А1 | 0,85 | А1 | 0,24 | А1 | 0,18 |
| Сбор. ч-ж ПП | А1 | 45,07 | А1 | 43,08 | А1 | 0,85 | А1 | 0,24 | А1 | 0,18 |
| Алгоритм и врем.диаграммы | А1 | 39,33 | А1 | 35,36 | А1 | 4,93 | А1 |  | А1 | 1,06 |
| Вед-ть специфик. | 3А4 | 12,38 | 3А4 | 11,88 |  |  |  |  | 3А4 | 0,55 |
| ТЗ | 3А4 | 10,4 | 3А4 | 9,9 |  |  |  |  | 3А4 | 0,55 |
| Пояснительная записка | 90А4 | 163,35 | 90А4 | 148,5 |  |  |  |  | 90А4 | 14,85 |
| Экономическая таблица | А1 | 15,44 | А1 | 14,08 |  |  |  |  | А1 | 1,06 |
| ВСЕГО часов |  | 532,8 |  | 501,0 |  | 12,9 |  | 0,5 |  | 19,7 |
|  рабочих дней |  | 66,6 |  | 62,6 |  | 1,6 |  | 0,06 |  | 2,5 |
|  календ. дней |  | 91,2 |  | 85,8 |  | 2,19 |  | 0,08 |  | 3,42 |

6.2. Определение цены научно-технической продукции

Цена научно-технической продукции (выполнения ОКР) определяется по формуле:

Цокр=Спл+Ппл+НДС

Цокр= 1257,5+377,25+326,95=1961,7 грн

Где Спл – плановая себестоимость выполнения ОКР;

Ппл – плановая прибыль:

Ппл=0.3Спл

Ппл=0.3\*1257,5=377,25 грн

Где 0.3 – норматив, учитывающий предельный уровень рентабельности, который установлен действующим законодательством для научно-технической продукции.

НДС=0.2(Спл+Ппл)

НДС=0.2(1257,5+377,25)=326,95 грн

Определим себестоимости ОКР, используя метод удельных весов. Для этого необходимо определить сумму основной заработной платы по всем исполнителям, участвующим в данной ОКР.

Размер заработной платы за выполненный объём работы по каждому исполнителю определяется по формуле:

Где Трд - трудоемкость выполнения ОКР каждым исполнителем ;

ЗПдн - заработная плата за один день работы исполнителя ,

где

Где ЗП мес - месячный должностной оклад .

 ,

Где ЗП min - ставка минимальной заработной платы (с 1.04.2000г. равна 90 грн.);

К тар - тарифный коэффициент исполнителя;

К пов - коэффициент повышения ставок и окладов;

Расчет заработной платы производится по формам, приведенным в таблицах 6.2.1 и 6.2.2 .

Таблица 6.2.1 — Расчет месячного должностного оклада

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Исполнитель | Разрядисполнителя | СтавкаЗПмингрн. | Тарифныйкоэффициент | КоэффициентПСиО | Месячныйдолж.окладгрн. |
| K III | 15 | 74 | 4,01 | 0,58 | 180 |
| K I | 18 | 74 | 5,34 | 0,67 | 270,7 |
| Технолог | 17 | 74 | 4,85 | 0,67 | 250,5 |
| Инж.стандарт. | 14 | 74 | 3,64 | 0,58 | 160,2 |

Таблица 6.2.2 — Расчет основной заработной платы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнители | Трудоемкостьраб.дни. | Месячныйдолж.окладгрн. | Дневнаяставкагрн. | СуммаОЗПГрн. |
| K III | 61,5 | 180 | 7,05 | 433,57 |
| K I | 1,8 | 270,7 | 10,6 | 19,09 |
| Технолог | 0,06 | 250,5 | 9,8 | 0,59 |
| Инж.стандарт. | 2,5 | 160,2 | 6,3 | 15,75 |
|  |  |  |  | 469,00 |

Для определения суммарных затрат на ОКР составляется калькуляция плановой себестоимости, которая используется для планирования и учета затрат на ОКР .

Таблица 6.2.3 — Калькуляция плановой себестоимости (5)

|  |  |  |
| --- | --- | --- |
| Статья затрат | СуммаГрн. | Расчет |
| 1. Материалы | 69 | Зм=(Ам/Ао)\*Зо |
| 2. Спецоборудование | 10,62 | Зоб=(Аоб/Ао)\*Зо |
| 3. Основная зар.плата | 469,00 | Зо (см. табл. 4) |
| 4. Дополнт. зар.плата | 46,9 | Зд=0,1\*Зо |
| 5. Отчисления в ФСС | 19,8 | Зсс=0,04(Зо+Зд) |
| 6. Отчисления в ПФ | 158,89 | Зпенс=0,32(Зо+Зд) |
| 7. Отчисления в ФБ | 7,5 | Збез=0,015(Зо+Зд) |
| 8. Отчисления в Фстроительства и дорог | 4,96 | Зсс=0,09(Зо+Зд) |
| 9. Отчислние в ИФ | 44,7 | Зин=0,1(Зо+Зд) |
| 10. Коммунал. Налог | 3,14 | Зком=17\*0,05(Зо+Зд)/165 |
| 11. Ком. Расходы | 0 | Зк=Ак\*Зо |
| 12. Услуги стор.орган. | 0 | Зус=Аус\*Зо |
| 13. Пр. прям.расходы | 45,14 | Зпр=Апр\*Зо |
| 14. Накладные расходы | 397,2 | Знр=Анр\*Зо |
| Планова себестоимость | 1276,84 | Спл=Σ(Зi) |

Затраты по статьям : материалы, спецоборудование, дополнительная заработная плата, командировочные расходы, услуги сторонних организаций, прочие прямые расходы и накладные расходы определяются с использованием метода удельных весов. В формулах расчета затрат использованы следующие условные обозначения: Ам , Аоб , Ао , Ак , Аус , Апр , Анр - соответственно удельный вес затрат на материалы , спецоборудования , основную зарплату , командировочные расходы , услуги сторонних организаций, прочие прямые расходы и накладные расходы в себестоимости КПП аналогичных изделий (Ам = 13, Аоб = 2, Ао = 70, Ак = 0,15, Аус = 0,1, Апр = 0,1, Анр = 0,8) ;

Расчет себестоимости и цены проектируемой ЭА

Определим укрупненными методами первоначальную себестоимость запроектированной радиоэлектронной аппаратуры. Покупные фабрикаты и покупные изделия представлены в перечне элементов, а общая стоимость элементов представлена в таблице 6.2.1.

Таблица 6.2.4. — Расчет стоимости покупных полуфабрикатов , комплектующих элементов и изделий.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Расходы на доставку материалов. |  |  |  | 50 |
| Всего |  |  |  | 350 |

Калькуляция плановой себестоимости навой ЭА составляется с использованием метода удельных весов по форме, приведенной в таблице 6.2.5 .

Таблица 6.2.5. — Калькуляция плановой себестоимости проектируемого изделия

|  |  |  |
| --- | --- | --- |
| Статья затрат | Суммагрн. | Расчет |
| 1. Сырье и материалы | 108,4 | Зм=(Ам/Апи)\*Зпи |
| 2.Покупные п-ф, к-щие изд.  | 350 | Зпи (см табл. 6) |
| 3. Основная зар.плата | 33 | Зо=Зпи\*Ао/Апи |
| 4. Дополнт. зар.плата | 3,3 | Зд=0,1\*Зо |
| 5. Отчисления в ФСС | 1,45 | Зсс=0,04(Зо+Зд) |
| 6. Отчисления в ПФ | 11,6 | Зпенс=0,32(Зо+Зд) |
| 7. Отчисления в ФБ | 0,5 | Збез=0,015(Зо+Зд) |
| 8. Отчисления в строит. и дорог | 0,36 | Зсс=0,09(Зо+Зд) |
| 9. Отчислние в ИФ | 3,27 | Зин=0,1(Зо+Зд) |
| 10. Коммунал. Налог | 0,06 | Зком=17\*0,05(Зо+Зд)/165 |
| 11. Общепроизводств. р-ды | 43,56 | Зпроиз=Апр\*Зо=(1,3…2,5)Зо |
| 12. Общехоз.р-ды | 25,41 | Зобщ=Аобщ\*Зо=(0,75…0,8)Зо |
| 13 . Произв.себест. | 560,91 | Спр=Σ(Зi) |
| 14. Коммерческие расходы | 112,18 | Зком=0,2\*Зо |
|  Полная себестоимость | 703,1 | Спол=Спр+Зком |

В приведенных расчетах использованы следующие условные обозначения: Ам =23, Апи = 70, Ао = 7, - соответственно удельный вес затрат на материалы, покупные полуфабрикаты, основную зарплату.

Цена ЭА определяется по формуле:

Где Сполн - полная себестоимость изделия ;

Пн - нормативная прибыль ;

НДС - налог на добавленную стоимость.

Пн= 211 грн.

где ρ - норматив рентабельности продукции по отношению к полной себестоимости (30%).


###### НДС = 175 грн

Тогда: Ц = 703,1 + 211 + 182,9 = 1097 грн

# 6.3 Расчет экономического эффекта

# Определение расчетного периода

Продолжительность расчетного периода зависит от длительности этапов конструкторской подготовки производства (КПП), технологической подготовки производства (ТПП), организационной подготовки производства (ОПП) и производства.

Продолжительность конструкторской подготовки производства определяется по следующей формуле:

# Ткпп = Ткр / (8.0\*0.73\*360) [год]

где: Ткр - длительность критического пути сетевого графика в часах;

8.0 - продолжительность трудового дня в часах;

360 - норматив перевода календарных дней в календарное количество лет;

0.73 - коэффициент перевода рабочих дней в календарные.

# Ткпп = 491,8 / (8.0\*0.73\*360) = 0,23 года

Продолжительность технологической подготовки производства определяется по следующей формуле:

Tтпп= αтпп \* Ткпп

где αтпп - коэффициент, характеризующий продолжительность технологической подготовки производства в сравнении с продолжительностью КПП. Выбирается из следующих соображений:

для проектирования простых РЭИ - 0.5;

для средней сложности -1.0;

для сложных сборочных единиц и высокой плотности компоновки РЭИ коэффициент составляет - 1.5.

Tтпп:= 1 \* 0,23=0,23 года

Продолжительность организационной подготовки производства определяется по следующей формуле:

Топп= αтпп \* (Ткпп + Ттпп)

Где αтпп - коэффициент, характеризующий продолжительность организационной подготовки производства в сравнении с продолжительностью КПП и ТПП. α принимает следующие значения:

для простых - 0.8;

для средней сложности - 1.2;

для сложных - 1.5.

Топп= 1,2\* 0,46 = 0,55 года

В расчетах экономического эффекта разновременные затраты и результаты приводятся к единому моменту времени – расчетному году. За начальный год расчетного периода принимается год начала финансирования работ по созданию нового изделия. - 2000 год. В качестве конечного года расчетного периода принимаем год снятия изделия с производства - 2003 год.

Расчетный период составит 4 года.

Коэффициент приведения разновременных затрат и результатов к расчетному году определяется по формуле:

αi=(1+Eн)Тр-Ti

где: Ен - норматив эффективности капитальных вложений, в расчетах принимаем Ен=0.1 ;

Тр - расчетный год расчетного периода ;

Ti - i-й год расчетного периода .

Результаты, характеризующие расчетный период, представляются в форме таблицы 6.3.1.

# Таблица 6.3.1 — Характеристики расчетного периода

|  |  |  |  |
| --- | --- | --- | --- |
| Этапы разработки и производства | Продолжит. этапа,Лет | Расчетный период — 4 года | Всегошт |
| 2000 | 2001 | 2002 | 2003 |
| КПП | 0.23 | 0.3 |  |  |  |  |
| ТПП | 0.23 | 0.2 | - |  |  |  |
| ОПП | 0.55 | 0.4 |  |  |  |  |
| Производство | 3 |  | 500 | 1000 | 1500 | 3000 |
| Коэффициент, учитывающий фактор времени | αi | 1 | 0.91 | 0.83 | 0.75 |  |

# Расчёт стоимостной оценки результатов

Стоимостная оценка результатов за расчетный период определяется по формуле:

где Ai — объём продаж ЭА в i-ом году расчетного периода в натуральном выражении;

Цi - цена реализации единицы продукции в i-ом году расчётного периода;

αi - коэффициент приведения разновременных затрат к сопоставимым величинам.

При выборе метода ценообразования в условиях рыночной экономики руководствуются стратегией прочного внедрения на рынок, которая предусматривает установление очень низкой цены на изделие в надежде на привлечение внимания покупателей и завоевания большей доли рынка. Установлению низкой цены благоприятствуют следующие условия:

* рынок очень чувствителен к ценам, и низкая цена способствует его расширению;
* с ростом объемов производства его издержки, а также издержки реализации сокращаются;
* низкая цена привлекательна для существующих и потенциальных конкурентов.

Используя метод ценообразования "средние издержки + прибыль" цена реализации единицы продукции в i-ом году расчётного периода определяется по формуле:

Ц=Сi +Пi+НДСi

где: Сi - полная себестоимость единицы продукции в i-ом году;

Пi - планируемая прибыль;

НДСi - налог на добавленную стоимость в цене реализации изделия в i-том году.

Таким образом, цена определяется на основе себестоимости и единицы прибыли, причём себестоимость единицы продукции в первом году расчётного периода определяется укрупненным методом, для чего на основе принципиальной схемы изделия составляется перечень покупных элементов и комплектующих изделий и определяется их стоимость. Результат расчетов сведены в таблице 6.2.4.

Калькуляция плановой себестоимости нового изделия рассчитывается с использованием метода удельных весов прямых затрат и структуры себестоимости аналогичной продукции по форме, приведенной в таблице 6.2.5.

Размер планируемой прибыли на единицу изделия:

Пi=Сi\*р

где

Сi - полная себестоимость изделия;

р - планируемый уровень рентабельности производства продукции (г=0.25).

# П= 703,1\* 0.3 =210,9 грн

Оптовая цена единицы изделия в i-том году:

Цоi=Сi+Пi

Для 2001 года оптовая цена будет: Цоi2001 = 914 грн

Цена реализации единицы продукции:

Црi=Цоi\*1.2

где коэффициент 1.2 обусловлен налоговой ставкой (квотой) по НДС

Для 2001 года цена реализации будет: Цр2001 = 1097 грн

При наличии роста объема производства в последующие годы расчетного периода себестоимость изделия:

Сi\*

Где Аi - объем реализации в натуральном выражении в i-том году;

Ai+1 - тоже для (i+1) года.

Коэффициент 0.23 характеризует влияние роста объема производства на себестоимость.

При уменьшении объема производства себестоимость принимается равной себестоимости предыдущего года.

По аналогии первого года определяется прибыль, цена оптовая и цена реализации в последующие годы расчетного периода:

С2002 = 703,1 \* (500 /1000)0.23 = 599,5 грн

П2002 = 179,85 грн,

Цо2002 = 779,35 грн,

Цр2002 =935,22 грн.

С2003 = 599,5 \* (1000 /1500)0.23 = 546,1 грн

П2003 = 163,83 грн,

Цо2003 = 709,93 грн,

Цр2003 =851,92 грн.

Стоимостная оценка результатов за расчетный период определена в табл. 6.3.2.

# Таблица 6.3.2 — Стоимостная оценка результатов

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование показателя | Условное обозначение | Годы производства | Всего |
| 2000 | 2001 | 2002 |  2003 |
| 1. Объем продаж, шт. | Ai | 0 | 500 |  1000 |  1500 | 3000 |
| 2. Цена реализации, грн. | Црi |  | 1097 |  935,22 |  851,92 |  |
| 3. Стоимостная оценка результатов, грн. | Ai\*Црi тыс.грн. |  | 548,5 |  935,22 |  1277,88 | 2761,6 |
| 4.Коэффициент, учитыв. фактор времени. | αi | 1 |  0.91 |  0.83 |  0.75 |  |
| 5. Стоимостная оценка результата с учетом фактора времени, грн. | Ai\*Цpi\* αiРт |  | 499,1 |  776,2 |  958,41 | 2633,71 |

# Расчет стоимостной оценки затрат

За расчетный период стоимостная оценка затрат определяется по формуле:

Где Кi - стоимостная оценка единовременных затрат в i-том году расчетного периода;

Лi - остаточная стоимость производственных фондов, высвобождаемых в i-том году расчетного периода;

Ui - стоимостная оценка текущих затрат и платежей в i-том году расчетного периода;

Единовременные затраты определяются по формуле:

Ki = Ккпп + Ктпп + Копп + Кобфi + Кофi

Ккпп, Ктпп. Копп - соответственно затраты на КПП, ТПП, ОПП;

Кобф - стоимость оборотных фондов в i-том году;

Коф - стоимость основных фондов в i-том году.

Сметная стоимость пакета КД, т.е. затраты на КПП с учетом НДС, определяется укрупненным методом по формуле:

Ккпп=(Зо/Аокпп)\*(1+R)+НДС

Где Зо - основная заработная плата разработчиков;

R - плановый уровень рентабельности (R = 0.25);

Аoкпп - удельный вес основной заработной платы в себестоимости КПП (Аокпп=0.3);

НДС - налог на добавленную стоимость:

НДС=(Зо\*(1+R)\*0.2)/Аокпп

Где 0.2 - налоговая ставка НДС.

Основная заработная плата разработчиков рассчитывается по формуле:

Где 3min - минимальная месячная заработная плата по действующему законодательству;

Ктарi - разрядный коэффициент по тарифной сетке;

Кпзпi - коэффициент повышения заработной платы;

Td - трудоемкость в месяцах.

### Td=Ti/8\*25.4

Где Ti - суммарная трудоемкость по видам работ (разработки, КК, ТК, НК) в часах,

25.4 — коэффициент пересчета рабочих дней в месяцы. Расчет основной заработной платы исполнителей проекта приведен в табл. 6.2.2.

# Сметная стоимость пакета КД, с учетом НДС

Затраты на КПП: Ккпп = 501\*(1 +0.20)/0.3+501\*(1 +0.20)\*0.2/0.3=2070,8 грн.

Затраты наТПП:

# Ктпп=Ккпп\*атпп

Где Ккпп - затраты на КПП с НДС,

агпп — коэффициент, характеризующий удельный вес ТПП по сравнению с затратами на КПП (атпп = 1,2).

Ктпп = 2484,96 грн.

Затраты на ОПП:

Копп =(Ккпп + Ктпп) аопп

Где аonn — коэффициент, учитывающий удельный вес зарплаты на ОПП по сравнению с Ккпп и Ктпп (аопп = 1,2).

Копп = 5466,9 грн.

Стоимость основных производственных фондов определяется по формуле:

Коф=ΔАi\*Цоi\*Ко\*Ке\*Кт

Где ΔАi - прирост объема производства в натуральном выражении в i-том году расчетного периода;

Цоi - оптовая предварительная цена на изделие в i-том году, грн.;

Ко - коэффициент, учитывающий объемы производства ( Ко= 0.105 - для серийного, 1.0 - для массового);

Ке - коэффициент, характеризующий фондоемкость (Кс = 1);

Кт - коэффициент, учитывающий тип производства ( Кт =1 - для серийного , 0.75 - для массового).

В 2001 и 2002 годах прироста производства нет, поэтому нет и затрат нa прирост основных фондов.

Kocф2001 = 500\*875\*0,5\*1\*1 = 218,75 тыс.грн

Коф2002 = (1000-500)\*796,25\*0,5\*1\*1 = 199,1 тыс.грн

Коф2003 = (1500-1000)\*745,3\*0,5\*1\*1 = 186,3 тыс.грн

Затраты оборотных фондов по годам расчетного периода:

Кобфi = Кофi\*0.3

Текущие затраты и платежи в бюджет по годам:

Ui = Спрi – Аоi + НДСдiб + Нпрiб

Где Cпрi - себестоимость объема продаж в i-том году, грн;

НДСдiб - налоговое обязательство в i-том году, (т.е. НДС, перечисленный в бюджет, грн);

Нпрiб - налог на прибыль, перечисленный в бюджет в i-том году расчетного периода, грн.

Себестоимость объема продаж (валовые издержки) по годам расчетного периода:

Cпр=Ci\*Ai

Где Ci - себестоимость единицы продукции в i-том году расчетного периода;

Ai - объем реализации (продаж) в i-том году расчетного периода.

Cпр2001 = C2001\* A2001 = 703,1\*500 =351,55 тыс. грн

Cпр2000 = C2002\* A2002 = 599,5\*1000 = 599,5 тыс. грн.

Cпр2003 = C2003\* A2003 = 546,1\*1500=819,15 тыс. грн.

Амортизационные отчисления на реновацию определены по формуле:

Аоi=Коф\*(На/100)

где Коф – основные фонды, используемые при производстве продукции в і-ом году.

Ао2001=435,5\*0,1=43,75 тыс. грн

Ао2002=(435,5+199,15)\*0,1=63,66 тыс. грн

Ао2001=(435,5+186,3+199,1)\*0,1=82,3 тыс. грн

Налоговое обязательство, перечисляемое в бюджет, в i-том году расчетного периода:

# НДСдiб = НДСiн - НДСiу

Где НДСiн - налоговое обязательство, полученное в i-том году расчетного периода, грн;

НДСiу - налоговый кредит, оплаченный в i-том году расчетного периода, грн.

Налоговое обязательство по годам расчетного периода:

НДСiн = НДСi\*Аi

Где НДСi - налоговое обязательство единицы продукции в i-том году, грн.

НДС н2001 = 0,175\*1000=175 тыс. грн

НДС н2002 = 0,159\*1500=238,9 тыс.грн

НДС н2003 = 0,149\*2000=298 тыс.грн

НДС н2002 =15620 грн

Налоговый кредит в i-том году расчетного периода:

НДСiу = (3ки + Зом ) \* Аi \* 0.2

Где Зки - стоимость комплектующих изделий и полуфабрикатов в i-том году расчетного периода, грн.,

Зом - стоимость материалов в i-том году, грн. (см. табл. 7).

НДСу2000 = 0 грн

НДСу2001 = (108,4+330) \* 1000 \* 0.2 = 87,68 тыс.грн

НДСу2002 = (108,4+330) \* 1500 \* 0.2 = 131,52 тыс.грн

НДСу2002 = (108,4+330) \* 2000 \* 0.2 = 175,4 тыс.грн

НДС бд2001 = 175 – 87,68 = 87,32тыс. грн

НДС бд2002 = 238,9 – 131,52 = 107,38 тыс.грн

НДС бд2003 = 298 – 175,4 = 122,6 тыс.грн

Налог на прибыль в i-том году расчетного периода:

Нпрiб = 0.3 \* Пi

где Пi - прибыль в i-том году расчетного периода.

Пi=СВДi – ВИi - Аoi

СВДi - скорректированный валовый доход в i-том году расчетного периода;

ВИi - валовые издержки в i-том году расчетного периода;

Аoi – сумма амортизационных отчислений в i-том году расчетного периода.

СВДi= ВДi - НДСi

где ВДi - валовый доход в i-том году расчетного периода.

Следовательно:

Пi=Ai\* (Цoi – Сi) – Аoi

Перечисление НДС на стадии научных работ:

НДСн = 0 - ( НДСкпп + НДСтпп + НДСопп)

П2001 = 500(875 – 673,1) – 43750 = 158,15 тыс. грн

П2001 = 1000(796,25 – 612,5) - 63660 = 212 тыс. грн

П2002 = 1500(745,3 – 573,3) - 82300 = 261,7 тыс. грн

Н бпр2001 = 0.3 \* 158,15 = 47,4 тыс.грн

Н бпр2002 = 0.3 \*212 = 63,6 тыс.грн

Н бпр2001 = 0.3 \* 261,7 = 78,5тыс.грн

U2001 = 764,07 тыс.грн

U2002 = 1026,07 грн

U2003 = 1265,4 грн

Стоимостная оценка затрат в грн., определенная за расчетный период, приведена в табл. 6.3.3:

# Таблица 6.3.3 — Стоимостная оценка затрат

|  |  |  |
| --- | --- | --- |
| Наименование показателя | Услово6означ. | Расчетный период |
| 2000 | 2001 | 2002 | 2003 |
| Затраты на КПП | Ккпп | 1,635 | — | — | — |
| Затраты на ТПП | Ктпп | 1,962 | — | — | — |
| Затраты на ОПП | Копп | 4,316 | — | — | — |
| Затраты на ОФ | Кофi |  | 437,5 | 199,1 | 186,3 |
| Затраты на Об.Ф | Кобфi |  | 131,25 | 59,8 | 55,9 |
| Всего единовременных за-грат | ΣКi | 7,913 | 568,75 | 258,9 | 242,2 |
| Коэффициент, учитывающий фактор времени | аi | 1 | 0.91 | 0.83 | 0.75 |
| Всего единовременных затрат с учетом фактора времени | ΣКiai | 7,9 | 517,5 | 214,9 | 180 |
| Остаточная стоимость ОФ | Лофi | — | — | — | 522,23 |
| Остаточная стоимость высвобожденных Об. Ф | Лобфi | — | —• | — | 156,6 |
| Остаточная стоимость высвобожденных фондов с учетом фактора времени | ΣЛiаi | — | — | — | 509,12 |
| Себестоимость реализованной продукции | Спр |  | 351,55 | 599,8 | 819,15 |
| Амортизационные отчисления на реновацию | Аоі |  | -43,75 | -63,66 | -82,3 |
| Налог на добавленную стоимость, перечис. в бюджет | Нбдсі |  | 87,32 | 107,38 | 122,6 |
| Налог на прибыль | Нбп |  | 47,4 | 63,6 | 78,5 |
| Всего платежей и затрат | ΣUi |  | 442,52 | 707,12 | 937,95 |
| Всего платежей и затрат с учетом фактора времени | с |  | 486,3 | 851,9 | 981,6 |

Тогда стоимостная оценка затрат будет равна:

З= ΣКiai- ΣЛiаi + ΣUiai=720,3-609,12+2188,8=2299,98 тыс.грн.

# Определение экономического эффекта

Экономический эффект за расчетный период определяется по формуле:

# Э=Рт-Зт

Где Рт — стоимостная оценка результата.

# Э = 2633,71-2299,98= 334 тыс. грн.

Прибыль, остающаяся в распоряжении предприятия в расчете на одно изделие составляет:

Пои=Э/ ΣАi

Пои= 334000/3000=111,3 грн.

Срок возврата единовременных затрат составляет:

# Tв=( ΣКi\*αi - ΣЛi\*αi)/Э

Тв = (920,3-509,12)/(3419,14-2496)=0,7 года

7. Охрана труда

7.1 Характеристика проектируемого прибора

Данный прибор выполняет функции приёма и передачи аналогового сигнала по радиоканалу при этом осуществляя функции модуляции и демодуляции используя цифровой сигнальный процессор фирмы Analog Devices ADSP 2115. При приёме демодулированный сигнал передаётся через последовательный порт в IBM - совместимый компьютер, где обрабатывается и данные выводятся на экран компьютера.

При передаче цифровой код принимается из компьютера и после модуляции направляется в радиопередатчик.

Данное устройство обработки сигналов работает совместно с компьютером и с блоком приёма и передачи сигнала по радиоканалу (БППСР). Подробная работа БППСР в данном дипломном проекте не рассматривается по различным причинам (большое кол-во входящих подблоков, огромное кол-во функций и т.п. ), мы просто принимаем его как необходимый узел для практического использования нашего устройства. В свою очередь наше устройство очень тесно связано с компьютером, даже более того, работа оператора с нашим устройством заключается в его работе с компьютером. Исходя из этого необходимо принять соответствующие меры для организации рабочего места оператора с точки зрения охраны труда.

Меры для организации рабочего места и безопасности проведения настроечных работ рассмотрим в пункте 7.3.

Разрабатываемому изделию следует определить набор требований по технике безопасности на этапе эксплуатации. В компьютерах наи-большую опасность представляет поражение электрическим током. Источники питания компьютеров (кроме блокнотных) и переферийных устройств рассчитаны на сеть переменного тока 220В 50Гц. Типовой блок питания включается в сеть трехпроводным шнуром питания с вилкой, bимеющей заземляющий контакт. Заземляющий провод соединен с корпусом и его "схемной землёй". Питающие провода проходят через высокочастотный фильтр, подавляющий импульсные помехи, емкость конденсаторов фильтра пропорциональна мощности источника питания.

Для обеспечения сохранности техники (интерфейсных цепей) заземляющие контакты всех соединяемых между собой устройств (компьютер, монитор, принтер) должны надёжно объединяться через соответствующие розетки. Питание монитора через транзитный разъём блока питания системного блока обеспечивает это соединение.

Для обеспечения электробезопасности соединенные между собой заземляющие контакты розеток должны соединяться с общим проводом заземления или с нулевым проводом питания сети (исключая возможность его переключения на фазный провод). Игнорирование этого правила может приводить к появлению на корпусе устройств напряжения прикосновения 110В переменного тока. Уровень ограничения тока при прикосновении определяется ёмкостью конденсаторов фильтра и зависит от мощности устройства. Ток через человека, имеющего контакт с землёй (например, через пол) и прикоснувшегося к устройству, может достигать десятков миллиампер,что является опасным для жизни.

Блок питания компьютера обычно имеет стандартный конструктив и набор проводов с разъёмами питания системной платы и перефирийных устройств. Устройство конструктивно выполнено в виде печатной платы, размещаемой в отдельном корпусе. Блок питается от источника питания напряжением 5 и 12 вольт постоянного тока. Сигнальные напряжения не превышают 5 вольт, что практически безопасно. Устройство выполнено на интегральных микросхемах и дискретных элементах. Суммарная рассеиваемая мощность на плате не превышает 5 Вт. Создаваемый тепловой поток меньше теплового потока создаваемого остальными узлами компьютера, дополнительная теплоизоляция не требуется.

Блок не создает сколь-нибудь значительных электромагнитных, СВЧ, ВЧ полей. Поскольку разрабатываемое устройство устанавливается на штатное место, особенности эксплуатации отсутствуют. Наружный кожух устройства изготавливается из изоляционного материала исключающего появление на нём потенциала опасного для человека ( пластмасса, дерево и т.п.). Ниже приведена конструкция кожуха и вентиляционных отверстий.

Рис. 7.1 Конструкция кожуха.

7.2 Анализ тестирования устройства

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Этап настройки | Приборы, участвующие в настройке | Опасности и вредные факторы | Мероприятия, устранящие выявленные опасности и вредные факторы |
| 1. | Программирова-ние цифрового сигнального процессора | ЭВМ(Intel Pentium) | Влияние мягкого рентгеновского излучения от ЭЛТ на зрение  | Применение защитного экрана, монитора типа Low Radiation, правильное расположение монитора на рабочем месте |
| 2. | Отладка программы | ЭВМ(Intel Pentium) | Влияние мягкого рентгеновского излучения от ЭЛТ на зрение  | Применение защитного экрана, монитора типа Low Radiation, правильное расположение монитора на рабочем месте |
| 3. | Тестирование входных и выходных сигналов | Осцилограф (C1-99),генератор (Г1-34) | Опасность поражения электрическим током | Заземление корпуса осцилографа |

Рис. 7.3 Блок - схема тестирования.

Как видно из блок-схемы на стадии тестирования прибора в которой существует опасность поражения эл. током, необходимо выполнять заземление используемых приборов. Для предотвращения влияния на зрение мягкого рентг. излучения необходимо выбирать рекомендуемый монитор.

7.3 Организация рабочего места и безопасность проведения тестирования

В данном дипломном проекте создается устройство которое напрямую связано с компьютером и управляется с помощью его. Для обслуживания данного устройства требуется опытный оператор, который должен находится за компьютером и следить за принимаемой информацией и при необходимости передавать другую информацию. В этом случае оператор постоянно находится за компьютером. Поэтому необходимо разработать меры по правильной организации рабочего места и безопасности проведения настроечных работ. Мероприятия для безопасности проведения тестирования Тестирование производится оператором на его рабочем месте. Под рабочим местом понимается зона, оснащенная необходимыми техническими средствами, в которой совершается трудовая деятельность исполнителя или группы исполнителей, совместно выполняющих одну работу или операцию. Главным требованием при выборе основного оборудования является обеспечение на данном рабочем месте необходимых безопасных комфортных условий труда и производительности труда, поэтому выбираемое оборудование должно отвечать требованиям эргономики, т.е. требованиям техники безопасности, психологическим и физиологическим возможностям работающего, с учетом его антропомет- рических показателей. Под планировкой рабочего места понимают взаимное расположение основного и вспомогательного оборудования, инвентаря и оснастки на отведенной производственной площади, обеспечивающее наиболее эффективное выполнение трудовых процессов, экономию усилий, минимальное утомление работающего и безопасность его труда. Рабочее место при наладке представляет собой стол, стул. На столе расположено следующее оборудование: - компьютер персональный типа INTEL PENTIUM ; - монитор цветной типа LG 14 ' LR; - осциллограф типа C1-99; - генератор сигналов типа Г1-34; - принтер Canan BJ-200; Оператор напрямую соприкасается с клавиатурой, мышью, осциллографом и монитором. Тестирование заключается в поиске ошибок в программном обеспечения, проверке электрических соединений. В помещениях с ПК обеспечивается равномерное распределение яркости (перепад яркости не превышает 1:3), исключаются яркие источники света и блестящие отражающие предметы в поле зрения оператора. Источником электромагнитных полей и излучений могут служить монитор и осциллограф. Выбранные монитор (ДП 14' LR) и осциллограф (С1-99) сконструированы таким образом, что все виды излучений находятся в диапазоне допустимых значений на любых расстояниях от них (нормируется по ГОСТ 12.1.006 - 84 и по ГОСТ 12.1.002 - 84). Источником шума и вибрации может служить принтер. Выбранный принтер Canon BJ-200 использует химическую технологию печати, уровень шума, создаваемого при работе, удовлетворяет самым строгим требованиям. Для освещения применяют газоразрядные лампы. Допустимая пульсация освещенности для зрительных работ составляет 10%. Поскольку весь комплекс оборудования при наладке устройства питается от сети переменного тока напряжением 220V, 50Hz то для защиты от воздействия электрического тока корпуса монитора, принтера, клавиатуры выполняются из диэлектрического материала. Корпуса, сделанные из металлического материала заземляются, все провода должны быть надежно заизолированы. Дополнительные меры безопасности:

* Конструкция всех элементов рабочего места (РМ) должна исключать возможность прикосновения человека к частям и элементам под напряжением свыше 36 В при любых, в том числе ошибочных, действиях пользователя, не связанных со вскрытием корпуса.
* Система электропитания РМ должна обеспечивать гальваническую развязку от потенциала "земли" с сопротивлением не менее 1 МОм.
* Система электропитания РМ должна быть оборудована устройством защитного отключения, обеспечивающим отключение питающих напряжений от рабочих мест при возникновении утечки на "землю" свыше 10 мА, при перегрузках и коротких замыканиях.
* Конструкция соединителей и разъемов должна исключать возможность ошибочного подсоединения к линиям с неверным напряжением. Кабели электропитания должны иметь достаточную механическую прочность (обеспечиваемую, например, защитной гибкой пластмассовой трубкой или гибким проволочным экраном.).

Мероприятия по эргономике и организации рабочего места:

Эргономика и эстетика производства являются составными частями культуры производства, т.е. комплекса мер по организации труда, направленных на создание благоприятной рабочей обстановки.

В основе повышения культуры производства лежат требования научной организации труда. Культура производства достигается правильной организацией трудовых процессов и отношений между работающими, благоустройством рабочих мест, эстетическим преобразованием среды.

Во время работы часто возникают ситуации, в которых оператор ЭВМ должен за короткий срок принять правильное решение. Для успешного труда в таких условиях необходимы рационально организованная окружающая среда, ограничивающая работника от воздействия посторонних раздражителей, которыми могут быть: мрачная окраска стен, неудобное расположение сигнализации, клавиш управления. Поэтому всеми средствами нужно снижать утомление и напряжение оператора ЭВМ, создавая обстановку производственного комфорта.

Размерные характеристики рабочего места

Конструкцией рабочего места должно быть обеспечено выполнение трудовых операций в пределах зоны досягаемости моторного поля. Зоны досягаемости моторного поля в вертикальной и горизонтальной плоскостях для средних размеров тела человека приведены в ГОСТ 12.2.032-78 на стр.2.

Выполнение трудовых операций "часто" и "очень часто" должно быть обеспечено в пределах зоны легкой досягаемости. Поскольку наладка блока может производиться как мужчинами, так и женщинами, следует учесть средние антропометрические показатели и мужчин и женщин. Так как высота рабочих поверхностей является регулируемой, то значения будем выбирать для роста 1800 мм. Оно составит 800 мм согласно номограмме 4 приведенной в ГОСТ 12.2.032-78 на стр.4. Конструкция регулируемого кресла должна соответствовать ГОСТ 21889- 76. Форма рабочей поверхности прямоугольная. Расположение оборудования на рабочей поверхности:

Рис.7.3.2.1 - Расположение приборов на рабочей поверхности стола и распределение зон их досягаемости.

Зона 1 - зона расположения наиболее важных органов управ-ления.

Зона 2 - зона расположения легкодоступного оборудования.

Зона 3 - зона размещения редкоиспользуемых органов управления.

В зоне 1 расположены монитор и клавиатура. В зоне 2 расположен осциллограф, мультиметр и налаживаемое устройство. В зоне 3 расположен принтер. Монитор, как самое важное средство отображения информации, должен быть установлен таким образом, чтобы центр ЭЛТ находился на линии взгляда или немного ниже.

7.4 Выводы по охране труда

В ходе разработки раздела "Охрана труда" было создано три параграфа характеризующие работу устройства , кодирования - декодирования информации на цифровом сигнальном процессоре, с точки зрения безопасности жизнедеятельности. В качестве детального изучения и устранения одной из опасностей была выбрана: Организация рабочего места оператора и безопасность проведения тестирования. Т.к. оператор нашего устройства должен работать за ЭВМ, все расматривалось с точки зрения обустройства и безопасности работы оператора ЭВМ. Оценка тяжести и напряженности труда по эргономическим показателям целью отнесения их к определенной категории осуществляется по наибольшему количественному критерию. В результате анализа эргономических показателей психофизиоло-гических факторов можно сделать вывод, что работа по наладке устройства относится к категории "легкая, малонапряженная".

Приложение 1

Главный модуль

main.c

// Подключаем заголовочные файлы и объявляем локальные и глобальные // переменные

#include <drivers.h>

#include <nbdp.h>

#include <string.h>

#include <ctype.h>

int PTT\_Recalc;

int DebugLevel;

/\*-------------------------------------------------------------\*/

int cntr;

int CALL\_ID[10];

// Объявление внешних функций и переменных

extern void FreeSignal(void);

extern int IsCharReady(void);

extern int FRE\_FLAG;

extern int setISS,setIRS;

/\*-------------------------------------------------------------\*/

void main()

{

int save\_dsw;

/\*-------------------------------------------------------------\*/

// Сброс выводов FL2..FL0

asm("reset fl0;");

asm("reset fl1;");

asm("reset fl2;");

init\_1847(); // Инициализация кодека AD1847

UART\_Init(); // Инициализация интерфейса UART

// Дадим команду host-устройству на инициалицацию

SendCommandHost("INIT","POWER ON");

// Инициализируем структуру NBDP (narrowband direct printing)

NBDP\_Init();

// Процесс обмена данными

while(1)

{

if(COMF & 1)

{

if(COMF & 0x02) arq(); // запустить протокол ARQ , описание в

// npdp\_arq.c

else if(COMF & 0x1C) fec(); // запустить протокол FEC , описание в

// npdp\_fec.c

else // Если другое сотояние COMF, то ошибку в host

{

SendHostError(5,COMF); /\* COMF ERROR \*/

StandBy();

}

}

if(FRE\_FLAG) // Если модем свободен

{

FRE\_FLAG=0;

// Отправим команду в host , что модем свободен

SendCommandHost("FRE",FRE\_FLAG == 1 ? "1":"0");

}

/\* Если символ готов к передаче , выбрать команды из порта\*/

if(IsCharReady()) SerialDriver();

// Передача символов в HOST

if((COMF & 1) && (ho\_count))

{

char block[15],d;

int i=0;

while(hoRead(&d) != -1)

{

block[i++] = d;

if(i>14) break;

}

block[i]=0;

SendCommandHost("TXT",block); // отправка блока в host

}

} // end of while

} // end of main()

/\*-------------------------------------------------------------\*/

int CheckCommRequest(char \*DATA) // Проверка корректности номера

// объекта связи (парохода), 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 могут быть кодом объекта

{

SendHostError(11,tmp); /\* ERR;01.1;ID WRONG DIGIT (%c),ID[j] \*/

return 1;

}

CALL\_ID[j] = tmp;

j++;

if(j>9) break;

}

CALL\_ID[j]=0;

if(j!=4 && j!=5 && j!=9) /\* incorrect ID \*/ // символ ';' в коде должен быть // третьим, четвертым или восьмым

{

SendHostError(12,j);

return 1;

}

return 0; // проверка успешна, возврат 0.

}

/\*-------------------------------------------------------------\*/

// 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);

/\*-------------------------------------------------------------\*/

Программа модуляции

За основу данного способа модуляции взята - частотная модуляция с использованием протокола морского телеграфа NBDP ( narrowband direct printing) - узкополосное буквопечатанье. В основе лежит таблица кодовых значений сиволов, которые представляются в виде комбинации 1 и 0 и после модулируются с соответственно частотами 1615 Hz и 1785 Hz. ( таблица символов в файле nbdp\_table.c ).

Mod.c

// Подключаем заголовочные файлы и объявляем локальные и глобальные // переменные

#include <drivers.h>

#include <template.h>

#include <nbdp.h>

/\*-------------------------------------------------------------\*/

/\*#define OUT\_KOEFF 0x6000\*/

#ifdef RX\_TO\_TX\_

extern int RX\_TO\_TX[2];

#endif

/\*-------------------------------------------------------------\*/

int PH\_ACC,TMP\_PH\_ACC;

int PH\_INC;

int mod\_ready=1;

int BITTIME=1;

int bits\_left;

int out\_data;

volatile int l\_out,r\_out;

extern int FDIV,FS\_PTT\_OFF;

int dbg\_cntr;

/\*-------------------------------------------------------------\*/

/\* MODULATOR \*/

/\*-------------------------------------------------------------\*/

extern void Timing(void);

void modulator(void)

{

// Возможна работа в тестовом режиме.

if(SelfTest) goto test\_modes;

FDIV++;

// через 10 мс обнуление отсчетов и посылка в хост синхронизирующего // сигнала цикла ARQ.

if(FDIV==80) {FDIV=0;NBDP\_THR\_TX();}

#ifdef RX\_TO\_TX\_

tx\_buf[1] = RX\_TO\_TX[0];

tx\_buf[2] = RX\_TO\_TX[1];

return;

#endif

// Проверка нужна ли модуляция, если нет то возврат

if(!Modulator\_ON)

{

PH\_ACC = 0;

return;

}

restart:

if(mod\_ready)

{

tx\_buf[1] = tx\_buf[2] = 0;

if(FS\_PTT\_OFF) // выключение модуляции

{

FS\_PTT\_OFF=0;

PTT\_OFF();

}

return;

}

test\_modes:

BITTIME--;

if(BITTIME==0)

{

/\*=== determine the new bit from out byte ===\*/

// Режим реальной работы

if(!SelfTest) out\_data <<= 1; /\* 7 bit mode \*/

if(out\_data & 0x80) /\* MARK \*/

{

PH\_INC = MARK\_INC;

asm("

#define PFDATA 0x3fe5

ar = b#0000000001000000; /\* set 1 PF6/TLG OUT \*/

ay1 = dm(PFDATA);

ar = ar or ay1;

dm(PFDATA) = ar;");

}

else /\* SPACE \*/

{

PH\_INC = SPACE\_INC;

asm("

ar = b#1111111110111111; /\* reset 1 PF6/TLG OUT \*/

ay1 = dm(PFDATA);

ar = ar and ay1;

dm(PFDATA) = ar;");

}

// Тестовый режим, работа в 8-битном режиме

if(SelfTest) out\_data <<= 1;

BITTIME = BITLENGTH;

bits\_left--;

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;");\*/

PH\_ACC += PH\_INC;

l\_out = sin\_i(PH\_ACC);

tx\_buf[2] = l\_out; // выдача результатов в порт ( связь с кодеком)

}

/\*-------------------------------------------------------------\*/

void modulating(void)

{

if(SelfTest)

{

if(SelfTest==1) /\* space \*/

{

out\_data = 0;

}

else if(SelfTest==2) /\* mark \*/

{

out\_data = 0xFF;

}

else if(SelfTest==3) /\* dot \*/

{

/\* 10101010 \*/

out\_data = 0xAA;

}

else if(SelfTest==4) /\* big dot \*/

{

/\* 11110000 \*/

out\_data = 0xF0;

}

else

{

SelfTest=0;

}

BITTIME=1;

bits\_left=8; // 8-битный режим работы

mod\_ready=0;

return;

}

// В случае тестового режима дальше не идем

if(mod\_ready == 0) return;

BITTIME=1;

if(ModulatorGet(&out\_data)==0) /\* nothing to get \*/

{

// обнуление флагов и возврат

/\* clear PF6 as TLG & PF7 as syncro\_out \*/

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

mod\_ready=0;

}

Программа демодуляции

Принцип демодуляции входного сигнала нашего устройства, основывается на том, что нам известны частоты поступающие от отдельного устройства приема сигнала по радиоканалу, уже отфильтрованные и поданные на вход нашего устройства. И нам необходимо выделить соответственно частоты 1785 Hz = MARK = ' 0 ' и 1615 Hz = SPACE = ' 1 ' и получить определенный код. После получении кода происходит раскодирование по таблице NBDP и передача в ЭВМ, где программа TERMINAL соответственно 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 out\_mcr(int data);

#define DMD\_KOEFF 0x019A

#define DMD\_LEVEL 0x1000 /\* порог срабатывания \*/

#define OUT\_MARK 'M'

#define OUT\_SPACE 'S'

#define OUT\_ERR 'E'

int my\_fir(int NewValue);

int my\_sqrt(int NewValue);

/\*-------------------------------------------------------------\*/

volatile int l\_in,r\_in;

int Demodulator\_ON=1; // Demodulator ON/OFF flag

int PH\_TONE\_ACC[2]; // опорный MARK/SPACE PHASE ACC

int PH\_TONE\_INC[2] = {MARK\_INC, SPACE\_INC};

int i,j; // временные счетчики

int R [4]; // временные результаты

int S [4]; // частичные суммы

int DL[4\*BITLENGTH]; // Delay lines for 4 bands

int DLp = 0; // DL pointer

int countN=BITLENGTH; // Cycle count

int REZ[2]; // Результаты

int PRZLT[2]; // Приблизительные результаты (prev. Rez)

int svMode; /\* Save multiplier mode location \*/

int JitterOff; /\* Bit syncro OFF flag \*/

int OutData[2];

void demodulator(void)

{

// Если демодулятор откл., то обнуление результатов и возврат

if(!Demodulator\_ON && (dip\_sw & DIP\_SW1))

{

REZ[0] = 0;

out\_mcr(0);

goto CheckCycle;

}

if(SelfTest) // Если тестовый режим, то возврат

{

return;

}

// Выключение режима целочисленной арифметики

asm("dis m\_mode;");

l\_in = rx\_buf[IN\_CHNL];

// ограничить входной сигнал для устранения переполнения фильтра

r\_in = \_FRACT\_MULTIPLY\_(r\_in,DMD\_KOEFF);

Заполняем массив временных результатов для MARK и SPACE

PH\_TONE\_ACC[0] += PH\_TONE\_INC[0];

R[0] = \_FRACT\_MULTIPLY\_(r\_in,sin\_i(PH\_TONE\_ACC[0]));

R[1] = \_FRACT\_MULTIPLY\_(r\_in,cos\_i(PH\_TONE\_ACC[0]));

PH\_TONE\_ACC[1] += PH\_TONE\_INC[1];

R[2] = \_FRACT\_MULTIPLY\_(r\_in,sin\_i(PH\_TONE\_ACC[1]));

R[3] = \_FRACT\_MULTIPLY\_(r\_in,cos\_i(PH\_TONE\_ACC[1]));

//Извлекаем старые данные и добавляем новые

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 результата по каждому фильтру.

PRZLT[0] = REZ[0]; // Предварительный результат

REZ[0] = \_FRACT\_MULTIPLY\_(S[0],S[0]) + \_FRACT\_MULTIPLY\_(S[1],S[1]);

REZ[1] = \_FRACT\_MULTIPLY\_(S[2],S[2]) + \_FRACT\_MULTIPLY\_(S[3],S[3]);

if(dip\_sw & DIP\_SW2)

{

REZ[0] = my\_sqrt(REZ[0]);

REZ[1] = my\_sqrt(REZ[1]);

}

R[1] = (REZ[0]-REZ[1]);

R[2] = (REZ[0]+REZ[1]);

R[0] = \_FRACT\_MULTIPLY\_((\_FRACT\_DIVIDE\_(R[1],R[2])),0x6400);

REZ[0] = R[0];

/\* debug solution output \*/

tx\_buf[1] = REZ[0];

/\*================================================

STEP 4.

Time supervision, bit detection, etc.

================================================\*/

CheckCycle:

countN--;

if(!countN)

{

if(abs(REZ[0]) > DMD\_LEVEL)

{

if(REZ[0] > 0)

{

OutData[0] = OUT\_MARK;

asm("reset fl0; set fl2; set fl1; ");

}

else

{

OutData[0] = OUT\_SPACE;

asm(" set fl0; reset fl2; set fl1; ");

}

}

else

{

OutData[0] = OUT\_ERR;

asm(" set fl0; set fl2; reset fl1; ");

}

countN = BITLENGTH;

// отправляем на ЦАП (кодек)

NBDP\_THR\_RX(OutData[0]);

}

// включение режима целочисленной арифметики

asm("ena m\_mode;");

}

Дополнительные программы и функции

1. Программа nbdp\_table.c

Содержит таблицу соответствия кодов ASCII и кодов формата NBDP.

#include <nbdp.h>

/\*B - MARK FREQ:1785Hz

/\*Y - SPACE FREQ:1615Hz

/\* ¦ +-----------------¦ 5-Unit ¦ 7-Unit ¦ \*/

/\* ¦No ¦LET ¦FIG ¦RUS ¦ Code ¦ Code ¦ \*/

UCHAR \_7bit\_code[]= { /\*+--+-----+-----+-----+---------+-----------¦ \*/

0x71 ,/\*¦01¦A 41 ¦- 2D ¦А 80 ¦ZZAAA ¦BBBYYYB 71 ¦ \*/

0x27 ,/\*¦02¦B 42 ¦? 3F ¦Б 81 ¦ZAAZZ ¦YBYYBBB 27 ¦ \*/

0x5c ,/\*¦03¦C 43 ¦: 3A ¦Ц 96 ¦AZZZA ¦BYBBBYY 5C ¦ \*/

0x65 ,/\*¦04¦D 44 ¦Wh?? ¦Д 84 ¦ZAAZA ¦BBYYBYB 65 ¦ \*/

0x35 ,/\*¦05¦E 45 ¦3 33 ¦Е 85 ¦ZAAAA ¦YBBYBYB 35 ¦ \*/

0x6c ,/\*¦06¦F 46 ¦Э\*9D ¦Ф 94 ¦ZAZZA ¦BBYBBYY 6C ¦ \*/

0x56 ,/\*¦07¦G 47 ¦Ш\*98 ¦Г 83 ¦AZAZZ ¦BYBYBBY 56 ¦ \*/

0x4b ,/\*¦08¦H 48 ¦Щ\*99 ¦Х 95 ¦AAZAZ ¦BYYBYBB 4B ¦ \*/

0x59 ,/\*¦09¦I 49 ¦8 38 ¦И 88 ¦AZZAA ¦BYBBYYB 59 ¦ \*/

0x74 ,/\*¦10¦J 4A ¦Ю\*07 ¦Й 89 ¦ZZAZA ¦BBBYBYY 74 ¦ \*/

0x3c ,/\*¦11¦K 4B ¦( 28 ¦К 8A ¦ZZZZA ¦YBBBBYY 3C ¦ \*/

0x53 ,/\*¦12¦L 4C ¦) 29 ¦Л 8B ¦AZAAZ ¦BYBYYBB 53 ¦ \*/

0x4e ,/\*¦13¦M 4D ¦. 2E ¦М 8C ¦AAZZZ ¦BYYBBBY 4E ¦ \*/

0x4d ,/\*¦14¦N 4E ¦, 2C ¦Н 8D ¦AAZZA ¦BYYBBYB 4D ¦ \*/

0x47 ,/\*¦15¦O 4F ¦9 39 ¦О 8E ¦AAAZZ ¦BYYYBBB 47 ¦ \*/

0x5a ,/\*¦16¦P 50 ¦0 30 ¦П 8F ¦AZZAZ ¦BYBBYBY 5A ¦ \*/

0x3a ,/\*¦17¦Q 51 ¦1 31 ¦Я 9F ¦ZZZAZ ¦YBBBYBY 3A ¦ \*/

0x55 ,/\*¦18¦R 52 ¦4Ч34 ¦Р 90 ¦AZAZA ¦BYBYBYB 55 ¦ \*/

0x69 ,/\*¦19¦S 53 ¦' 27 ¦С 91 ¦ZAZAA ¦BBYBYYB 69 ¦ \*/

0x17 ,/\*¦20¦T 54 ¦5 35 ¦Т 92 ¦AAAAZ ¦YYBYBBB 17 ¦ \*/

0x39 ,/\*¦21¦U 55 ¦7 37 ¦У 93 ¦ZZZAA ¦YBBBYYB 39 ¦ \*/

0x1e ,/\*¦22¦V 56 ¦= 3D ¦Ж 86 ¦AZZZZ ¦YYBBBBY 1E ¦ \*/

0x72 ,/\*¦23¦W 57 ¦2 32 ¦В 82 ¦ZZAAZ ¦BBBYYBY 72 ¦ \*/

0x2e ,/\*¦24¦X 58 ¦/ 2F ¦Ь 9C ¦ZAZZZ ¦YBYBBBY 2E ¦ \*/

0x6a ,/\*¦25¦Y 59 ¦6 36 ¦Ы 9B ¦ZAZAZ ¦BBYBYBY 6A ¦ \*/

0x63 ,/\*¦26¦Z 5A ¦+ 2B ¦З 87 ¦ZAAAZ ¦BBYYYBB 63 ¦ \*/

0x0f ,/\*¦27¦CR 0D -- ----- ¦AAAZA ¦YYYBBBB 0F ¦ \*/

0x1b ,/\*¦28¦LF 0A -- ----- ¦AZAAA ¦YYBBYBB 1B ¦ \*/

0x2d ,/\*¦29¦LET 16 -- ----- ¦ZZZZZ ¦YBYBBYB 2D ¦ \*/

0x36 ,/\*¦30¦FIG 0F -- ----- ¦ZZAZZ ¦YBBYBBY 36 ¦ \*/

0x1d ,/\*¦31¦SP 20 -- ----- ¦AAZAA ¦YYBBBYB 1D ¦ \*/

0x2b ,/\*¦32¦RUS 00 -- ----- ¦AAAAA ¦YBYBYBB 2B ¦ \*/

/\*-------------------------------------------------------------- \*/

#ifdef RUSSIAN // структура данных под русский алфавит

struct CVT CVT\_TABLE[] =

{

/\*¦ +-----------------¦ 5-Unit ¦ 7-Unit ¦ \*/

/\*¦ No ¦LET ¦ FIG ¦RUS ¦ Code ¦ Code ¦ \*/

/\*+----+-------+-------+---------+--------------------¦ \*/

{0x41,0x2D,'a' },/\*¦01¦A 41 ¦- 2D ¦А 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 ¦Е 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 ¦Х 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 ¦М 8C ¦AAZZZ ¦BYYBBBY 4E ¦ \*/

{0x4E,0x2C,'n' },/\*¦14¦N 4E ¦, 2C ¦Н 8D ¦AAZZA ¦BYYBBYB 4D ¦ \*/

{0x4F,0x39,'o' },/\*¦15¦O 4F ¦9 39 ¦О 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 ¦Р 90 ¦AZAZA ¦BYBYBYB 55 ¦ \*/

{0x53,0x27,'s' },/\*¦19¦S 53 ¦' 27 ¦С 91 ¦ZAZAA ¦BBYBYYB 69 ¦ \*/

{0x54,0x35,'t' },/\*¦20¦T 54 ¦5 35 ¦Т 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 ¦В 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 ¦ \*/

{0x03,0x03,0x03},/\*¦32¦RUS 00 -- ----- ¦AAAAA ¦YBYBYBB 2B ¦ \*/

2. Программа nbdp.c

Содержит функции для работы с протоколом NBDP, а также объявление всех глобальных переменных используемых 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; /\* COMMUNICATION FLAG \*/

/\* b0 : 0=stanby 1=communication \*/

/\* b1 : 1=ARQ Mode \*/

/\* b2 : 1=CFEC Mode \*/

/\* b3 : 1=SFEC Mode \*/

/\* b4 : 1=GFEC Mode \*/

/\* b5 : 0=send 1=recv \*/

/\* b6 : 1 = SLAVE wait PHASING, 0 = SLAVE PHASED \*/

/\* b7 : \*/

UCHAR STATE; /\* COMMUNICATION 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 counter for FEC \*/

UINT RxErr,RxOK; /\* Received Err and OK symbol \*/

UCHAR SHIFT; /\* LET=0,FIG=1,RUS=2 \*/

/\*UCHAR PTT;\*/ /\* -BK intenal bit \*/

UCHAR WRU\_SUPPORT=1;

UCHAR DX[5]; /\* INPUT DX LINE FOR FEC \*/

UCHAR RX[5]; /\* INPUT RX LINE FOR FEC \*/

UCHAR ID\_IN[7]; /\* INPUT SFEC ID SHIFT REG \*/

UCHAR SFSIS4[4]; /\* SFEC self identification signal for 4-digit ID \*/

UCHAR SFSIS5[4]; /\* SFEC self identification signal for 5-digit ID \*/

UCHAR SFSIS9[7]; /\* SFEC self identification signal for 9-digit ID \*/

UCHAR SCID[7]; /\* SFEC called station ID \*/

UINT n[10]; /\* ARQ(and other) supervisor counters \*/

UCHAR connector;

UCHAR CB[3][3]; /\* CALLING BLOCK DURING SLAVE PHASING \*/

/\* AND FOR CALL DURING ARQ MASTER \*/

UCHAR CIB[3][3]; /\* CALLER ID BLOCK FOR 7-digit SLAVE SYNC \*/

UCHAR CCS[3]; /\* CS DURING ARQ MASTER CALL \*/

/\* AND DURING SLAVE PHASING \*/

UCHAR CKERR; /\* CK errors during phasing \*/

/\*UCHAR RCB[3][3];\*/ /\* REPHASING: CALLING BLOCK DURING SLAVE REPHASING \*/

UCHAR RDATA[3]; /\* RECEIVE DATA BLOCK \*/

UCHAR SLICE\_BUF[2];

UCHAR LAST\_CS;

UCHAR LAST\_BLOCK[3];/\* last sended block \*/

UCHAR LAST\_BL\_NO;

UCHAR LAST\_ID[10]; /\* last comm try ID \*/

UCHAR RQ; /\* REPEAT REQUEST \*/

UCHAR RPTF; /\* REPEAT FLAG \*/

UCHAR ID\_TYPE; /\* 4- or 7-DIGIT ID TYPE. 0-4,1-7 \*/

UCHAR WRU\_CNT; /\* MULTIPLE WRU REQUST COUNT \*/

UCHAR strAAB[21]; /\* not converted AAB \*/

UCHAR AAB[AAB\_LEN];/\* converted AAB \*/

UCHAR SIS4[6]; /\* self id signal for 4-digit ID \*/

UCHAR SIS5[6]; /\* self id signal for 5-digit ID \*/

UCHAR SIS9[9]; /\* self id signal for 9-digit ID \*/

UCHAR SCS9[3]; /\* self id check sum for 9-digit ID \*/

char o\_shift=4; /\* output shift \*/

char RQ3SIG [3] = "\x33\x33\x33"; /\* rq-rq-rq \*/

char B\_A\_B [3] = "\x66\x78\x66"; /\* b-a-b \*/

char EOC\_SIG[3] = "\x78\x78\x78"; /\* a-a-a \*/

/\*UCHAR PRE\_KEY; \*/ /\* PRE KEY IN MS (0..100) \*/

/\*UCHAR POST\_KEY;\*/ /\* POST KEY IN MS (0..100) \*/

UCHAR CFEC\_RX=1; /\* CFEC RECEIVING ON/OFF, default ON \*/

UCHAR FEC\_RATE=50; /\* FEC ERROR RATE, default 50% \*/

UCHAR IN\_CHNL =2; /\* INPUT CHANNEL L/R, default RIGHT \*/

UCHAR OUT\_CHNL=1; /\* OUTPUT CHANNEL L/R, default LEFT \*/

UCHAR LAST\_RX\_CS = 0;

UCHAR LAST\_RX\_BLOCK = 0;

UCHAR after\_rephase = 0;

extern int SlaveAnswerTime; /\* Called mode answer delay \*/

/\*-------------------------------------------------------------\*/

void StandBy(void) // В алгоритме вызывается при ошибке

{

MODEM\_STATE=5; /\* state = 5 - RESET \*/

COMF = STATE = SHIFT = RxErr = RxOK = /\*RQ = RPTF = \*/ 0;

LAST\_STATE = 0;

IDLEAcnt = 0;

connector = 0;

/\*memset(DX,0,5);memset(RX,0,5);\*/

FirstIdleBlock = 0;

SendFCCInfo = 0;

arqf = 0;

LAST\_CS = 0;

o\_shift = 4; /\* output shift not defined \*/

WRU\_SUPPORT=1; /\* Set default as ON \*/

hiClear();

hoClear();

Clear();

doClear();

if(dip\_sw & DIP\_SW3)

{

if(SlaveAnswerTime != (21+2))

{

SlaveAnswerTime = 21+2; /\* Called mode answer delay 20 ms \*/

goto inform\_host;

}

}

else

{

if(SlaveAnswerTime != (21+5))

{

SlaveAnswerTime = 21+5; /\* Called mode answer delay 50 ms \*/

inform\_host:

SendCommandHostDBG(1,"CHANGE SLAVE ANSWER TIME [%X0 msec]",SlaveAnswerTime-21);

}

}

send\_stat();

/\*FS[0] = 1;\*/ /\* restart FREE SIGNAL \*/

SendCommandHostDBG(1,"STAND BY");

}

/\*-------------------------------------------------------------\*/

void send\_stat(void) /\* Послать статус устройства в host-компьютер \*/

{

/\*

SendCommandHost("STAT","%02X;%02X;%02X;%03X;%02X;%02X",

COMF,STATE,arqf,hiIFree,hi\_count,connector);

\*/

SendCommandHost("STAT","%02X;%02X;%02X;%03X;%02X;%X;%X",

COMF,STATE,arqf,hiIFree,hi\_count,MODEM\_STATE,connector);

}

/\*-------------------------------------------------------------\*/

/\*-------------------------------------------------------------\*/

/\* Mode \*/

/\* 0 - ARQ \*/

/\* 1 - CFEC \*/

/\* 2 - SFEC \*/

/\*-------------------------------------------------------------\*/

UCHAR convert(char ch) // Конвертер в NBDP

{

if(ch == ERR\_SMBL) return ERR\_SMBL;

{

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 ERR\_SMBL;

}

/\*-------------------------------------------------------------\*/

void NBDP\_Init(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();

}

3. Программа serial.c

Содержит функции для работы с последовательным портом.

#include <stdarg.h>

#include <string.h>

#include <drivers.h>

/\*-------------------------------------------------------------\*/

#define MAX\_CMD\_LEN 127

unsigned int CMDS, /\* command rx state \*/

CMDL; /\* command rx pointer \*/

char CMD[MAX\_CMD\_LEN+1]; /\* command rx buffer \*/

/\*-------------------------------------------------------------\*/

int chr;

int SerialDriver(void) // проверка работы и инициализация последоват. порта

{

/\*int chr;\*/

repeat:

if(get\_char(&chr))

{

if(CMDS) /\* command receiving \*/

{

if(CMDL>=MAX\_CMD\_LEN)

{

SendHostError(0,0);

reset\_state:

CMDL=CMDS=0;

return 0;

}

if((CMD[CMDL-1] == ';') && (chr == '@')) /\* EOC \*/

{

CMD[CMDL++] = chr;

CMD[CMDL] = 0;

HostCommandParser(&CMD[0]);

goto reset\_state;

}

else

{

CMD[CMDL++] = chr;

goto repeat;

return 0;

}

}

else /\* command waiting \*/

{

if(chr=='$')

{

CMDL=0;

CMD[CMDL++] = chr;

CMDS=1;

goto repeat;

}

}

}

return 0;

}

/\*-------------------------------------------------------------\*/

extern unsigned int No;

char BUFFER[MAX\_CMD\_LEN+1];

int SendCommandHost(char \*cmd, char \*fmt,...) // функция для передачи

// команд в терминал

{

/\* Make data string to out \*/

/\*int len;\*/

va\_list argptr;

/\* Out the packet header: $CMD;channel\_no; \*/

out\_char('$');

out\_string(cmd/\*,strlen(cmd)\*/);

out\_char(';');

out\_char(No + 0x30);

out\_char(';');

/\* Out the variable part \*/

va\_start(argptr, fmt);

/\*len =\*/ vprinter(&BUFFER[0], fmt, argptr);

va\_end(argptr);

out\_string(BUFFER/\*,len\*/);

/\* Out the end part of packet \*/

out\_string(";@\r\n"/\*,4\*/);

return 0;

}

/\*-------------------------------------------------------------\*/

int SendCommandHostDBG(int level, char \*fmt,...)

{

/\* Make data string to out \*/

/\*int len;\*/

va\_list argptr;

if(!(dip\_sw & DIP\_SW4)) return 0; /\* not output debug \*/

if(level > DebugLevel) return 0;

/\* Out the packet header: $CMD;channel\_no; \*/

out\_string("$DBG;"); out\_char(No + 0x30); out\_char(';');

/\* Out the variable part \*/

va\_start(argptr, fmt);

vprinter(&BUFFER[0], fmt, argptr);

va\_end(argptr);

out\_string(BUFFER);

/\* Out the end part of packet \*/

out\_string(";@\r\n");

return 0;

}

Критичные по быстродействию функции выполняем на языке ассемблер под ADSP 2181.

Sin.dsp

Функции разложения на синус и косинус используемые при модуляции и демодуляции для вычисления "налету".

.MODULE/RAM \_SIN\_COS\_INTEGER\_;

{---------------------------------------------------------------

Sine/Cosine approximation for 1.15 format

int Y = sin\_i(int X)

int Y = cos\_i(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 sin\_coeff[5];

.INIT sin\_coeff : 0x3240, 0x0053, 0xAACC, 0x08B7, 0x1CCE;

.ENTRY sin\_i\_;

.ENTRY cos\_i\_;

cos\_i\_:

AY1 = 0x4000; { AY0 = PI/2 }

AR = AR + AY1; { AR = X+PI/2 }

sin\_i\_:

SI = AR; { save AR }

I1 = ^sin\_coeff; {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 2181\_hdr.dsp

Содержит код инициализации ADSP 2181.

.MODULE/ABS=0 ADSP2181\_Runtime\_Header;

#define \_MY\_HANDLER 1

// объявление внешних функций Си для использования их в ассемблере

.EXTERNAL \_\_\_lib\_setup\_everything;

.EXTERNAL main\_;

#ifndef \_MY\_HANDLER

.EXTERNAL \_\_\_lib\_sp0r\_ctrl;

#endif

.EXTERNAL stat\_flag\_;

.EXTERNAL next\_cmd\_;

.EXTERNAL process\_a\_bit; { uart }

.EXTERNAL IRQE\_Flag\_;

// Установка векторов прерываний

// По сбросу загружать функцию Си main

\_\_Reset\_vector: CALL \_\_\_lib\_setup\_everything;

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 \_\_\_lib\_sp0r\_ctrl;NOP;NOP;NOP;

#else

\_\_Sport0\_recv: JUMP \_sp0rx;NOP;NOP;NOP;

#endif

\_\_InterruptE: ena sec\_reg; ar = 1; dm(IRQE\_Flag\_)=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;

\_\_Timer\_interrupt: jump process\_a\_bit; /\* 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(REG\_SAVE\_SP0RX + 0) = PX; /\* 1 \*/

dm(REG\_SAVE\_SP0RX + 1) = L0; /\* 2 \*/

dm(REG\_SAVE\_SP0RX + 2) = I1; /\* 3 \*/

dm(REG\_SAVE\_SP0RX + 3) = L1; /\* 4 \*/

dm(REG\_SAVE\_SP0RX + 4) = M2; /\* 5 \*/

dm(REG\_SAVE\_SP0RX + 5) = M3; /\* 6 \*/

dm(REG\_SAVE\_SP0RX + 6) = M5; /\* 7 \*/

dm(REG\_SAVE\_SP0RX + 7) = L5; /\* 8 \*/

dm(REG\_SAVE\_SP0RX + 8) = I6; /\* 9 \*/

dm(REG\_SAVE\_SP0RX + 9) = M6; /\* 10 \*/

dm(REG\_SAVE\_SP0RX + 10) = L6; /\* 11 \*/

/\* enable second register bank \*/

ena sec\_reg;

/\* set predefined INTR modes \*/

DIS BIT\_REV, DIS AR\_SAT, DIS AV\_LATCH, ENA M\_MODE;

L0=0;

L1=0;

L5=0;

L6=0;

M2=0;

M6=0;

/\* restore unaltered registers \*/

PX = dm(REG\_SAVE\_SP0RX + 0)/\* = PX\*/; /\* 1 \*/

L0 = dm(REG\_SAVE\_SP0RX + 1)/\* = L0\*/; /\* 2 \*/

I1 = dm(REG\_SAVE\_SP0RX + 2)/\* = I1\*/; /\* 3 \*/

L1 = dm(REG\_SAVE\_SP0RX + 3)/\* = L1\*/; /\* 4 \*/

M2 = dm(REG\_SAVE\_SP0RX + 4)/\* = M2\*/; /\* 5 \*/

M3 = dm(REG\_SAVE\_SP0RX + 5)/\* = M3\*/; /\* 6 \*/

M5 = dm(REG\_SAVE\_SP0RX + 6)/\* = M5\*/; /\* 7 \*/

L5 = dm(REG\_SAVE\_SP0RX + 7)/\* = L5\*/; /\* 8 \*/

I6 = dm(REG\_SAVE\_SP0RX + 8)/\* = I6\*/; /\* 9 \*/

M6 = dm(REG\_SAVE\_SP0RX + 9)/\* = M6\*/; /\* 10 \*/

L6 = dm(REG\_SAVE\_SP0RX + 10)/\* = L6\*/; /\* 11 \*/

rti;

#endif

.var/dm REG\_SAVE\_SP0TX[11];

\_sp0tx:

/\* enable second register bank \*/

ena sec\_reg;

ar = dm(stat\_flag\_);

ar = pass ar;

if ne jump next\_cmd\_;

/\* save unaltered registers \*/

dm(REG\_SAVE\_SP0TX + 0) = PX; /\* 1 \*/

dm(REG\_SAVE\_SP0TX + 1) = L0; /\* 2 \*/

dm(REG\_SAVE\_SP0TX + 2) = I1; /\* 3 \*/

dm(REG\_SAVE\_SP0TX + 3) = L1; /\* 4 \*/

dm(REG\_SAVE\_SP0TX + 4) = M2; /\* 5 \*/

dm(REG\_SAVE\_SP0TX + 5) = M3; /\* 6 \*/

dm(REG\_SAVE\_SP0TX + 6) = M5; /\* 7 \*/

dm(REG\_SAVE\_SP0TX + 7) = L5; /\* 8 \*/

dm(REG\_SAVE\_SP0TX + 8) = I6; /\* 9 \*/

dm(REG\_SAVE\_SP0TX + 9) = M6; /\* 10 \*/

dm(REG\_SAVE\_SP0TX + 10) = L6; /\* 11 \*/

/\* enable second register bank \*/

/\*ena sec\_reg;\*/

/\* set predefined INTR modes \*/

DIS BIT\_REV, DIS AR\_SAT, DIS AV\_LATCH, ENA M\_MODE;

L0=0;

L1=0;

L5=0;

L6=0;

M2=0;

M6=0;

call modulator\_;

/\* restore unaltered registers \*/

PX = dm(REG\_SAVE\_SP0TX + 0)/\* = PX\*/; /\* 1 \*/

L0 = dm(REG\_SAVE\_SP0TX + 1)/\* = L0\*/; /\* 2 \*/

I1 = dm(REG\_SAVE\_SP0TX + 2)/\* = I1\*/; /\* 3 \*/

L1 = dm(REG\_SAVE\_SP0TX + 3)/\* = L1\*/; /\* 4 \*/

M2 = dm(REG\_SAVE\_SP0TX + 4)/\* = M2\*/; /\* 5 \*/

M3 = dm(REG\_SAVE\_SP0TX + 5)/\* = M3\*/; /\* 6 \*/

M5 = dm(REG\_SAVE\_SP0TX + 6)/\* = M5\*/; /\* 7 \*/

L5 = dm(REG\_SAVE\_SP0TX + 7)/\* = L5\*/; /\* 8 \*/

I6 = dm(REG\_SAVE\_SP0TX + 8)/\* = I6\*/; /\* 9 \*/

M6 = dm(REG\_SAVE\_SP0TX + 9)/\* = M6\*/; /\* 10 \*/

L6 = dm(REG\_SAVE\_SP0TX + 10)/\* = L6\*/; /\* 11 \*/

rti;

.ENDMOD;