

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Политехнический институт
Факультет приборостроения,
информационных технологий и
электроники

Кафедра
«Информационная безопасность
систем и технологий»

Специальность 100503 –

Информационная безопасность
автоматизированных систем

Специализация -

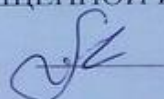
Защищенные автоматизированные
системы управления

ДИПЛОМНЫЙ ПРОЕКТ

на тему

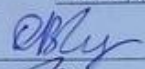
УНИФИЦИРОВАННЫЙ УЗЕЛ ДЛЯ АВТОМАТИЗИРОВАННОЙ
ОБРАБОТКИ ЗАЩИЩЕННОЙ ИНФОРМАЦИИ

Студент



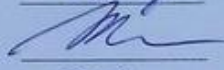
Погодина Анна Евгеньевна

Руководитель



О.В. Кулагин

Нормоконтролёр



А.П. Иванов

Рецензент Зам. начальника кибос



С.А. Васильев

Работа допущена к защите (протокол заседания кафедры от 20.06.2016 № 14)

Заведующий кафедрой



С.И. Зефирова

Работа защищена с отметкой

хорошо

(протокол заседания ГЭК

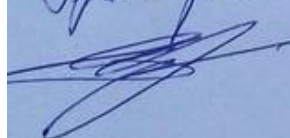
от 23.06.16 № 7)

Секретарь ГЭК



А.П. Иванов

Пенза 2016

Проверка ВКР – Ткачев
 С.И. Зефирова

РЕФЕРАТ

Дипломный проект содержит 61 с., 9 рис., 1 табл., 8 источников.

ПЕРЕНОСИМОСТЬ, ПРОГРАММА, СТРУКТУРИЗАЦИЯ, ЗАЩИЩЕННАЯ АСОД, УТАОИ, АППАРАТНАЯ ЧАСТЬ

Объектом разработки является унифицированный узел для программно-аппаратной обработки информации.

Целью работы является разработка унифицированного узла для программно-аппаратной обработки информации в составе защищенной автоматизированной системы обработки данных (АСОД).

Результатом работы является структурная и функциональная схема узла, построенная на основе предложенной профессором В.А. Герасименко унифицированной технологии автоматизированной обработки информации (УТАОИ).

В процессе разработки была поставлена и успешно решена задача повторного использования аппаратного и программного задела при разработке защищенных АСОД, что означает в первую очередь решение задачи переносимости программного обеспечения.

СОДЕРЖАНИЕ

Введение.....	8
1 Подход к структуризации и переносимости программного обеспечения	11
1.1 Развитие подходов к переносимости программного обеспечения	11
1.2 Метод структуризации программного обеспечения, облегчающий его переносимость	13
2 Защита информации и унифицированная технология автоматизированной обработки информации.....	17
2.1 Необходимость комплексного подхода к защите информации	17
2.2 Постановка задачи комплексной защиты информации в современных автоматизированных системах обработки данных.....	18
2.3 Унифицированная технология автоматизированной обработки информации	20
3 Принципы структуризации аппаратных узлов.....	29
3.1 Принципы и методы структурирования архитектуры технических систем и процессов их функционирования	29
3.2 Виды структуризации	40
3.3 Рекомендации по структуризации аппаратных узлов с программной поддержкой	41
4 Выбор элементной базы	42
4.1 Элементная база отечественного производства.....	42
4.1 Интерфейс Ethernet	42
4.2 Микроконтроллер 1986BE3T.....	44
4.3 Другие микросхемы	49
5 Разработка узла.....	54
5.1 Разработка структурной схемы узла	54
5.2 Разработка функциональной схемы узла.....	56
Заключение	60
Список использованных источников	61

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

УДК 004.056.55

Политехнический институт
Факультет приборостроения,
информационных технологий и
электроники
Выпускающая кафедра:
Информационная безопасность систем
и технологий
учебная группа 11ПИ1

УТВЕРЖДАЮ
Зав. кафедрой ИБСТ
к.т.н., доцент С.Л. Зефиров

27.04 2016 г.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

студентки группы 11ПИ1 Погодиной Анны Евгеньевны

Специальность 100503 – Информационная безопасность
автоматизированных систем

Тема дипломного проекта: Унифицированный узел для
автоматизированной обработки защищенной информации.

Руководитель ВКР
Инженер I категории АО «ПНИЭИ», к.т.н.



Кулагин О.В.

Нормоконтролер

к.т.н., доцент



Иванов А.П.

1 Цели и задачи ВКР

Цель работы – разработка унифицированного узла для программно-аппаратной обработки информации в составе защищенной автоматизированной системы обработки данных (АСОД).

2 Тактико-технические требования к выполнению ВКР

Технические требования:

- узел должен быть выполнен на отечественной элементной базе;
- узел должен поддерживать следующие входные интерфейсы: Ethernet 100, USB 2.0, аудио (вход и выход);

– узел должен осуществлять криптографическую защиту по алгоритму ГОСТ 28147-89 в режиме гаммирования;

- связь с АСОД должна осуществляться через двухпортовое ОЗУ.

В ВКР должны быть рассмотрены следующие вопросы:

- выбор элементной базы;
- принципы защиты информации на основе унифицированной технологии автоматизированной обработки информации (УТАОИ);
- разработка метода структуризации аппаратного обеспечения защищенных АСОД;
- разработка метода структуризации программного обеспечения (ПО) облегчающего переносимость этого ПО;
- описание принципов построения унифицированного узла;
- разработка схемы унифицированного узла.

3 Требования к разрабатываемой документации

Отчет по ВКР выполняется в соответствии с требованиями ГОСТ 7.32.



Перечень и содержание графической части ВКР.

- схема УТАОИ- 1л., ф. А1 (плакат);
- принцип построения переносимых программ-1л., ф. А1 (плакат);
- структурная схема унифицированного узла - 1л., ф. А1;
- функциональная схема унифицированного узла- 3л., ф. А1.

Электронная копия отчета и графической части ВКР должна быть представлена на магнитном или оптическом носителе.

4 Сроки выполнения ВКР

Тема утверждена приказом ректора ПГУ № 491а/о от 25.04.2016г.

Дата выдачи задания «20» февраля 2016г.

Время дипломного проектирования с 18.04 2016г. по 10.06 2016г.

Задание к исполнению принял «___» _____ 2016г.

Исполнитель ВКР _____ Погодина А.Е.

6

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

АСОД – автоматизированная система обработки данных

АЦП – аналого-цифровой преобразователь

ОЗУ – оперативное запоминающее устройство

ПЭМИН – побочные электромагнитные излучения и наводки

ПЗУ – постоянное запоминающее устройство

ПЛИС – программируемая логическая интегральная схема

ПО – программное обеспечение

САПР – система автоматизированного проектирования

УКЗИ – устройство криптографической защиты информации

УТАОИ – унифицированная технология автоматизированной обработки информации

ЦАП – цифро-аналоговый преобразователь

ЭВМ – электронная вычислительная машина

ЯПВУ – язык программирования высокого уровня

ВВЕДЕНИЕ

Разработка аппаратуры, обрабатывающей закрытую информацию, значительно сложнее разработки аналогичной аппаратуры, предназначенной только для открытой информации. Если в последнем случае достаточно правильно подобрать элементную базу и средства программирования, собрать и настроить аппаратуру, то в первом случае весь цикл разработки будет сопровождать контролирующая организация. Она предъявит свои требования к аппаратным и программным решениям в этой аппаратуре, а потом проведет исследования по оценке уровня и информативности побочных электромагнитных излучений и наводок (ПЭМИН), которые будет излучать вновь разработанная аппаратура. Для сокращения объема работ контролирующей организации выгодно широкое использование в новых разработках отработанных и хорошо зарекомендовавших себя заделов в программной и аппаратной частях.

Однако это требование вступает в противоречие с нынешним состоянием дел в современной элементной базе. Цифровая элементная база представлена двумя основными классами интегральных схем (ИС) – процессорами и программируемыми логическими интегральными схемами (ПЛИС). Все они отличаются высокой сложностью и жизненный цикл у всех у них очень мал и измеряется несколькими годами – обычно пять лет. Это сильно затрудняет использование задела в новых разработках и заставляет искать новые методы проектирования аппаратуры, обрабатывающей закрытую информацию, которые бы снизили трудоемкость ее разработки. Так возникает проблема переносимости аппаратных узлов (проектов для ПЛИС) и, особенно, программ на другую элементную базу – со старой на новую или на имеющуюся в наличии.

Для проектов ПЛИС проблема переносимости успешно решена путем использования языков описания аппаратной части – VHDL или Verilog. Системы автоматизированного проектирования (САПР) всех фирм-производителей ПЛИС поддерживают эти языки, и поэтому для переноса

проекта с одной ПЛИС на другую достаточно просто иметь описание этого проекта на языке VHDL или языке Verilog.

Гораздо сложнее дела обстоят с переносимостью программного обеспечения (ПО). Здесь тоже были приложены большие усилия к созданию удобочитаемых, в отличие от машинных кодов и языков типа ассемблера, языков программирования высокого уровня (ЯПВУ). К настоящему времени, как известно, на роль универсального языка выдвинулся язык Си.

В течение 30 лет ЯПВУ хорошо справлялись с задачей переносимости ПО – до тех пор, пока компьютеры имели небольшую номенклатуру однотипной периферии и использовались как законченные устройства для вычислений.

Положение резко изменилось с появлением однокристальных специализированных процессоров, каждый из которых может иметь свое адресное пространство, свою разнообразную периферию. Кроме того, такой процессор в отличие от прежних больших ЭВМ, предназначен не для простого выполнения программ, а для программной реализации функций в составе аппаратов. То есть, к программе для такого процессора фактически изначально предъявляется требование переносимости. Но как раз оно-то и затрудняется самой архитектурой специализированных однокристальных процессоров.

Язык Си перестает обеспечивать переносимость программ между специализированными процессорами из-за различия их периферии. Там, где у одного процессора находятся регистры какого-либо периферийного устройства, у другого будут просто ячейки памяти. Все это вынуждает искать новые подходы к переносимости программ.

В данной работе намечен общий подход к структуризации аппаратной и программной частей аппаратуры на основе предложенной профессором В.А. Герасименко унифицированной технологии автоматизированной обработки информации (УТАОИ).

При этом целью работы поставлена разработка унифицированного узла для программно-аппаратной обработки информации в составе защищенной

автоматизированной системы обработки данных (АСОД). Этот узел может быть использован в качестве универсального периферийного устройства для новых разработок. Он отличается тем, что все функции по управлению периферией этого узла сосредотачиваются в самом этом узле. Вся информация – как передаваемая, так и принимаемая этим узлом, записывается в определенные зоны оперативного запоминающего устройства (ОЗУ), которое может быть отображено в адресное пространство другого (центрального) процессора. То есть программа центрального процессора таким образом освобождается от функций управления периферией, а вся проходящая через периферийные устройства информация видится для программы центрального процессора как информационные массивы разной длины и с разными начальными адресами. А это позволит легко переносить программу центрального процессора при помощи языка Си.

1 Подход к структуризации и переносимости программного обеспечения

1.1 Развитие подходов к переносимости программного обеспечения

Потребность в переносимости программ между электронно-вычислительными машинами (ЭВМ) разных типов впервые возникла на рубеже 1950-х – 1960-х годов, когда рост парка ЭВМ привел к резкому увеличению масштабов использования средств программирования и потребовал перехода от хаотичного подхода к проектированию ПО, приемлемого для отдельных научных вычислительных центров, к машинно-независимому и высокоуровневому программированию [1,2]. Так появились языки программирования высокого уровня (ЯПВУ), трансляторы и операционные системы, полностью решавшие задачу переносимости ПО до начала 1980-х годов. Следует отметить, что первый компилятор (с языка логических схем алгоритмов) под названием «программирующая программа» (ПП-1) был создан в СССР в 1954 году Институтом математики АН СССР [3].

Однако бурный прогресс микроэлектроники породил большую номенклатуру специализированных микропроцессоров с богатым набором периферийных устройств, каждое из которых имеет собственный интерфейс. Для переноса программ между такими микропроцессорами недостаточно уже трансляции текста с ЯПВУ в машинный код. Необходима еще и модификация всех заложенных в программу взаимодействий с периферией. Так развитие микропроцессоров и усложнение их функциональности снова ставит в повестку дня проблему переносимости программ. Но в этом случае никакой ЯПВУ, в том числе и Си, сам по себе проблему переносимости не решит. Необходимы другие подходы.

В работе [4] предложено целенаправленно внедрить свойство переносимости в цикл разработки ПО. Процесс переноса ПО предлагается разделить на два этапа – транспортирование (физическое перемещение ПО) и адаптацию (ручную, автоматизированную или автоматическую модификацию переносимого ПО). Для количественной оценки переносимости предложен экономический показатель, названный степенью переносимости (СП),

определяемый следующим образом:

$$СП = 1 - \frac{\text{цена переноса ПО}}{\text{цена новой разработки ПО}}.$$

Если значение СП больше нуля, то перенесение эффективнее новой разработки. Различные взаимодействия программы с внешним окружением в работе [4] названы интерфейсами. Программная унификация этих интерфейсов в некотором классе сред сделает использующее их ПО переносимым между данными средами. Предлагаются три принципа увеличения переносимости ПО:

- управление интерфейсами;
- выделение зависимостей;
- тонкий перенос.

Первый принцип предполагает стандартизацию всех интерфейсов ПО со средой. Второй принцип предполагает выделение фрагментов программного модуля, подлежащих адаптации, и их последующую изоляцию (например, в небольшое число отдельных модулей). Третий принцип наиболее фундаментален из всех и предполагает учет возможностей будущего перенесения нового ПО на всех этапах его разработки: специфицировании, проектировании, компиляции, тестировании и верификации.

На этапе специфицирования следует обозначить возможность будущей переносимости ПО как явной цели и ее увязки с другими системными требованиями. Это не должны быть абстрактные требования вида «эта программа должна быть легко переносима на новые платформы». Вместо этого нужно охарактеризовать переносимость способом, который будет выполнимым и измеряемым. На этапе проектирования необходимо использовать стандартизированные интерфейсы с внешней средой, а системные зависимости выделять в отдельные программные модули.

На этапе компиляции необходимо выбрать хорошо стандартизированные и общедоступные ЯПВУ. Необходимо также выделить

своего рода «переносимое подмножество» этого языка, поддерживаемое в тех средах, куда предполагается переносить разрабатываемое ПО. Для связи со средой следует использовать только стандартизированные ранее интерфейсы.

Для этапа тестирования выдаются следующие рекомендации:

- разработка повторно используемых тестов для системно-независимой части ПО и отдельных тестов для системно-зависимых программных модулей;
- тщательная регистрация всех выявленных ошибок;
- непосредственное тестирование переносимости как свойства будущего ПО. Это потребует анализа, сравнивающего ПО с одним или несколькими типичными средами в классе назначения. Такое сравнение может использоваться для оценки затрат перенесения и последующего вычисления степени переноса ПО.

Наконец, при формальной верификации возникает вопрос о том, какая часть проведенных ранее верификационных испытаний должна быть повторена после перенесения ПО для того, чтобы гарантировать сохранение свойств перенесенного ПО и, следовательно, справедливость ранее сделанного заключения. Это – открытая область для исследований [4].

1.2 Метод структуризации программного обеспечения, облегчающий его переносимость

В данной работе разработан унифицированный узел для взаимодействия защищенной АСОД с периферией. Поэтому, из перечисленных выше особенностей проектирования переносимых программ в данной работе наиболее важным будет выделение в отдельные подпрограммы фрагментов программного кода, требующих изменения при переносе программы. В данном случае такими фрагментами будут обращения к периферийным устройствам процессора. Взаимодействие с оператором с алгоритмической точки зрения ничем не будет отличаться от обмена с периферией, но этот вопрос в этой работе не рассматривается.

С алгоритмической точки зрения взаимодействие программы с периферийными устройствами сводится к обмену информацией между

процессором и периферийным устройством в соответствии с некоторым арбитражем, который обычно определяет:

- порядок обмена информационными сигналами (для периферийного устройства это адреса и данные, для оператора – вводимая и выводимая информация);

- порядок обмена служебными сигналами для периферийных устройств (прерывания, готовности, арбитраж прямого доступа к памяти).

Взаимодействие с периферийными устройствами может иметь вид либо отдельных подпрограмм, либо отдельных команд или операторов, разбросанных по основной программе. Ясно, что в последнем случае программа будет непереносимой.

При переносе программы указанные подпрограммы обмена могут потребовать модификации. Для возможности стыковки таких подпрограмм с переносимыми программами должен быть стандартизирован протокол обмена с ними, который должен соблюдаться уже при написании переносимой программы.

Наиболее простой способ информационного взаимодействия подпрограмм (процессов) между собой – через выделенные области памяти. Память у любого процессора одинаковая, а использовать полученную информацию он будет в зависимости от своей программы. Поэтому, для целей данной работы необходимо использовать следующую методику структуризации ПО:

- всякая программная поддержка периферийных устройств должна быть выделена в отдельные подпрограммы;

- процессор, выполняющий основную программу, не должен быть занят настройкой периферийных устройств;

- обмен процессора с любыми периферийными устройствами должен сводиться к пересылке некоторого массива информации в собственную область памяти, одновременно доступную и для соответствующего периферийного устройства;

— программная поддержка периферийных устройств должна быть выполнена на отдельных микроконтроллерах этих самых устройств.

Следует отметить, что текст главной программы должен быть написан на ЯПВУ, в качестве которого сейчас используются разные варианты языка Си. Для микроконтроллеров обычно используются C++ и ANSI C. В этом случае главная программа будет полностью переносимой, так как кроме как с собственной памятью выполняющий ее процессор ни с чем взаимодействовать не будет. Подпрограммы обслуживания периферийных устройств могут быть написаны на любом языке программирования, так как в случае замены микроконтроллера их все равно придется переписывать.

Таким образом, непереносимая программа рассчитана на один микропроцессор и включает следующие основные операции, в произвольном порядке разбросанные по тексту программы:

- вычисления;
- обмен с памятью;
- настройка периферии;
- обмен с периферией.

Переносимая программа должна быть рассчитана на вычислительную систему, состоящую из главного процессора, на котором выполняется основная переносимая программа, и связанные с ним через общую двухпортовую память периферийные устройства, каждое из которых может иметь собственную (непереносимую) программу, обслуживающую эту периферию. Принцип построения переносимых программ представлен на рисунке 1.

Он заключается в том, что переносимая программа строится не для конкретного специализированного микроконтроллера, а для вычислительной системы, в которой связь между главным процессором, выполняющим главную программу и периферией осуществляется через двухпортовое ОЗУ, которое одновременно вписано в область адресов памяти как главного процессора, так и периферийного контроллера. Все подпрограммы управления периферией должны быть выполнены только на периферийном контроллере. Главная же

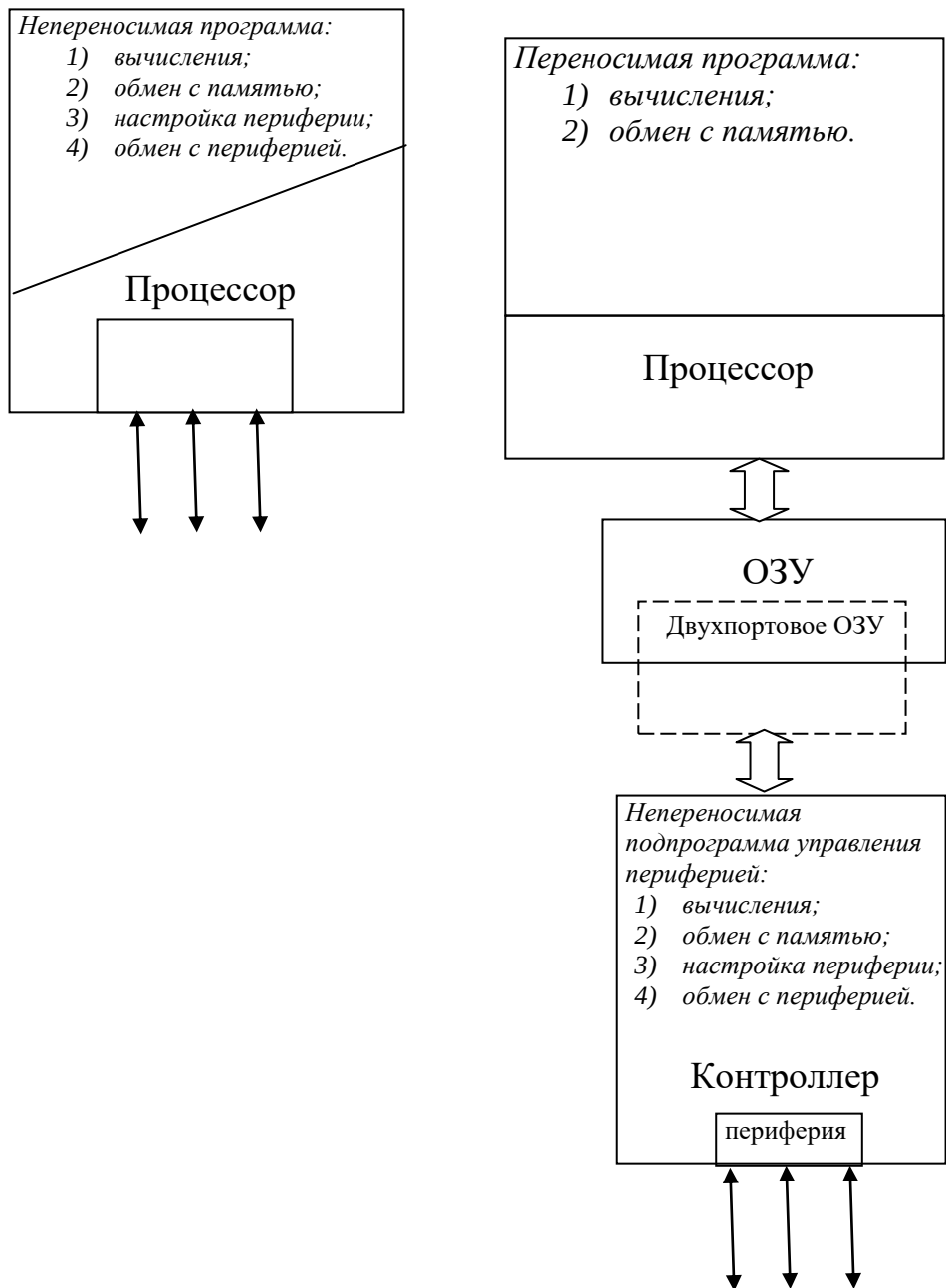


Рисунок 1 — Принцип построения переносимых программ
программа таким образом избавляется от явной связи с периферией и становится переносимой на другие процессоры.

2 Защита информации и унифицированная технология автоматизированной обработки информации

2.1 Необходимость комплексного подхода к защите информации

Информация, накапливаемая, хранимая и обрабатываемая в автоматизированных системах обработки данных (АСОД) является сильно уязвимой, причем как с точки зрения опасности ее искажения или уничтожения (нарушения физической целостности), так и с точки зрения несанкционированного доступа к ней лиц, не имеющих на это полномочий, и получения ее в злоумышленных целях. Но если первую опасность специалисты предполагали изначально и поэтому предусматривали соответствующие средства и методы защиты, то вторая стала неожиданностью. В зарубежной печати, начиная с семидесятых годов, периодически публиковались сведения о конкретных данных хищения информации, находящейся в АСОД. Внезапность, повышенная эффективность, некоторая детективность указанного явления способствовали тому, что защита информации сконцентрировалась, главным образом, вокруг обеспечения ее безопасности, то есть предупреждения несанкционированного получения. В этом плане истекшее время характеризовалось постоянным противоборством злоумышленников и специалистов по защите.

Однако по мере развития вычислительной техники и расширения масштабов и сфер ее применения появились и новые угрозы информации, причем особую опасность представляют две: первая — несанкционированная модификация находящейся в АСОД информации (особенно, заражение машинных программ так называемыми электронными вирусами) и вторая — несанкционированное размножение машинных программ и информационных массивов с целью наживы. Разрозненная защита информации по каждому виду опасности оказалась и недостаточно эффективной и дорогостоящей; объективно назрела необходимость разработки концепции комплексной защиты. Но в соответствии с логикой здравого смысла комплексной постановке задачи защиты информации должны соответствовать и комплексные подходы к

ее решению, причем как с точки зрения расширенной интерпретации самого понятия защиты, так с точки зрения активности концепции защиты, то есть влияние их на концепции построения и организации функционирования АСОД.

На основе изложенных оценок проблемы защиты информации в современных АСОД на факультете защиты информации Российского государственного гуманитарного университета под руководством профессора Герасименко В.А. сформирована и научно обоснована концепция управления комплексной защиты информации и предложена УТАОИ. Эта технология представляет собою высокоупорядоченный конвейер автоматизированной обработки информации.

При такой строго определенной структуре УТАОИ создаются хорошие предпосылки для унификации подходов к обеспечению в ней качества информации. В самом деле, нетрудно видеть, что для обеспечения требуемого качества информации достаточно обеспечить его на каждом из этапов технологии и в интерфейсах между этапами. Строгая же функциональная определенность каждого из этапов позволяет целенаправленно решать все соответствующие вопросы. То же самое можно сказать и относительно интерфейсов между этапами. Однако унификацию подходов к управлению качеством информации можно распространить еще дальше. Основанием для этого может послужить возможность унифицированного представления структуры этапов УТАОИ.

2.2 Постановка задачи комплексной защиты информации в современных автоматизированных системах обработки данных

Под защитой информации в АСОД понимается использование в них средств и методов, принятие мер и осуществление мероприятий с целью обеспечения требуемой защищенности информации. В свою очередь под защищенностью информации понимается поддержание на заданном уровне тех параметров находящейся в АСОД информации, которые характеризуют установленный статус ее хранения, обработки и использования.

Одним из важнейших выводов, полученных на основе результатов

теоретических исследований и анализа опыта практического решения проблем защиты, является вывод о том, что современным потребностям и условиям защиты информации адекватным может быть только комплексный подход к решению данных проблем. При этом само понятие комплексности является сложным и включает в себя, по крайней мере, следующие аспекты:

— во-первых, комплексность целевую, то есть защиту по всей совокупности показателей защищенности информации и всей совокупности факторов, влияющих на защищенность;

— во-вторых, комплексность временную, то есть непрерывную защиту информации во все время и на всех этапах жизненного цикла АСОД;

— в-третьих, комплексность концептуальную, то есть изучение и реализация проблем защиты в общей совокупности всех проблем развития, построения и использования АСОД.

Содержание целевой комплексности, то есть структура и общее содержание целей комплексной защиты информации представлена на рисунке 2.

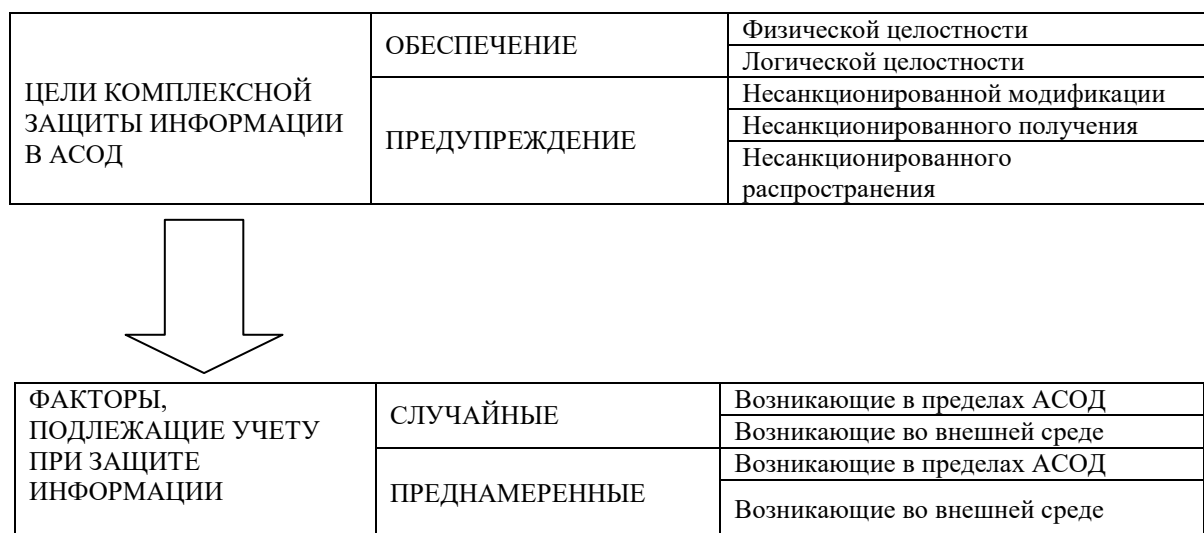


Рисунок 2 — Структура и общее содержание целей комплексной защиты информации в АСОД

Временная комплексность, как это сформулировано выше, предполагает непрерывность осуществления мероприятий по защите информации и притом не только в процессе непосредственной автоматизированной обработки

информации, но и на всех других этапах жизненного цикла АСОД. Это обстоятельство само собой предполагает организацию и обеспечение целенаправленного управления всей совокупности этих мероприятий.

Концептуальная комплексность интерпретируется в том смысле, что потребности, возможности и условия защиты информации должны органически учитываться в концепциях построения и организации функционирования АСОД, то есть в концепциях развития и реализации концепций информатизации общества.

2.3 Унифицированная технология автоматизированной обработки информации

В работе [5] показано, что наилучшим образом задача комплексной защиты информации реализуется при использовании модели УТАОИ в качестве основы для построения защищенной АСОД. Для защиты такой АСОД необходимо в первую очередь защищать информацию на каждом этапе УТАОИ. Суть же УТАОИ сводится к структуризации схемы обработки информации подобно тому, как современное промышленное производство основано на разбиении сложного технологического процесса на менее сложные операции. Как для выполнения отдельной операции нужна меньшая квалификация работника, чем для выполнения всего технологического процесса, так и менее сложный отдельный этап УТАОИ легче защитить, чем весь алгоритм обработки информации в целом.

Структуризация технологических схем обработки информации в АСОД относится к числу условий, играющих повышенную роль в плане эффективности управления качеством информации, в том числе (и особенно) с точки зрения ее защиты. Такая роль структуризации технологии обработки обусловлена целыми рядом обстоятельств, а именно:

- информация наиболее уязвима именно на технологических маршрутах ее обработки;

- технологические схемы обработки информации в современных АСОД являются чрезвычайно сложными, а потому труднотруктурируемыми;

— структуризации технологических схем автоматизированной обработки информации до настоящего времени очень мало уделялось внимания.

Ранее отмечалось, что подавляющее большинство технологических схем обработки может быть представлено совокупностью участков трех типов: линейных, ветвящихся и циклических [6]. Этим создаются объективные предпосылки для структуризации технологических схем. Однако проведенные в последнее время исследования показали, что в настоящее время имеются все объективные предпосылки для построения технологии, полностью структурированной и универсальной широким диапазоне практических приложений. Для этого и была создана УТАОИ.

Чтобы быть в достаточной степени универсальной, УТАОИ должна:

— обеспечивать комплексную обработку входных, внутренних и выходных потоков информации;

— обеспечивать комплексную обработку всех видов документальной и фактографической информации;

— обеспечивать требуемую обработку информации, необходимой для решения всей совокупности функциональных задач в системе;

— обеспечивать в полном объеме как групповое, так и индивидуальное (персональное) информационное обеспечение всех пользователей и абонентов системы;

— создавать наиболее благоприятные условия для обеспечения требуемого качества информации;

— создавать всем пользователям и абонентам системы возможно более комфортные условия работы с информацией.

Чтобы быть полностью структурированной, УТАОИ должна удовлетворять, по крайней мере, такой совокупности условий:

— сквозное модульное построение на всех уровнях (от самой общей схемы до блок-схемы элементарных технологических операций);

— автономность организации всех модулей на всех уровнях;

- полная структуризация внутренней организации всех модулей на всех уровнях;
- структуризация всех видов обеспечений.

В соответствии с основными концептуальными посылками УТАОИ представляет собою непрерывный и высокоупорядоченный конвейер автоматизированной обработки всей информации, поступающей в систему (предприятие, организацию, учреждение), обращающейся внутри системы и выдаваемой из нее другим системам. Тогда в самом общем виде структуру УТАОИ можно представить так, как на рисунке 3.

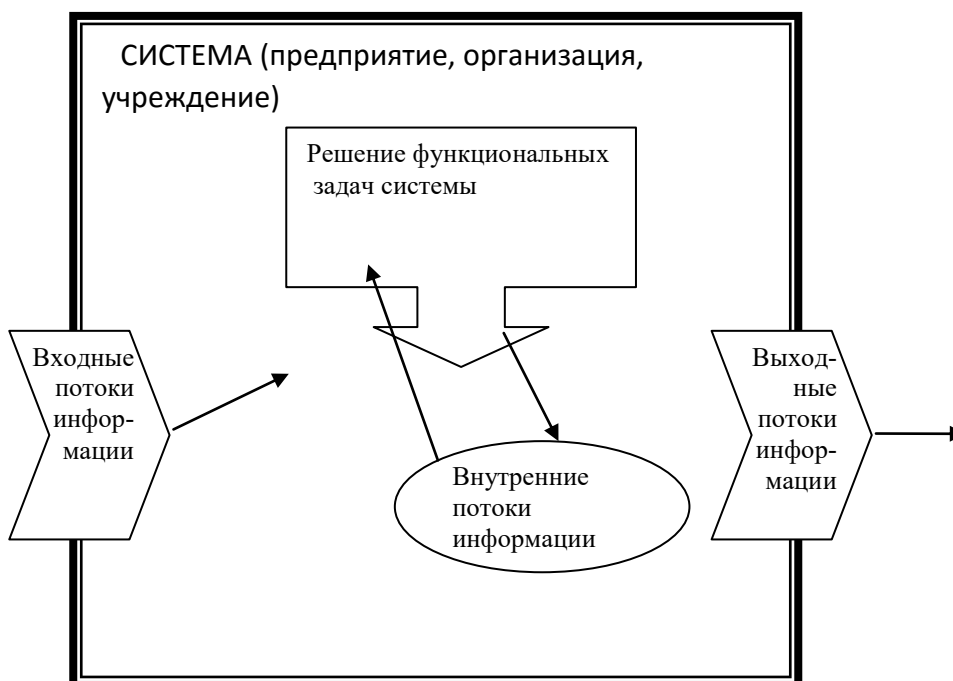


Рисунок 3 — Общая структура УТАОИ

Уже из такого обобщенного представления УТАОИ можно заключить, что все процедуры обработки информации в ней делятся на три группы:

- А — прием всех потоков входной информации;
- Б — организация циркуляции и обработка внутренних потоков информации;
- В — организация выходных потоков информации.

На структуру, содержание и последовательность процедур обработки внутренних потоков информации определяющее влияние оказывают, с одной

стороны, функциональное назначение информации в системе, а с другой - потенциальные возможности современных средств и методов обработки информации, естественно, при их рациональном использовании. С учетом названных факторов основные цели комплексной автоматизации обработки внутренних потоков информации могут быть сформулированы следующим образом:

- создание наиболее благоприятных условий для повышения общей организованности и регулярности обработки информации;

- обеспечение своевременной и рациональной обработки быстро меняющейся информации в реальном масштабе времени;

- оптимизация использования всей поступающей информации;

- максимизация объемов производной информации, получаемой путем переработки информации, извлекаемой из входных потоков в процессе их обработки;

- сокращение сроков обработки информации по всему циклу ее циркуляции;

- создание наиболее благоприятных условий для регулярного накопления рационального хранения и использования всей появляющейся в системе и необходимой для ее функционирования информации, включая и первоисточники;

- обеспечение требуемого качества информации по всей совокупности показателей;

- максимизация эффективности функциональной обработки информации решения функциональных задач системы;

- создание специалистам руководителем системы наиболее благоприятных условий для работы с информацией.

- максимальное высвобождение специалистов и руководителей системы от выполнения нетворческих процедур, связанных с обработкой информации;

- создание благоприятных условий для максимального использования

интеллектуальных ресурсов системы в процессе обработки и использования информации;

- создание условий для максимальной рационализации труда технического и обслуживающего персонала системы;

- обеспечение свободной адаптации технологии обработки на любые структуры и состав входных потоков информации, на любые структуры и функциональное назначение системы на любые условия, складывающиеся в процессе обработки информации;

- минимизация затрат на разработку, внедрение обеспечение функционирования информационных технологий.

Перечисленная совокупность целей может быть достигнута, если технология обработки внутренних потоков будет построена с учетом следующих принципов:

- унификация представления содержания обработки информации для самого общего случая;

- полная структуризация унифицированного содержания обработки информации;

- возможно более четкое разделение формализуемых и неформализуемых процедур обработки информации;

- максимальное освобождение специалистов и руководителей от выполнения формализуемых процедур обработки;

- максимально возможная алгоритмизация (схематизация) неформализуемых процедур обработки информации;

- автономная обработка быстроменяющейся информации;

- самостоятельная организация массивов информации на каждом из этапов ее обработки;

- разделение процессов обработки исходных и регламентных данных;

- разделение процедур обработки поступающих потоков информации и процедур функционирования обработки массивов регламентных данных.

Перечень и содержание процедур обработки выходных потоков

информации определяется совокупностью следующих параметров:

- вида выдаваемой информации;
- канала выдачи;
- степени регламентированности выдачи;
- форма выдачи.

Классификация и идентификация выходной информации по первым двум параметрам, очевидно, будут аналогичны классификации и идентификации входной информации.

С учетом всего изложенного структуру и содержание УТАОИ в самом общем, укрупненном виде можно представить так, как показано на рисунке 4.

Прием входных потоков информации определяется как совокупность процедур, основное содержание которых сводится к фиксации поступающей информации в буферных запоминающих устройствах, проверки правильности передачи информации по внешним признакам и организацию передачи ее обработки. Обработка быстро меняющейся информации в реальном масштабе времени объединяет всю совокупность процедур, необходимых для решения в системе задач оперативно-диспетчерского управления.

Под базовой обработкой входной информации организацией исходных данных следует понимать всю совокупность процедур, предназначенных для того, чтобы вся поступившая информация всех видов была обработана в установленные сроки, и чтобы при этом из поступившей информации были извлечены рациональным образом и организованы все те данные, которые могут понадобиться для организации и обеспечения функционирования системы. С точки зрения построения технологии обработки в соответствии с приведенными выше целями и принципами совокупность этих процедур целесообразно осуществлять три этапа:

- организация (формирование) массивов поступивших входной информации;
- собственно обработка с целью извлечения необходимых данных;
- организация (формирование) массивов выходной информации и



Рисунок 4 — Укрупненная схема УТАОИ

извлечение (исходных) данных с целью последующего использования. Извлеченные и накопленные исходные данные в установленные регламентом функционирования системы должны подвергаться совокупной обработке с

целью формирования таких данных, которые принимаются в качестве регламентных и будут затем использоваться для решения функциональных задач системы. Процесс формирования регламентных данных на основе исходных назван аналитико-синтетической обработкой.

Ввиду повышенной сложности и слабой структурированности аналитико-синтетической обработки ее целесообразно выделить в самостоятельный этап УТАОИ.

Полученные в итоге аналитико-синтетической обработки регламентные данные подлежат накоплению в самостоятельные массивы (базы). При этом, поскольку регламентные данные используются для решения всех функциональных задач системы, к самому процессу ведения этих массивов (баз) должны быть предъявлены повышенные требования с точки зрения обеспечения целостности данных, соблюдения всех юридических норм и правил обращения с информацией. Кроме того, при надлежащей организации процесса формирования и ведения массивов (баз) регламентных данных многие задачи функциональной обработки этих данных вполне могут решаться без непосредственного участия специалистов и руководителей основных подразделений системы, что имеет важное значение для достижения целей комплексной автоматизации обработки данных. Всем этим обуславливается настоятельная целесообразность выделения всей совокупности процедур ведения баз регламентных данных в самостоятельный этап УТАОИ. Функциональная обработка данных по самому своему названию включает всю совокупность процедур, необходимых для информационного обеспечения всех задач, решаемых в процессе функционирования системы. В то же время в силу чрезвычайно большого разнообразия систем, для обеспечения функционирования которых предназначается УТАОИ, дальнейшая классификация рассматриваемых процедур по функциональному признаку является затруднительной. Поэтому предлагается функциональную обработку данных представить в УТАОИ в виде одного обобщенного этапа. Под выдачей данных, как этапом УТАОИ, понимается вся совокупность процедур,

осуществляемых с целью организации и обеспечения выдачи всех видов информации по всем каналам выдачи. Всю совокупность этих процедур по тем же причинам, которые указаны выше, предлагается также представить в виде одного этапа УТАОИ.

С учетом всего изложенного общая схема УТАОИ может быть представлена в виде девяти этапов автоматизированной обработки информации, как показано на рисунке 5.

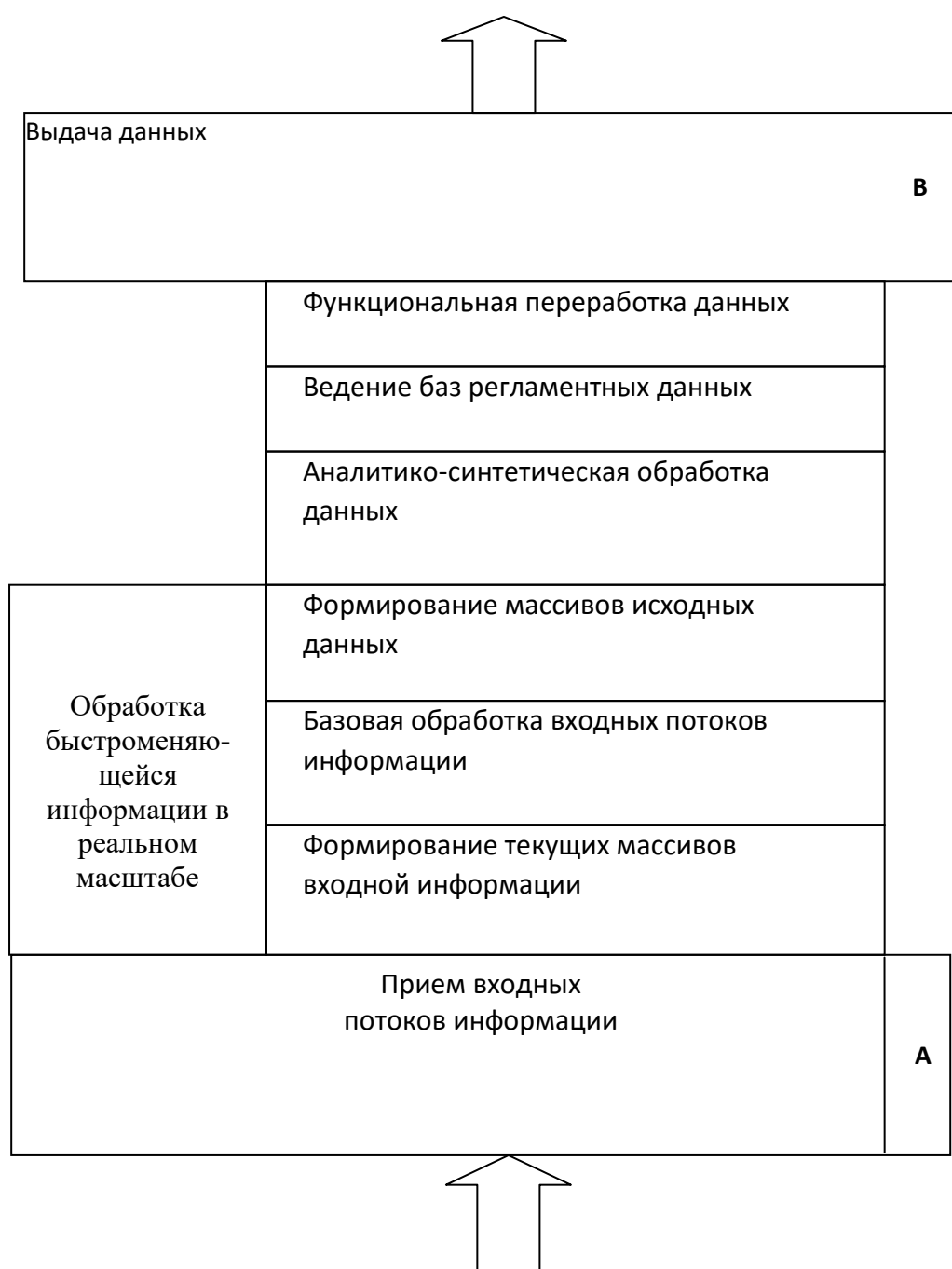


Рисунок 5 — Последовательность и общее содержание этапов УТАОИ

3 Принципы структуризации аппаратных узлов

3.1 Принципы и методы структурирования архитектуры технических систем и процессов их функционирования

Структурирование представляется как расширительная (относительно формализации) методология описания архитектуры больших систем и процессов их функционирования. В традиционной (формальной) теории систем предполагается, что указанные атрибуты моделируемых систем представлены в строго формальном описании. Многолетняя и многообразная практика показала, что для технических систем такое представление, как правило, легко может быть осуществлено средствами классической теории множеств и смежных научных дисциплин. Но формализация описания систем организационного типа, то есть таких систем, основу структурного построения которых составляют люди и коллективы людей, а основу процессов функционирования — целенаправленная деятельность людей, наталкивается на значительные трудности, попытка преодоления которых традиционным путем, как правило, ведет к потере существенно значимых характеристик системы. Для преодоления указанных трудностей без потери существенно значимых характеристик систем и формируется методология структурирования, которая в настоящее время находится лишь в начальной стадии своего становления.

Методология структурирования формируется таким образом, чтобы предоставить пользователю достаточный инструментарий для адекватного описания систем любой природы и чтобы как в процессе разработки, так и в процессе использования соответствующих моделей преодолевались перечисленные выше трудности. Следует отметить, однако, что рассматриваемая методология находится лишь в самой начальной стадии формирования.

Однако сама по себе идея структурирования каких-либо систем и их компонентов не является новой. Так, господствующее ныне во всех промышленно развитых странах поточно-индустриальное производство

непрерывно предполагает структурированность как производимых изделий, так и технологии их производства.

Поточное производство определяется как метод организации производства, при котором изготовление деталей, сборка машин, возведение сооружений производятся в поточных линиях, представляющих собой совокупность обрабатывающих машин, рабочих мест или объектов производства, расположенных по ходу технологического процесса. За каждой машиной или рабочим местом поточной линии закрепляется одно или несколько изделий, изготавливаемых одновременно или попеременно. Поточное производство характеризуется непрерывностью процесса производства, то есть передачей изделий на следующую операцию немедленно после предыдущей при помощи специальных межоперационных транспортных устройств или передвижением рабочих и материалов по ходу технологического процесса при неподвижности объекта производства. Это обеспечивает равномерный ход производства, ритмичный запуск и выпуск продукции, высокую производительность и снижение себестоимости изготавливаемого изделия.

Таким образом, в основе поточного производства лежат два основных принципа:

- строгая регламентация технологического процесса производства;
- специализация исполнителей и оборудования.

При этом предполагается, что конструкция производимого изделия отвечает условиям поточного производства, что предполагает:

- возможность расчленения изделия на структурно и функционально самостоятельные узлы (агрегаты), причем такое расчленение должно быть многоуровневые иерархическим;
- агрегируемость, то есть соединяемость деталей в узлы, узлов в агрегаты и т.д. до полной сборки машины (изделия);
- максимальную простоту, унифицированность и стандартность конструктивных решений и технологических операций.

Непременными условиями эффективности поточного производства также являются:

- применение передовых приемов производства;
- планирование и организация производства;
- управление процессом производства.

Изложенные принципы производства реализованы также и при промышленном производстве технических средств обработки информации, и прежде всего — современных технических средств вычислительной техники. Поэтому применительно к техническому обеспечению современных систем обработки данных можно утверждать, что оно или уже является полностью структурированным или для такой структуризации имеются все необходимые предпосылки. Другие компоненты системы (особенно математическое, программное и информационное обеспечение), не имея аналогов в предшествующем развитии техники и производства, развивались в значительной мере стихийно, вследствие чего оказались практически неструктурированными. Отрицательные последствия такого положения начали сказываться тогда, когда названные выше компоненты стали перерастать в большеобъемные, а их разработка стала все более принимать характер промышленного производства. Неструктурированность компонентов в этих условиях стала угрожать весьма серьезными и непредсказуемыми последствиями. Такое положение особенно остро и наглядно проявилось в конце 1960-х и начале 1970-х годов относительно программного обеспечения для ЭВМ и информационно-вычислительных систем и сетей.

До тех пор, пока разрабатываемые для ЭВМ и построенных на их основе систем программы были сравнительно небольшого объема, а использовались они эпизодически и автономно, неструктурированность программ практически не оказывала влияния ни на их разработку, ни на их использование, а потому вопрос о структурировании программ просто-напросто не возникал, в нем не было необходимости. Так и складывался процесс стихийного программирования. Но когда просто программы стали все более

целенаправленно объединяться в развитый и регулярно используемый компонент — программное обеспечение (произошло это, примерно, на рубеже 60-х и 70-х годов), тогда и стали все больше и больше проявляться негативные последствия неструктурированности. Со временем стало ясно, что без принятия решительных и кардинальных мер очень скоро наступит серьезный кризис. Из-за этой опасности и возникло так называемое структурное программирование, основу которого составляет методология разработки структурированных программ [6].

Структурное программирование обуславливается сочетанием особых значений четырех параметров (факторов):

- архитектурного построения разрабатываемой программы;
- принципов разработки программ;
- организации процесса разработки;
- организацией спецификаций на разрабатываемые программы.

В зависимости от проработки каждого из этих параметров довольно четко выделяются три уровня программирования, которые соответственно могут быть названы уровнем стихийного программирования, уровнем полуструктурного программирования и уровнем структурного программирования.

Таким образом, структурным называется такое программирование, в процессе которого канонизированы все четыре параметра:

- архитектура программы;
- принцип разработки программы (только сверху-вниз);
- организация процесса программирования (только по принципу главного программиста);
- спецификации на разрабатываемые программы.

Все перечисленные аспекты к настоящему времени разработаны достаточно полно, апробированы на практике и подтвердили высокую эффективность.

На основе обобщения принципов и методов структурного программирования сформулированы условия структурированности изучаемых

систем и процессов их функционирования и тех принципов, следование которым создает объективные предпосылки для структурированного представления систем и процессов.

Структурированным следует называть такое описание изучаемой системы и (или) процессов ее функционирования, которое системно отвечает следующей совокупности условий:

- полнота, то есть отображение всех существенно значимых элементов и атрибутов системы, а также их взаимосвязей;

- адекватность, то есть обеспечение возможностей воспроизведения в процессе моделирования (имитации) всех существенно значимых характеристик процессов функционирования системы;

- простота внутренней организации элементов описания и взаимосвязей элементов между собою;

- стандартность и унифицированность внутренней структуры элементов и структуры взаимосвязей между ними;

- модульность, то есть автономная организация элементов, позволяющая стандартными способами объединять элементы в сложные структуры и заменять любые элементы и их совокупности;

- гибкость, то есть практически неограниченные возможности расширения и реорганизации структуры одних компонентов модели без изменения (или с несущественными изменениями) других компонентов;

- прозрачность, то есть простота изучения структуры элементов модели, любой их совокупности и взаимосвязей между элементами и их совокупностями;

- инперсонификация, то есть доступность для изучения и использования модели любому специалисту средней квалификации соответствующего профиля.

Приведенная выше система условий структурированности составляет первую часть методологии структурирования. Вторую часть естественно должны составлять те принципы, следование которым создает объективные

предпосылки для структурирования разрабатываемых систем или для построения структурированных моделей существующих неструктурированных систем. Обоснование совокупности этих принципов будем производить на основе анализа проблем структурирования компонентов АСОД. Нетрудно однако будет увидеть, что формулируемые принципы обладают достаточно высокой степенью общности, адекватностью их для широкого спектра различных систем.

Таким образом, относительно структурирования ПО можно говорить о наличии достаточно полной и стройной концепции. В отношении других компонентов систем обработки данных состояние дел с их структурированием далеко не такое благополучное. Общее состояние может быть оценено следующим образом. Различается три уровня в состоянии разработанности проблем структурирования компонентов систем обработки данных:

- осознание целесообразности и актуальности структурирования;
- наличие разработанных принципов и методов;
- наличие практического опыта.

Анализ концепции структурирования разрабатываемых программ дает основания утверждать, что условия структурированности могут быть удовлетворены лишь при комплексном подходе, охватывающем внутреннюю организацию компонентов систем обработки данных структурируемой системы, адекватное отражение этой организации в технической документации (которая сама должна удовлетворять условиям структурированности) и организацию процесса разработки, обеспечивающую создание структурированных компонентов. Именно к такой постановке привело развитие концепции структурного программирования.

Внутренняя организация однозначно определяется структурой составных элементов и структурой связей между ними. Поэтому, чтобы внутренняя организация отвечала условиям структурированности, необходимо:

- во-первых, канонизировать состав элементов, на которые расчленяются соответствующие компоненты на всех иерархических уровнях;

— во-вторых, максимально упростить, унифицировать и стандартизировать внутреннюю структуру элементов;

— в-третьих, максимально упростить и унифицировать структуру взаимосвязей между элементами.

Отсюда можно сформулировать следующие принципы структурирования внутренней организации компонентов систем обработки:

- канонизации состава элементов;
- унификации и стандартизации структуры элементов;
- унификации взаимосвязей между элементами.

Отражение организации в документах будет адекватным, если в каждый момент времени документы будут содержать следующую информацию:

- перечень и идентификаторы всех элементов компонента;
- описание внутренней структуры каждого элемента;
- описание связей между элементами;
- количественные характеристики элементов.

При этом, как указывалось выше, сами документы должны удовлетворять условиям структурированности.

Минимальную совокупность технических документов, адекватно отражающую организацию структурируемого компонента и удовлетворяющую условиям структурированности, будем называть канонической системой спецификаций этого компонента. Тогда следующий, четвертый принцип структурирования можно сформулировать как принцип канонизации системы спецификаций. Большое значение для структурирования компонентов имеет организация их разработки. Очевидно, не всякая организации может обеспечить получение в итоге структурированных компонентов. Поэтому первая проблема, которая при этом возникает, заключается в выборе такого способа проектирования компонентов, который в принципе может привести к структурированным компонентам.

В общем случае возможны два различных способа проектирования компонентов систем обработки данных: сверху вниз и снизу вверх.

Проектирование по принципу сверху вниз заключается в том, что к разработке любого элемента приступают тогда, когда проведена разработка того элемента иерархической структуры, который является включающим для проектируемого. Проектирование по принципу снизу вверх, наоборот, предполагает, что к разработке любого элемента приступают только тогда, когда завершена соответствующая разработка всех элементов, входящих в проектируемый.

Характеристики названных способов проектирования по каждому из условий структурированности приведены в таблице 1, из которой следует что, всей совокупности условий структурированности удовлетворяет способ проектирования сверху вниз, следовательно, следующий, пятый принцип структурирования может быть сформулирован как принцип проектирования по способу сверху вниз.

Таблица 1 — Характеристики способов проектирования по условиям структурированности

Условия структурированности	Способ проектирования	
	Сверху вниз	Снизу вверх
Полнота	Обеспечивается всегда	Не может гарантироваться
Адекватность	Может быть обеспечена	Может быть обеспечена
Простота внутренней структуры	Обеспечивается всегда	Может быть обеспечена лишь в частном случае
Стандартность и унифицированность построения	Обеспечивается всегда	В общем случае не обеспечивается
Прозрачность	Может быть обеспечена	Может быть обеспечена
Модульность построения	Обеспечивается наилучшим образом	Принципиально может быть обеспечена
Гибкость	Обеспечивается наилучшим образом	Принципиально может быть обеспечена
Инперсонификация	Может быть обеспечена	Может быть обеспечена

Но выбранный способ проектирования допускает много различных вариантов организации самого процесса разработок. Поэтому, вторая проблема организации проектирования состоит в том, чтобы из множества возможных вариантов выбрать наиболее эффективный с точки зрения производительности. Очевидно, необходимым условием максимальной производительности является

соответствие способа организации разработки компонентов способу их проектирования. Рассмотренный выше способ проектирования сверху вниз предполагает, что сначала разрабатывается самый верхний элемент структуры компонента (он, как правило, один), затем входящие в него (их, как правило, несколько), затем входящие во входящие и так далее, то есть фронт работ постепенно разрастается, но разрастается дискретно в соответствии с дискретностью иерархической структуры разрабатываемого компонента. Тогда, исходя из отмеченного выше условия соответствия способа организации разработки и способа проектирования компонентов системы обработки, представляется естественной следующая организация разработки. Самый верхний элемент структуры компонента разрабатывает тот специалист, который будет (или является) самым старшим руководителем разработки данного компонента; естественно, предполагается, что этот специалист должен быть и самым компетентным. Помощниками разработчика верхнего элемента должны быть те специалисты, которые затем станут ответственными разработчиками элементов, входящих в данный. Когда эти специалисты приступят к разработке соответствующих элементов, к ним в качестве помощников должны назначаться те специалисты, которые будут разрабатывать элементы следующего уровня и так далее. Описанная схема организации разработки в наглядном виде представлена на рисунке 6.

Нетрудно видеть, что при такой организации процесса разработок обеспечивается последовательная планомерная разработка, причем, поскольку все требования к структуре нижестоящих элементов формулируются в процессе разработки включающего (то есть вышестоящего) элемента, то существенно снижается вероятность принципиальных ошибок. Немаловажным является также то обстоятельство, что структуру каждого элемента будут знать по крайней мере два специалиста, что повышает устойчивость процесса разработки относительно возможных изменений состава разработчиков.

Описанный способ организации применительно к разработке программного обеспечения получил название метода ведущего программиста.

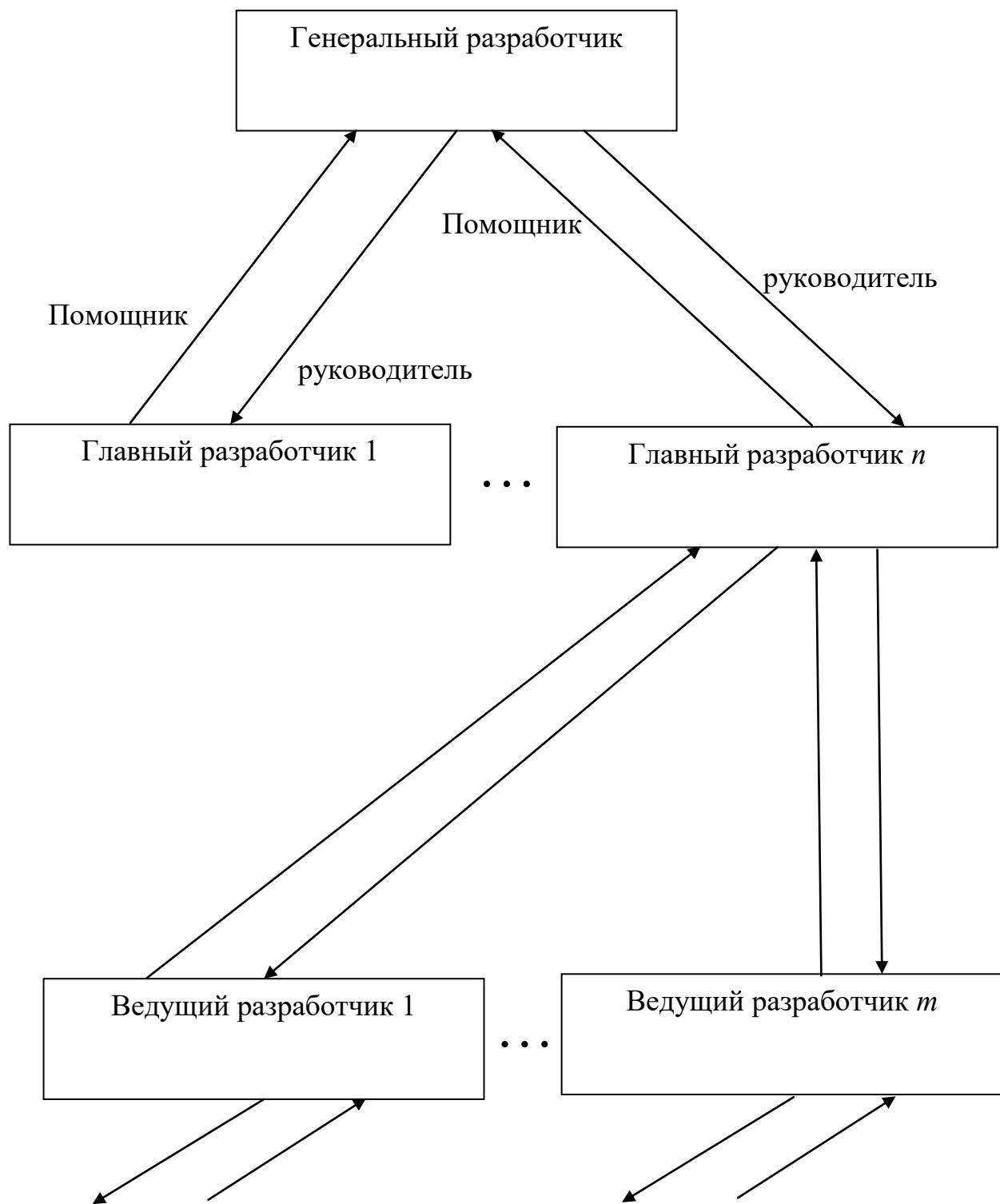


Рисунок 6 — Общая схема организации разработки структурированных элементов систем обработки данных

По аналогии с этим применительно к разработке всех компонентов систем обработки данных для обозначения рассмотренного способа можно предложить термин: «метод ведущего специалиста». Тогда, видимо, шестой

принцип структурирования можно назвать принципом организации проектирования по методу ведущего специалиста.

Поскольку в процессе разработки проектируемый компонент будет постоянно изменять свое содержание, то с точки зрения удовлетворения условиям (структурированности на любом этапе разработки, а не только конечного результата) большое значение приобретает своевременное генерирование необходимого комплекта технических документов на проектируемый компонент и фиксация в этих документах всех изменений, вносимых в компонент. Это может быть достигнуто в том случае, если документы будут вестись параллельно с разработкой соответствующего компонента, а сами документы будут структурированными. Регуляция процесса параллельного ведения документов возможна в том случае, если в группах ведущих специалистов, осуществляющих разработку элементов компонентов систем обработки, будут предусмотрены специальные лица, ответственные за ведение и актуализацию технических документов. В теории и практике структурного программирования специалист, выполняющий указанные функции, назван секретарем-библиотекарем. Представляется, что этот термин может быть распространен также и на другие компоненты АСОД.

В соответствии с изложенным, седьмой принцип структурирования может быть сформулирован как принцип параллельного ведения структурированных документов.

В процессе разработки и практического использования методологии структурного программирования выработан ряд практических правил, руководство которыми способствует успеху разработки, а именно:

- постоянно управлять разработками, используя последовательный план жизненного цикла;
- непрерывно производить оценку и проверку сделанного;
- поддерживать дисциплину разработки;
- использовать в полном объеме принципы структурного программирования;

- трезво оценивать результаты работы;
- использовать ограниченное количество лучших специалистов;
- постоянно содержать комиссию для усиления процесса разработки.

Нетрудно видеть, что перечисленные правила составляют систему (то есть наибольший эффект достигается при следовании всем правилам без исключения), и что эта система правил является целесообразной применительно к разработке широкого класса систем или их моделей.

На основе изложенных принципов должны разрабатываться методы структурирования различных компонентов конкретных систем.

3.2 Виды структуризации

Можно выделить несколько видов структуризации аппаратных узлов. Исторически сложившимся видом является разбиение аппаратных узлов по функциональному признаку. Так, например, в аппаратуре связи выделяют канальную часть (модем), устройство криптографической защиты информации (УКЗИ), узел связи с оконечным оборудованием и другие.

Для аппаратных узлов, не имеющих встроенного программного обеспечения, такая функциональная структуризация вполне достаточна. Однако, без встроенного ПО сейчас не обходится никакое серьезное изделие. И в этом случае функциональная структуризация оказывается недостаточной. Ведь отдельный узел при этом может быть выполнен на специализированном микроконтроллере и получить переносимое ПО. А задача разработчика – максимально использовать существующий задел, в том числе и в программной части, чтобы в процессе разработки новой аппаратуры сократить работы сопровождающей организации.

И в этом случае, как было показано в разделе 1, необходима структуризация аппаратных узлов иного типа. В аппаратных узлах с программной поддержкой должен быть выделен узел главного процессора ни с чем, кроме своей памяти, не взаимодействующего. Он должен выполнять главную программу, которая может быть перенесена на процессор другого типа. А для связи с периферией должны быть выделены отдельные

микроконтроллерные узлы с небольшой (непереносимой) программной поддержкой. Такая структуризация хорошо согласуется и с УТАОИ, в которой весь технологический процесс обработки информации уподоблен производственному конвейеру и разбит на отдельные составляющие.

Итак, для аппаратных узлов с программной поддержкой необходимо учитывать УТАОИ и исключить непосредственное управление периферийными устройствами процессором, выполняющим главную программу.

3.3 Рекомендации по структуризации аппаратных узлов с программной поддержкой

Процессор, выполняющий основную программу, должен взаимодействовать только с памятью. Все передаваемые и принимаемые пакеты должны быть отображены в определенные массивы памяти этого процессора.

Для информирования процессора о приеме или передаче периферийным устройством очередного пакета могут быть использованы прерывания.

Все периферийные устройства должны управляться собственными микроконтроллерами с минимальной программной поддержкой, которая должна сводиться к настройке и переключению режимов устройства, приему и передаче пакетов. Вся связь такого устройства с главным процессором должна осуществляться через двухпортовое ОЗУ, у которого один порт связан с периферией, а другой – с главным процессором.

Для соединения разных периферийных устройств между собой и для бесконфликтного объединения их областей памяти в единое целое должен быть проработан соответствующий протокол, который бы позволял это делать автоматически. Его разработка не входит в число задач данной работы.

4 Выбор элементной базы

4.1 Элементная база отечественного производства

Проблема импортозамещения в течение 1990-х годов практически не рассматривалась. Требования к элементной базе в это время, в связи с экономической и политической ситуацией в стране, были сильно снижены и разработчики пользовались предоставленной возможностью поработать «со всеми передовыми достижениями ведущих мировых фирм». Это положение подкреплялось еще и тем, что в это время практически не было обвального снятия с производства элементной базы. Состояние отечественной элементной базы практически полностью определялось советским заделом.

Положение стало меняться в начале 2000-х годов, когда стали появляться отечественные микросхемы, хоть как-то пригодные для создания новых разработок. В настоящее время имеются и достаточно скоростные процессоры, и достаточно большие ПЛИС отечественного производства. В сегодняшней внешнеполитической обстановке очевидно, что средства защиты информации и военная техника должны строиться только на отечественной элементной базе. Исключение делается только для изделий, не освоенных отечественной промышленностью, например, жидкокристаллических (ЖК) дисплеев среднего и большого размера. Еще отдельные коллективы разработчиков пытаются изыскать возможность применения импортных микросхем из-за их дешевизны и быстрых сроков поставки. От использования импортных микросхем следует отличать так называемые допустимые замены – это когда в изделии вместо какого-либо отечественного элемента разрешается применение его импортного аналога. Отечественный элемент здесь используется, но при сжатых сроках изготовления в крайнем случае может быть применен импортный элемент.

Проектируемый же узел сделан целиком на отечественной элементной базе.

4.2 Интерфейс Ethernet

Технологии Ethernet [7] за последние годы глубоко проникли в нашу

жизнь. Если изначально сети Ethernet применялись только для соединения компьютеров, то сегодня на их основе строят охранные и противопожарные системы, сети цифрового ТВ-вещания и телефонии, организуют удаленный мониторинг и управление различными промышленными объектами. Такая популярность интерфейса Ethernet обусловлена множеством факторов: высокой скоростью передачи данных, простотой организации линий связи, высокой надежностью соединения и защитой от различного рода электромагнитных помех. Неудивительно, что Ethernet востребован и разработчиками специальной техники. Однако до последнего времени для его реализации не было отечественной элементной базы. Ситуация изменилась с появлением ИС компании «ПМК Миландр», позволяющих реализовать интерфейсы Ethernet в специальной аппаратуре.

На сегодняшний день одним из лучших микроконтроллером «ПМК Миландр» с Ethernet-интерфейсом является 32-разрядный микроконтроллер 1986BE3T с двумя независимыми встроенными контроллерами Ethernet 10/100 Base-T. Структура микроконтроллера представлена на рисунке 7.

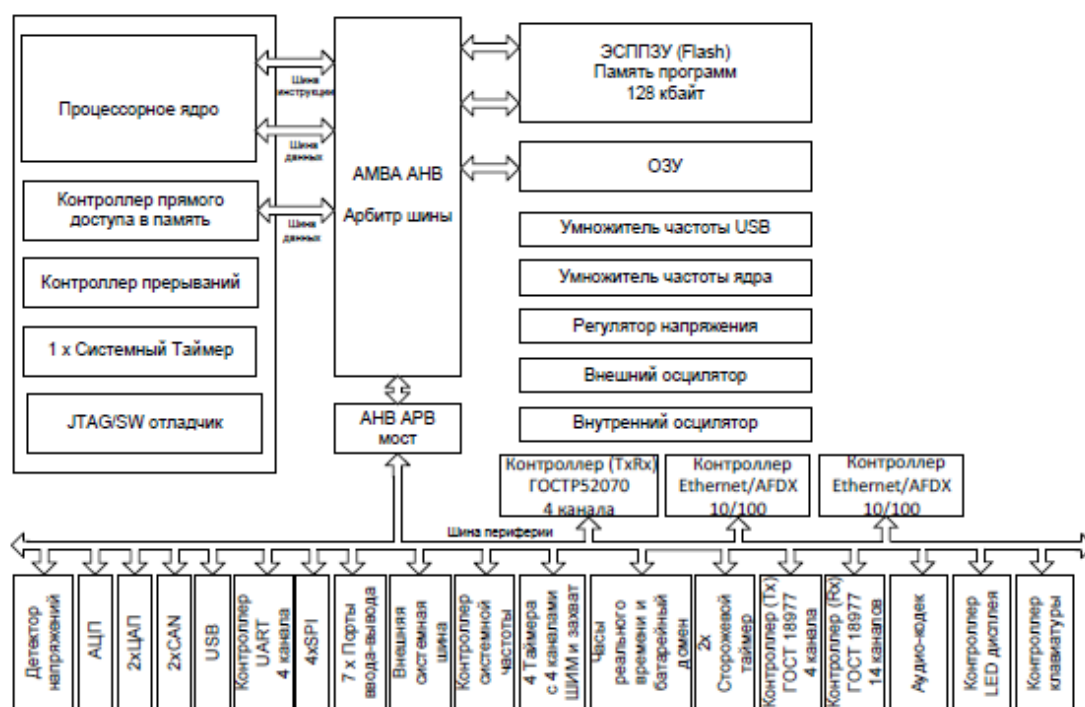


Рисунок 7 — Структура микроконтроллера 1986BE3T

Благодаря двум контроллерам Ethernet, 1986BE3T можно применять в

качестве шлюза между внешней сетью и сетью, организованной в составе оборудования. Кроме того, с помощью ИС 1986BE3T можно строить Ethernet-сети с архитектурой кольца. Такая топология крайне редко используется в компьютерных Ethernet-сетях, но находит применение в промышленных сетях автоматизированных систем управления технологическими процессами.

Преимущество кольцевой топологии в задачах управления технологическими процессами – минимизация кабельных линий. Каждое устройство такой сети выполняет функцию ретранслятора, что обеспечивает высокую стойкость к внешним помехам. Очевидно, что для построения подобных устройств наиболее оптимальна микросхема с двумя встроенными Ethernet-контроллерами.

Еще одна важная задача для ИС с двумя Ethernet-контроллерами – резервирование сети по числу каналов. В случае выхода из строя одного из каналов обмена происходит переключение на резервный канал. Такая возможность особенно актуальна в специальных задачах.

Другое преимущество контроллера 1986BE3T – наличие встроенных стыков USB и аудиокодека, которые должны быть реализованы в узле.

4.3 Микроконтроллер 1986BE3T

Основные характеристики микроконтроллера 1986BE3T [8]:

- 32-битное RISC ядро, тактовая частота до 80 МГц, производительность 0,8 DMIPS/МГц при нулевой задержке памяти;
- умножение 32х32 за три цикла.
- встроенная энергонезависимая память программ флеш-типа размером 128 Кбайт;
- встроенное ОЗУ размером 48 Кбайт;
- контроллер внешней системной шины с поддержкой микросхем сверхоперативного ЗУ, постоянные ЗУ, NAND Flash;
- внешнее питание от 3,0 до 3,6 В;
- встроенный регулятор напряжения на 1,8 В для питания ядра;
- встроенные схемы контроля питания;

- встроенный домен управления батарейным питанием;
- встроенные подстраиваемые RC генераторы 40 кГц и 8 МГц;
- внешние осцилляторы от 2 до 16 МГц, от 20 до 30 МГц, 32 кГц;
- встроенные умножители тактовой частоты PLL для ядра и для контроллера USB;
- батарейный домен с часами реального времени и регистрами аварийного сохранения;
- 12-разрядный АЦП (до 8-ми каналов) с амплитудой измеряемых сигналов от 0 до 3,6 В;
- двухканальный 12-разрядный ЦАП;
- 32-разрядный контроллер широтно-импульсной модуляции (ШИМ) (до 8 каналов);
- одноканальный АЦП с разрядностью выходных отчетов 16 бит;
- одноканальный ЦАП с разрядностью входных отчетов 16 бит;
- температурный датчик;
- контроллер прямого доступа в память с функциями передачи периферия-память, память-память;
- два контроллера CAN интерфейса;
- цифровой контроллер интерфейса по ГОСТ 18977-79;
- два цифровых контроллера интерфейса по ГОСТ Р 52070-2003;
- цифровой контроллер интерфейса Ethernet 10/100 и PHY Transceiver;
- контроллер USB интерфейса с функциями работы Device и Host;
- контроллеры интерфейсов 4xUART, 4xSPI;
- контроллер аудиокодека;
- до 144 пользовательских линий ввода/вывода.
- последовательные отладочные интерфейсы SWD и JTAG.

Контролер USB реализует функции контроллера функционального устройства (Device) и управляющего устройства (Host) в соответствии со спецификацией USB 2.0.

Контроллер USB поддерживает режимы работы FullSpeed (12 Мбит/с)

и LowSpeed (1,5 Мбит/с), контроль ошибок с помощью циклического избыточного кода (CRC), NRZi-код приема/передачи, управляющие, сплошные, изохронные передачи и передача по прерываниям. Также поддерживается конфигурирование контроллера USB-device от одной до четырех окончечных точек, автоматическая отправка пакетов SOF, вычисление оставшегося в кадре времени. Каждая окончечная точка контроллера USB-device имеет собственный буфер FIFO размером 64 байта. Контроллер USB-host поддерживает до 16 окончечных точек. USB-host имеет буфер FIFO размером 64 байта.

При включении питания в первую очередь должны быть заданы параметры тактового сигнала блока USB. Параметры задаются в блоке «Сигналы сброса и тактовой частоты». Источником тактового сигнала для блока USB может быть либо встроенный высокочастотный генератор (HSI) или внешний осциллятор (HSE). Блок USB функционирует на частоте 48 МГц. Требуемая частота может быть получена умножением частоты одного из двух генераторов до требуемого значения. Умножение выполняется встроенным блоком PLL-USB.

Блок умножения позволяет провести умножение входной тактовой частоты на коэффициент от 2 до 16, задаваемый в поле PLLUSBMUL регистра PLL_CONTROL. Входная частота блока умножителя должна быть в диапазоне от 2 до 16 МГц, а выходная частота должна составлять 48 МГц. При выходе блока умножителя тактовой частоты в расчетный режим вырабатывается сигнал PLLRDY. Блок включается с помощью сигнала PLLUSBON. Выходная частота используется как основная частота протокольной части USB интерфейса.

При работе контроллер интерфейса Ethernet (блок MAC) выполняет 2 основные функции:

- формирование пакета уровня звена данных протоколов Ethernet/IEEE802.3 и передача его на физический уровень;
- прием с физического уровня и разбор пакета уровня звена данных протоколов Ethernet/IEEE802.3.

Работа блока возможна в полно- и полудуплексном режимах. Переключение режима осуществляется битом HD_EN регистра G_CFG («1» – полудуплексный режим работы). Также в общее управление блоком входит разрешение обработки пакета PAUSE (бит PAUSE_EN регистра G_CFG), управление размером окна распознаваний коллизий (бит ColWnd регистра G_CFG). И режимом работы буферов приемника и передатчика (бит BUFF_MODE регистра G_CFG).

Режимы работы буферов линейный (BUFF_MODE=2'b00), автоматический (BUFF_MODE=2'01) и режим FIFO (BUFF_MODE=2'10) различаются способом обработки указателей границ пустых и полных областей буферов приемника и передатчика. В линейном режиме границы начала пустой области буфера приемника (head_R) и передатчика (tail_X) определяются записью в соответствующие регистры управления. В автоматическом режиме эти границы определяются автоматически по адресам последних чтения или записи в соответствующие области. В режиме FIFO границы определяются автоматически, при этом обращение к приемнику производится через адрес 0x0000, а передатчику через адрес 0x0004.

Для передачи пакетов необходима предварительная настройка блока передатчика MAC. Настройка параметров работы передатчика осуществляется посредством регистра управления передатчика X_CFG:

- разрешение работы передатчика (бит EN);
- порядок следования байт в буфере (бит BE);
- порядок следования бит в байте (бит MSB1st);
- выбор события при передаче, выводимого на вывод EVNT[1] (поле EVNT_MODE);
- управление дополнением пакета до минимальной длины PAD-ами (бит PAD_EN);
- управление дополнением пакета преамбулой (бит PRE_EN), SFD добавляется в любом случае;
- управление дополнением пакета, автоматически подсчитываемым

полем CRC (CRC_EN);

- управление интервалом между отправлением пакетов (бит IPG_EN);
- управление максимальным числом повторений (поле RtryCnt).

Далее необходимо записать пакет для передачи в буфер передатчика.

Пакет для передачи содержит 3 поля (все поля должны быть выровнены по границе слова буферного ОЗУ):

- поле управления передачей пакета;
- собственно данные пакета уровня звена данных;
- поле состояния передачи пакета.

Поле управления содержит количество байт пакета в буфере, которые должны быть переданы.

Поле состояния заполняется по завершении процедуры отправки пакета (успешной или нет) и содержит статусную информацию по отправке пакета: о наличии ошибок при его передаче, о количестве попыток передачи пакета и прочее.

Если выбран линейный режим работы буферов, то после помещения пакета для передачи необходимо записать в регистр управления tail_X границу пустой области (адрес, следующий за последним словом пакета).

По завершении передачи пакета блок MAC выставит один из флагов прерываний передатчика.

Во время отправки пакета или по его завершении выставляется событие, запрограммированное в поле EVNT_MODE регистра X_CFG.

Контроллер аудиокодека обладает следующими основными характеристиками:

- одноканальный АЦП с разрядностью выходных отчетов 16 бит;
- одноканальный ЦАП с разрядностью входных отчетов 16 бит;
- возможность выбора фильтра с конечной или бесконечной импульсной характеристикой в трактах АЦП и ЦАП. А так же возможность исключения этих фильтров из трактов;
- АЦП и ЦАП работают с фиксированным значением OSR равным 256.

Выбор частоты дискретизации осуществляется изменением входного синхросигнала;

- дифференциальный и недифференциальный аналоговый вход/выход;
- возможность микширования данных тракта АЦП и ЦАП;
- встроенный аналоговый сглаживающий фильтр;
- ручная регулировка усиления в тракте АЦП;
- IP блок совместим с APB интерфейсом для построения SOC;
- IP блок содержит независимые FIFO (16 отсчетов) для трактов АЦП и ЦАП;
- IP блок имеет независимые DMA каналы для тракта АЦП и ЦАП;
- IP блок формирует сигнал прерывания по внутренним событиям;
- имеет встроенный формирователь синхросигналов;
- ЦАП имеет защиту от короткого замыкания.

4.4 Другие микросхемы

Фирма «ПКК Миландр» производит и другие микросхемы [8], практически полностью перекрывающие потребности разрабатываемого узла.

Для программирования ПЛИС 5576XC3 и 5576XC4 выпускается микросхема последовательного ПЗУ 5576PC1Y, емкостью 4 Мбита, с напряжением питания от 3,0 до 3,6 В. Схема ее подключения представлена на рисунке 8.

Резисторы R1 и R2 имеют сопротивление 4,7 кОм, а конденсаторы C1 и C2 – емкость 0,1 мкФ. Элемент G3 – источник питания 3,3 В. JTAG – разъем для прошивки. Микросхема прошивается непосредственно из САПР Quartus II.

Для энергонезависимого хранения программ в узле может быть использована микросхема флеш-ПЗУ K1636PP2Y емкостью 2Мх8, напряжением питания от 3,0 до 3,6 В и временем выборки 65 нс.

Для обмена информацией с внешними узлами очень удобны микросхемы двухпортовых ОЗУ. Такое ОЗУ представляет собой общий массив памяти, к которому возможен доступ с двух сторон или портов. Каждый порт представляет собой стандартный интерфейс обычного ОЗУ – шину адресов,

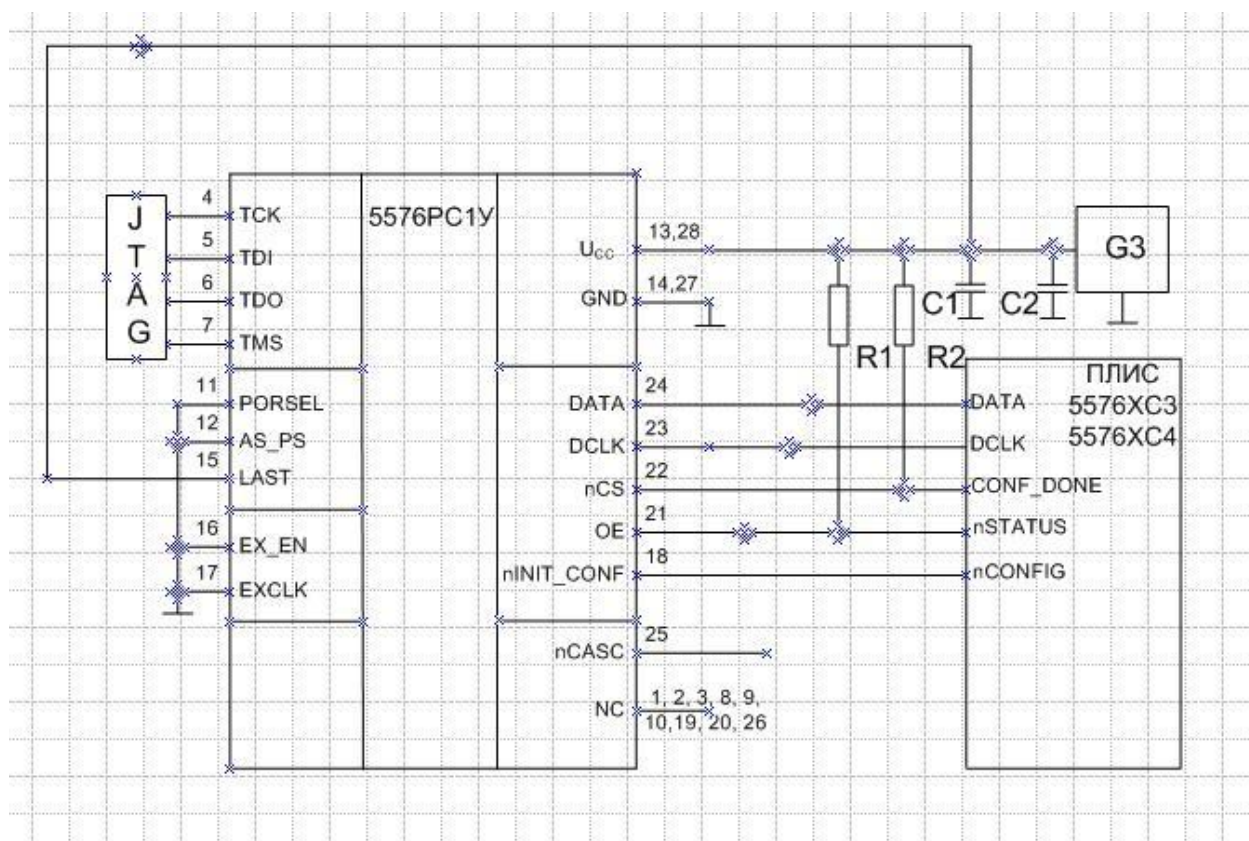


Рисунок 8 — Подключение ПЗУ 5576PC1У к ПЛИС серии 5576

входы записи, чтения, выборки кристалла и двунаправленную шину данных. Через два этих порта разные процессоры могут обращаться к разным ячейкам памяти одного и того же массива.

В этом случае отпадает необходимость использования обычного в таких случаях арбитража обращения к памяти, который может включать и приостановку работы одного процессора на время обращения другого процессора к его памяти. Фирма «ПКК Миландр» выпускает микросхемы двухпортового ОЗУ К1645РК1У с организацией 32Кх8, напряжением питания 5 В и временем выборки 50 нс.

И только воронежское ОАО «КТЦ «Электроника»» выпускает используемую в узле ПЛИС 5576XC4Т емкостью 200000 вентиляей, представляющую собой функциональный аналог ПЛИС типа EPF10K200 фирмы Altera. Напряжение питания – 3,3 В.

Особенности проектирования на основе ПЛИС 5576XC4Т состоят в следующем. Для создания проектов может быть использована САПР фирмы Altera – MAX+PLUS II или Quartus II.

Проектирование подразумевает ввод проекта, функциональное моделирование, автоматическое размещение элементов и трассировку межсоединений, создание файлов для конфигурирования ПЛИС.

Для компиляции проекта в САПР MAX+PLUS II или Quartus II во вкладке Assign выбрать пункт Device, затем Device Family: FLEX10KE и Device: EPF10K200SRC240–3, далее вести проектирование согласно поставленной задаче, руководствуясь описанием выбранного ПО и таблицами соответствия выводов. Номера выводов ПЛИС 5576XC4T не соответствуют номерам выводов ПЛИС EPF10K200SRC240–3. Для задания определения номеров выводов ПЛИС 5576XC4T существует специальная таблица соответствия. Если, например, необходимо использовать вывод 69 ПЛИС 5576XC4T для конкретного внешнего сигнала, то согласно таблице соответствия, в настройках САПР необходимо сделать назначение этому сигналу – вывод 6.

Так же возможна автоматическая генерация файла с размещением выводов ПЛИС 5576XC4T в корпусе аналога для САПР Quartus II при помощи специально разработанной программы «create_quartus_assignments». В этом случае, пользователь в качестве входных данных указывает только имя и номер вывода в корпусе 4244.256–3, а программа создает файл с размещением для нужного корпуса аналога. Описание программы приведено в данной инструкции. Кроме этого, ПЛИС 5576XC4T имеют расширенную функциональность элементов ввода-вывода (режимы Pull-Up, Pull-Down, Bus-Hold), а также дополнительные режимы работы (SCRUBBING, VERIFICATION, FREQ_DIV), которые отсутствуют в аналоге. Эти режимы реализуются при помощи специального программного обеспечения «5576XC4T Configuration Edit Tools». Описание режимов и программного обеспечения для их реализации приведено в инструкции к данному ПО.

Перед конфигурированием микросхем необходимо обеспечить следующие условия:

— все выводы 1GND, 2GND должны быть подключены к

отрицательному выводу источника питания (общему);

- все выводы 1VCC должны быть подключены к положительному выводу источника питания 1,8 В;

- все выводы 2VCC должны быть подключены к положительному выводу источника питания 3,3 В;

- незадействованные в проекте пользовательские входы должны быть подключены к отрицательному или положительному 3,3 В выводу источника питания напрямую или через внешние резисторы. Также возможно использование режимов Pull-Up или Pull-Down;

- для пользовательских входов или входов/выходов, используемых в проекте в качестве входов, на которых возможно появление Z-состояния во время работы, должен быть включен режим Pull-Up, Pull-Down или Bus-Hold. Также возможно использование внешних резисторов;

- выводы RES, RES_MODE, TEST_MODE всегда должны быть подключены к положительному выводу источника питания 3,3 В;

- подключение служебных выводов следует согласовать с выбором режима конфигурирования в соответствии с данной инструкцией.

Для конфигурирования ПЛИС по порту Download существуют следующие конфигурационные схемы:

- PS (Passive Serial – пассивное последовательное) конфигурирование;
- AS (Active Serial – активное последовательное) конфигурирование;
- PS (Passive Parallel Synchronous – пассивное параллельное синхронное) конфигурирование;
- PPA (Passive Parallel Asynchronous – пассивное параллельное асинхронное) конфигурирование.

Особенности ПЛИС 5576XC4T:

- блоки встроенной памяти для реализации мегафункций;
- логические блоки для основных логических функций;
- напряжение питания ядра ПЛИС UCC1 равно 1,8 В;
- напряжение питания периферии ПЛИС UCC2 = 3,3 В;

— встроенный блок тестирования (Joint Test Action Group (JTAG)) с использованием схемы периферийного сканирования (СПС), совместимый с IEEE Std. 1149.1;

— встроенная система реконфигурации (ICR), обеспечивающая многократное перепрограммирование посредством специальной микросхемы памяти 5576PC1У, загрузочных кабелей типа ByteBlaster, микропроцессора;

— программируемый режим верификации конфигурационной памяти без выхода из рабочего состояния (VERIFICATION);

— программируемый режим циклической перезаписи конфигурационной памяти (SCRUBBING).

Особенности элементов ввода/вывода:

— индивидуальный сигнал разрешения третьего состояния для каждого вывода;

— возможность использования опции «открытый сток» для каждого вывода;

— снижение помех при переключениях за счет программируемой опции slew-rate control;

— программируемые опции «bus-hold», «pull-up», « pull-down »;

— программируемый триггер.

Особенности внутренних межсоединений ПЛИС:

— внутренние быстрые глобальные межсоединения (Fast Track Interconnect);

— специализированная цепь переноса, обеспечивающая поддержку арифметических функций, таких как быстродействующие сумматоры, счетчики и компараторы (используется автоматически программой);

— специализированная цепь каскадирования, обеспечивающая поддержку высокоскоростных функций с высоким коэффициентом разветвления на входе (используется автоматически программой);

— до шести глобальных тактовых сигналов и четырех глобальных сигналов сброса.

5 Разработка узла

5.1 Разработка структурной схемы узла

Разработан модуль (узел) для предварительной обработки передаваемой или принимаемой информации. При этом всякая принимаемая информация будет преобразовываться в информационный кадр определенного размера. Передаваемая информация, наоборот – из унифицированного кадра, помещаемого в определенную область ОЗУ узла, будет передаваться по соответствующему определенному стыку (Ethernet, USB и прочие).

Структура разработанного узла представлена на рисунке 9.

Основу узла составляет микроконтроллер 1986BE3T, содержащий все стыки, описанные в задании. Для хранения программы его работы использовано флеш-ПЗУ, для хранения промежуточных результатов работы программы микроконтроллера использовано внешнее ОЗУ. Для связи с внешними узлами использовано двухпортовое ОЗУ.

Разработанный узел предназначен для работы в защищенной АСОД, поэтому в данном узле предусмотрено УКЗИ для криптографической защиты передаваемой и принимаемой информации. УКЗИ осуществляет шифрование или расшифрование информации по алгоритму ГОСТ 28147-89 в режиме гаммирования.

УКЗИ выполнено как еще одно периферийное устройство для процессора узла. Для работы УКЗИ необходима ключевая информация (КИ), которая принимается в УКЗИ по специальному каналу из узла, осуществляющего ее рассылку. Для пересылаемой по разным каналам информации необходима разная КИ. Вся она хранится непосредственно в УКЗИ, а выбирается для конкретного пакета по команде от процессора. УКЗИ работает по команде от процессора. УКЗИ имеет два регистра: входного слова, выходного слова, режима работы (выбирается КИ и определяется тип преобразования – зашифрование или расшифрование).

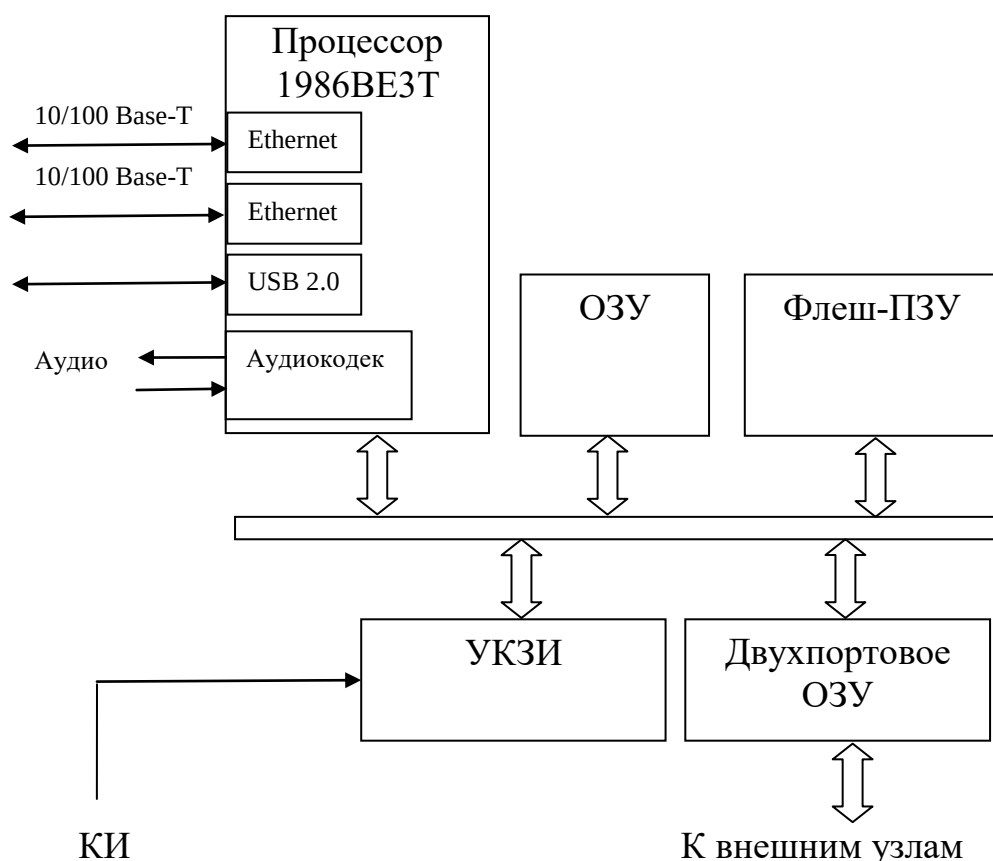


Рисунок 9 — Структура узла

Передаваемый через узел по стыкам Ethernet и USB пакет информации поступает сначала в двухпортовое ОЗУ. Процессор по очереди считывает слова пакета, каждое считанное слово помещает в УКЗИ, затем извлекает оттуда зашифрованное слово, и отправляет его в требуемый стык.

Принимаемый по стыкам Ethernet и USB пакет информации поступает в процессор и сначала помещается в его внешнее ОЗУ. Затем каждое принятое слово процессор пересылает в УКЗИ, оно там расшифровывается, а затем процессор помещает его в двухпортовое ОЗУ для последующей передачи внешним узлам.

Несколько иначе работает узел в режиме передачи и приема речи через аудиокодек. В этом случае в отличие от стыков Ethernet и USB принимаемая речь шифруется, а передаваемая (на аудиовыход) – расшифровывается. Зашифрованная речь может через двухпортовое ОЗУ передаваться другим узлам, а может отправляться сразу в стык Ethernet для передачи в канал.

5.2 Разработка функциональной схемы узла

Узел полностью построен на отечественной элементной базе, причем практически каждому блоку структуры соответствует одна микросхема. Основу узла составляет микроконтроллер 1986BE3T. УКЗИ создана на ПЛИС 5576XC4T.

Для приема КИ из узла, ее распределяющего, выделена одна цепь, по которой передача осуществляется стартстопным способом. Это позволяет сократить до предела число цепей, по которым передается КИ, что особенно важно для узла, распределяющего КИ, который будет рассылать КИ в несколько УКЗИ по индивидуальным связям.

Особенностью этого микроконтроллера является возможность подключения цепей стыков Ethernet, USB и аудиокодека непосредственно к соответствующим разъемам без каких-либо согласующих цепей. Микросхемы памяти также подключаются практически напрямую к процессору, если не считать того, что для них необходима отдельная схема адресного дешифратора, которая реализована на ПЛИС.

Особенностью двухпортового ОЗУ является возможность обмена данными между двумя процессорами, к которым подключено это ОЗУ, без остановки их работы. Его использование позволяет «вписать» периферийные устройства в область адресов главного процессора, и таким образом «отвязать» главную программу от особенностей архитектуры конкретного процессора.

Программы работы узла хранятся во флеш-ПЗУ, куда они записываются на заводе-изготовителе. Для записи программы используется отладочный порт JTAG процессора, к которому подключается эмулятор, позволяющий отлаживать программы работы непосредственно на самом устройстве. Режим записи флеш-ПЗУ включается только в процессе записи микросхемы на заводе-изготовителе. Для этого подается нужный уровень на специальный вывод аппаратной блокировки операций записи и стирания.

В двухпортовом ОЗУ каждый порт в любой момент времени независимо от другого порта может проводить чтение по любому адресу. Процесс записи в

различные адреса также не вызывает помехи со стороны противоположного порта.

Если запись одного порта начнется раньше, чем по тому же адресу другой порт начнет чтение, то на информационные выходы читающего порта поступит та информация, которая записывается другим портом. Считывание будет достоверным, если длительность обращения читающего порта будет достаточной для того, чтобы обновленная информация по этому адресу зафиксировалась его выходными защелками.

В случае если оба порта намереваются провести запись по одному и тому же адресу, то запись пройдет успешно у того порта, который первым начнет эту процедуру. «Запоздавшему» порту запись будет заблокирована. Для того, чтобы процессоры, управляющие портами, могли «согласовать» свои действия, и не возникло конфликтной ситуации, в микросхемах двухпортовых ОЗУ имеются такие функции, как «прерывание», логика занятости и «семафоры».

Прерывания работают следующим образом. Если пользователь выбирает применение функции прерывания, то часть основной памяти будет использоваться, как «почтовый ящик или центр сообщений». Флаг прерывания для левого порта ($nINTL$) выставляется, когда правый порт записывает (8 бит) в основную память по адресу 7FFE. Запись осуществляется в соответствии с таблицей истинности, если $nCER = R/nWR = L$. Когда левый порт при $nCEL = nOEL = L$ обращается к основной памяти по адресу 7FFE, то флаг прерывания для левого порта снимается. Логический уровень на входе R/nWL левого порта при этом может быть любым.

Аналогично для правого порта выставляется флаг прерывания ($nINTR$), когда левый порт проводит запись в память по адресу 7FFF и для снятия (очистки) этого флага прерывания правый порт должен обратиться (при чтении или записи) к памяти по адресу 7FFF.

Смысловое содержание сообщения (8 бит) по адресам 7FFE или 7FFF предопределяется пользователем сразу же, как только условлено, что это место

ОЗУ отведено для сообщений. Если функция прерываний не применяется, то память по адресам 7FFE и 7FFF не используются как «почтовый ящик», а используются как часть общей памяти.

Логика занятости обеспечивает аппаратную индикацию при обращении обоих портов одновременно в одно и то же место ОЗУ. Это позволяет одному из двух портов завершить операцию и сигнализировать другому порту, что ячейка ОЗУ занята. Логическое состояние на выходе nBUSY в это время может быть использовано для остановки доступа к ячейке памяти одного порта до завершения аналогичной операции другим портом. Если операция записи предпринимается со стороны порта, который получил сигнал занятости, то внутри микросхемы сигнал записи этого порта блокируется для предотвращения этого процесса.

Использование логики занятости не является обязательным для любых применений. В некоторых случаях это может быть удобно: по логике ИЛИ можно объединить выходы занятости вместе и использовать индикацию занятости как источник прерывания для обозначения ошибочного события или нелогичной операции.

Если блокировка функции записи логикой занятости нежелательна, то работа логики занятости может быть запрещена путем установки микросхемы в режим «Ведомый» через вывод M/nS. В режиме «Ведомого» сигнал nBUSY функционирует исключительно как вход запрещения записи. Нормальная работа без запрета записи обеспечивается путем подачи на вход nBUSY высокого логического уровня. Если требуется избежать непреднамеренной записи в этот порт, то на вывод nBUSY должен быть подан низкий логический уровень.

Выход занятости в ОЗУ 1645PK1У в режиме «Мастер» не требует резистора подтяжки для работы микросхемы. Если для увеличения емкости используется несколько микросхем одновременно, то для индикации результирующего сигнала занятости всего массива ОЗУ требуется использование внешнего элемента «И».

Для организации семафоров выделяют дополнительный восьмиадресный массив, предназначенный для двоичных семафорных флагов. Эти флаги позволяют любому из двух процессоров на левой или правой стороне двухпортового ОЗУ претендовать на привилегии над другим процессором в действиях, определяемых разработчиком системного программного обеспечения. Как пример, семафорные флаги могут быть использованы одним процессором для запрещения другому процессору доступа к участку памяти данного двухпортового ОЗУ или к другим общим ресурсам.

Особенностью двухпортового ОЗУ является быстрый доступ и полная независимость в действиях портов друг от друга. Это означает, что действия левого порта не влияют на скорость доступа правого порта. Оба порта идентичны по функциональности стандартному статическому ОЗУ комплементарной логики и могут читать, или записывать одновременно. При этом конфликт между портами возникнет только при попытке портов записать информацию в одну и ту же ячейку основной памяти.

Семафоры защищают от такой неоднозначной ситуации и могут использоваться системной программой таким образом, чтобы избежать обращения портов в одинаковые адреса основной памяти двухпортового ОЗУ.

Системы, которые наилучшим способом используют двухпортовое ОЗУ, содержат несколько процессоров или контроллеров и, обычно, являются высокоскоростными системами, которые контролируются исполняемой программой. Программное взаимодействие между процессорами обеспечивает максимальную гибкость в системе путем распределения общих ресурсов в различных конфигурациях. Флаги семафоров не используются для аппаратного управления какими-либо ресурсами, тем самым раскрывая перед разработчиками системы полную гибкость в выборе архитектуры системы.

Преимущество использования семафоров по сравнению с общими методами аппаратного арбитража заключается в том, что не возникают циклы ожидания для каждого процессора. Это может обеспечить двухпортовому ОЗУ основное преимущество применения в высокоскоростных системах.

ЗАКЛЮЧЕНИЕ

В данной ВКР предложен вариант построения унифицированного узла для автоматизированной обработки защищенной информации, отличающийся от аналогичных узлов тем, что для обеспечения переносимости программного обеспечения, обрабатывающего закрытую информацию, при построении узла решено исходить из технологии УТАОИ и из того, что взаимодействие с периферийными устройствами должно быть выделено в отдельные подпрограммы и вынесено в отдельные контроллеры. Предложен вариант построения переносимой программы. Определены рекомендации по структуризации аппаратных узлов с программной поддержкой. Осуществлен выбор отечественной элементной базы для разработанного узла.

Таким образом, цель дипломного проекта достигнута, требования технического задания на выпускную квалификационную работу выполнены в полном объеме.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Ершов А. П., Шура-Бура М. Р. Становление программирования в СССР (переход ко второму поколению языков и машин) – Новосибирск, 1976. – 42 с.

2 Наука и образование. Электронное научно-техническое издание. [Электронный ресурс]: В.П. Корячко, С.В. Скворцов, И.А. Телков, Математическая модель мобильных вычислений. Режим доступа: <http://technomag.edu.ru/doc/54577.html>, свободный, дата доступа: 10.05.2016.

3 Криницкий Н.А., Миронов Г.А., Фролов Г.Д. Программирование и алгоритмические языки. Под ред. А.А. Дородницына. – М.: Наука, 1975. – 496 с.

4 Software Portability [Электронный ресурс]: J.D. Mooney. Bringing Portability to the Software Process. Режим доступа: http://www.cs.wvu.edu/~jdm/research/portability/reports/TR_97-1.pdf, свободный, дата доступа: 17.05.2016.

5 Герасименко В.А. Защита информации в автоматизированных системах обработки данных. В 2-х кн. – М.: Энергоатомиздат, 1994. – 176 с.

6 Герасименко В.А., Гудинович Г.И. Математические основы структурного программирования // Зарубежная радиоэлектроника, 1977, №12. – 268 с.

7 А. Андреев, С. Гусев, С. Шумилин. Ethernet-решения для специальной аппаратуры: отечественная элементная база // Электроника: наука, технология, бизнес, 2014, №2. – 241 с.

8 ПКК «Миландр» [Электронный ресурс]: Документация. Режим доступа: <http://milandr.ru/index.php?page=dokumentaciya>, свободный, дата доступа: 13.03.2016.

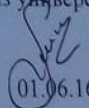
Ректору Пензенского
государственного университета
А. Д. Гулякову
студентки 5 курса ФПИТО
очной формы обучения
Погодиной Анны Евгеньевны

заявление.

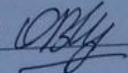
Я, Погодина Анна Евгеньевна, студент 5 курса очной формы обучения специальности 10.05.03 – «Информационная безопасность автоматизированных систем», прошу провести проверку с использованием системы «Антиплагиат» дипломного проекта на тему «Унифицированный узел для автоматизированной обработки защищенной информации», выполненной мной самостоятельно, на содержание элементов плагиата.

Все прямые заимствования из печатных и электронных источников, а также из защищенных ранее выпускных квалификационных работ, кандидатских и докторских диссертаций имеют соответствующие ссылки.

Я ознакомлен с действующим Стандартом университета «Выпускная квалификационная работа обучающихся по образовательным программам высшего образования – программам бакалавриата, программам специалитета и программам магистратуры», согласно которому обнаружение плагиата является основанием для недопуска ВКР к защите и отчисления из университета.

 /А.Е. Погодина/
01.06.16

ВКР представлена на проверку

Руководитель ВКР  /О.В. Кулагин/

Протокол
проверки на оригинальность в системе «Антиплагиат.ВУЗ» выпускной
квалификационной работы на тему «Унифицированный узел для
автоматизированной обработки защищенной информации» студента 5 курса
А.Е. Погодиной специальности 10.05.03 – «Информационная безопасность
автоматизированных систем».

Руководитель ВКР к.т.н. инженер I категории АО «ПНИЭИ» Кулагин
Олег Витальевич.

03.06.2016

Вывод отчета на печать - Антиплагиат

Уважаемый пользователь! Обращаем ваше внимание, что система «Антиплагиат» отвечает на вопрос, является ли тот или иной фрагмент текста
заимствованным или нет. Ответ на вопрос, является ли заимствованный фрагмент текстом плагиатом, а не законной цитатой, система оставляет на
ваше усмотрение.

Отчет о проверке № 1

ФИО: Иванов Алексей Петрович
дата загрузки: 03.06.2016 06:26:40
пользователь: ap_ivanov@mail.ru / ID: 1267385
отчет предоставлен сервисом «антиплагиат»
на сайте <http://www.antiplagiat.ru>

Информация о документе

№ документа: 39
Имя исходного файла: ВКР_Погодина_АЕ_111111.pdf
Размер текста: 972 кб
Тип документа: Не указано
Символов в тексте: 90562
Слов в тексте: 16555
Число предложений: 408

Информация об отчете

Дата: Отчет от 03.06.2016 06:26:40 – Последний готовый отчет
Комментарий: не указано
Оценка оригинальности: 96,4%
Заимствования: 3,6%
Цитирование: 0%



Оригинальность: 96,4%
Заимствования: 3,6%
Цитирование: 0%

08.06.2016 г.

Секретарь ГЭК

О.В. Кулагин
А.П. Иванов

О.В. Кулагин

А.П. Иванов

ОТЗЫВ
на дипломную работу студентки
Пензенского государственного университета
Погодиной Анны Евгеньевны

Тема работы: «Унифицированный узел для автоматизированной обработки защищенной информации».

Целью настоящей работы является разработка унифицированного узла для программно-аппаратной обработки информации в составе защищенной автоматизированной системы обработки данных (АСОД). При этом в процессе данной разработки была успешно решена задача увязки между собой двух взаимоисключающих тенденций, особенно остро проявляющихся при проектировании защищенных АСОД. Первая тенденция заключается в необходимости максимального использования существующего задела, особенно в программной части, для сокращения работ контролирующей организации, сопровождающей разработку такой АСОД. Вторая заключается в быстрой смене элементной базы, в первую очередь процессоров, которая сильно затрудняет использование программного задела.

Ценность данной работы заключается в успешном решении указанной выше сложной задачи, позволяющей повторно использовать ранее отлаженное программное обеспечение защищенных АСОД. Это решение основано на предложенной профессором В.А. Герасименко технологии УТАОИ и сводится к рекомендации выделения из основной программы всех обращений к периферийным устройствам в отдельные подпрограммы и их последующий перенос на отдельные микроконтроллеры. В результате основная программа становится переносимой на другие процессоры и может быть использована без переделок в новых разработках.

В пояснительной записке представлены: концепция защиты информации на основе УТАОИ, подходы к структуризации программной и аппаратной части АСОД, выбор элементной базы и описание разработанного узла.

Задание на дипломную работу выполнено полностью и на высоком техническом уровне. Считаю, что дипломная работа может быть допущена к защите в ГАК и заслуживает оценки «отлично» (15 баллов), а дипломница Погодина Анна Евгеньевна – присвоения квалификации специалиста по защите информации.

Руководитель дипломной работы:
к.т.н., инженер I категории АО «ПНИИ»



Кулагин О.В.

РЕЦЕНЗИЯ
на дипломную работу студентки
Пензенского государственного университета
Погодиной Анны Евгеньевны

Тема работы: Унифицированный узел для автоматизированной обработки защищенной информации.

Представленная на рецензию дипломная работа содержит пояснительную записку в объеме 61 страницы и графическую часть на 6 листах. В техническом задании на дипломную работу требовалось разработать унифицированный узел для программно-аппаратной обработки информации в составе защищенной автоматизированной системы обработки данных (АСОД). Определены требования к узлу и перечень вопросов, которые должны быть рассмотрены.

Ценностью работы является увязывание вопроса уменьшения трудоемкости разработки защищенной АСОД с вопросом переносимости аппаратных и программных частей, т.е. с вопросом максимального использования существующего задела в новых разработках.

В пояснительной записке изложены принципы структуризации программной и аппаратной частей АСОД, представлена предложенная профессором В.А. Герасименко унифицированная технология автоматизированной обработки информации. Также представлено описание разработанного узла.

Требования технического задания выполнены полностью. В пояснительной записке материалы работы изложены полностью и на высоком техническом уровне.

В качестве недостатка работы следует отметить тот факт, что в работе никак не оценивается достаточность объема двухпортового ОЗУ в 32 Кб для обмена информацией разработанного узла (имеющего несколько стыков для обмена информацией с внешней средой) с остальной АСОД.

Однако, ценность работы от этого не снижается. Считаю, поэтому, что дипломная работа Погодиной Анны Евгеньевны может быть допущена к защите в ГАК и заслуживает оценки "ОТЛИЧНО" (9 баллов), а выпускница – присвоения квалификации специалиста по защите информации.

Рецензент:

Начальник НИО-102



Сомкин С.А.