

ВЛАДИМИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ АЛЕКСАНДРА ГРИГОРЬЕВИЧА
И НИКОЛАЯ ГРИГОРЬЕВИЧА СТОЛЕТОВЫХ

На правах рукописи

ТАХААН ОСАМА

**РАЗРАБОТКА КОРРЕКТИРУЮЩИХ КОДОВ ДЛЯ
ИНФОРМАЦИОННОЙ ЗАЩИТЫ ТЕЛЕКОММУНИКАЦИЙ
КОМПЬЮТЕРНЫХ СЕТЕЙ**

Специальность:

05.12.13 – « Системы, сети и устройства телекоммуникаций »

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
д.т.н., профессор
Галкин А.П.

Владимир - 2012

СОДЕРЖАНИЕ

СПИСОК ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ.....	5
ВВЕДЕНИЕ.....	7
ГЛАВА 1. ЗАЩИТА ТЕЛЕКОММУНИКАЦИОННЫХ КОРПОРАТИВНЫХ КОМПЬЮТЕРНЫХ СЕТЕЙ.....	21
1.1. Основные понятия и допущения.....	21
1.2. Отказоустойчивые функциональные ядра телекоммуникационных систем	25
1.2.1. Анализ отказоустойчивых ЭВМ	25
1.2.2. Анализ построения российских отказоустойчивых ЭВМ.....	30
1.2.3. Построение отказоустойчивого ядра ТККС.....	34
1.3. Анализ резервирования на основе линейных кодов.....	37
1.3.1. Обоснование требований к методам кодирования информации функционального ядра ТККС.....	37
1.3.2. Выбор метода кодирования, обеспечивающего минимальную сложность декодирующего устройства.....	40
1.3.3. Исследование коррекции ошибок заданной кратности.....	42
1.3.4. Проблема использования корректирующих кодов для обеспечения отказоустойчивости функционального ядра.....	43
1.4. Финансовая и информационная безопасность и риски при проектировании.....	45
Выводы к главе 1.....	51
ГЛАВА 2. ОБНАРУЖЕНИЕ И КОРРЕКЦИЯ ПРИ ФУНКЦИОНАЛЬНО- КОДОВОЙ ЗАЩИТЕ ЯДРА ТККС.....	54
2.1. Поэлементная и комплексная информационная защита ТККС.....	54
2.2. Разработка модифицированных итеративных кодов повышенной обнаруживающей и корректирующей способности.....	61
2.3. Обоснования рациональной методики построения итеративного кода для обнаружения и коррекции ошибок в устройствах хранения	

информации функционального ядра ТККС.....	65
2.3.1. Оценка аппаратных затрат на реализацию предлагаемого метода кодирования.....	65
2.3.2. Достоверность функционирования отказоустойчивого запоминающего устройства.....	71
2.3.3. Обоснование выбора метода обнаружения и коррекции ошибок в устройствах хранения и передачи информации.....	71
2.3.4. Сравнительная оценка аппаратных затрат при реализации предлагаемых методов кодирования информации.....	75
2.3.5. Сравнительная оценка достоверности функционирования при реализации предлагаемых методов кодирования информации.....	77
2.3.6. Обоснование методик кодирования информации при увеличении числа информационных разрядов.....	77
2.3.7. Сравнительная оценка предлагаемого подхода с существующими методами.....	78
2.4. Рекомендации для технической реализации предлагаемых методик кодирования.....	80
2.5. Разработка методики алгоритмизации декодирования.....	81
2.6. Модель функционально-кодовой защиты устройства памяти.....	84
2.7. Анализ угроз информационной безопасности и основные мероприятия по их предотвращению.....	86
Выводы к главе 2.....	90
ГЛАВА 3. МЕТОДИКИ ЗАЩИТЫ ПРОЦЕССОРА И ЭЛЕМЕНТОВ КИТС.....	91
3.1. Выбор комплекса защиты информации для корпоративных информационно-телекоммуникационных сетей.....	91
3.2. Обеспечение отказоустойчивости сумматора на основе корректирующих линейных кодов.....	101

3.3. Разработка подхода обнаружения и коррекции ошибок арифметических операций функционального ядра КСОН.....	103
3.4. Разработка функционально-кодовой защиты процессора при выполнении логических операций.....	106
3.4.1. Разработка способа коррекции ошибок при выполнении операции сложения по mod 2.....	106
3.4.2. Разработка способа коррекции ошибок при выполнении операции сдвига.....	107
3.4.3. Разработка способа коррекции ошибок при выполнении логической операции ИЛИ.....	109
3.4.4. Разработка способа коррекции ошибок при выполнении логической операции И.....	110
3.4.5. Разработка методики коррекции ошибок при выполнении логической операции НЕ.....	112
3.5. Разработка функциональной схемы отказоустойчивого процессора повышенной достоверности функционирования.....	113
Выводы к главе 3.....	142
ЗАКЛЮЧЕНИЕ.....	143
СПИСОК ЛИТЕРАТУРЫ.....	145
ПРИЛОЖЕНИЕ 1.....	155
ПРИЛОЖЕНИЕ 2.....	161
ПРИЛОЖЕНИЕ 3.....	168

СПИСОК ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ

- АБС** - автоматизированная биллинговая система
- АСУ ТП** - автоматическая система управления технологическими процессами
- БВ** - блок восстановления
- БИС** - большая интегральная схема
- БУ** - блок управления
- БЧХ** - коды Боуза-Чаудхури-Хоквингема
- ГК** - групповые контроллеры
- ДК** - дисковой контроллер
- ЗИНВ** - защита информации от непреднамеренного воздействия
- ЗНСД** - защита от несанкционированного доступа
- ЗУ** - запоминающее устройство
- ИБ** - информационная безопасность
- ИМС** - интегральная микросхема
- ИП** - интерфейсный процессор
- ИС** - информационные системы
- ИТ** - информационные технологии
- КИТС** - корпоративная информационная телекоммуникационная сеть
- КК** - корректирующие коды
- ККС** - корпоративные КС
- КОГ—ОД Г** - коды с коррекцией ошибок в одной группе и обнаружением в двух группах
- КО—ОД** - коды с коррекцией одиночной и обнаружением двойной ошибки
- КО—ОД—ООГ** - коды с дополнительной способностью обнаружения кратной ошибки в одной группе битов
- КС** - компьютерные сети
- КСОН** - КС особого назначения
- КТ—ОО** - код с коррекцией тройных и обнаружением одиночных ошибок
- ЛК** - локальные контроллеры

НСД - несанкционированный доступ

ОЗУ - оперативное ЗУ

ПД - процессор данных

ПД - процессор данных

ПЗУ - постоянное ЗУ

ПК - персональный компьютер

ПЛИС - программируемая логическая интегральная схема

ПО - программное обеспечение

РабС - рабочие станции

РОш - регистр ошибок

РП - регистр повторения

РС - коды Рида-Соломона

СБИС - сверхбольшая интегральная схема

СЗ - средства защиты

СС - схема сравнения

ТЗСК - триггер захвата от схем контроля

ТИД - триггер исходных данных

ТИОш - триггер игнорирования ошибок

ТККС - телекоммуникационные ККС

ТЭЗ - Типовой элемент замены

УГ - угроз

УУП - устройство управления памятью

Ц - цель

ЦП - центральный процессор

ШИУ - шинное интерфейсное устройство

ЭВМ - электронная вычислительная машина

ВВЕДЕНИЕ

Аналогично мировой экономике современная экономика Сирии опирается на новейшие информационные и телекоммуникационные технологии.

В настоящее время в стране широко используются компьютерные сети, которые привели к бурному распространению глобальных информационных сетей, открывающих принципиально новые возможности информационного обмена.

В то же время, в информационном пространстве Сирии потенциально существует угроза использования различных приемов создания мешающих воздействий (кибертерроризма).

Кибернетический терроризмом (кибертерроризм, электронный терроризм) преследует основную цель, направленную на несанкционированную модификацию, блокирование или разрушение данных, нарушение заданных режимов функционирования информационно-технических систем и их отдельных элементов, модификацию или разрушение программ, внедрение вредоносных программ.

При этом преднамеренно или неумышленно создается опасность для жизни или здоровья людей или наступления других тяжелых последствий, преследуются цели получения преимуществ при решении политических, экономических или социальных проблем. Кибертерроризм является одной из опасных преднамеренных угроз государственной и общественной безопасности и всей страны и отдельных корпоративных сетей.

Одним из перспективных направлений обеспечения работоспособности компьютерных сетей в экстремальных условиях, является разработка адаптивных отказоустойчивых систем, обеспечивающих автоматическое обнаружение, локализацию и исправление возникающих ошибок.

В свою очередь, в данной аппаратуре наиболее важное место занимает компьютер, предназначенный для управления и координации работы различных функциональных узлов, устройств, получения и обработки информации, систем контроля технического состояния системы и решения широкого круга других специфических задач поэтому, с точки зрения надежности и достоверности функционирования, особую актуальность приобретает вопрос обеспечения широкого использования в них автоматизированных систем управления.

Объектом исследования работы являются функциональные узлы компьютеров банковских электронных сетей, систем управления движением транспортных средств, правительственных систем связи, элементов технических систем безопасности Сирии.

Ответственность задач, возлагаемых на данную аппаратуру, определяет целый ряд специфических проблем, связанных с организацией обслуживания и обеспечением высокой эффективности рассматриваемой техники.

Данные системы, как правило, относятся к системам, восстанавливаемым вне процесса применения, важным показателем надёжности которых является **вероятность безотказной работы**.

Так как выдача ошибочной информации рассматриваемыми системами может привести к значительному ущербу, а в ряде случаев и к катастрофическим последствиям, то для систем такого рода в качестве основного показателя целесообразно использовать **достоверность функционирования**.

Под достоверностью функционирования устройства будем понимать свойство вычислительного устройства, характеризующее способность средств контроля признать результат работы устройства правильным или ошибочным при наличии пропуска ошибок или выдаче ложных сигналов ошибок средствами контроля.

Рассматриваемые системы относятся к объектам, работающим в реальном масштабе времени. Поток информации, поступающий в эти устройства, носит случайный характер, и устройство в любой момент времени должно быть готово к её обработке. Потеря текущей информации не может быть восполнена никакими иными способами. Переключение резерва в таких устройствах практически исключается.

Поэтому важной характеристикой рассматриваемых систем является их **быстродействие**.

Таким образом, при построении отказоустойчивых компьютерных сетей (КС) возникает необходимость обеспечения высоких значений вероятности безотказной работы, достоверности функционирования и быстродействия рассматриваемой аппаратуры.

Исходя из предъявляемых требований и условий применения КС, особую важность приобретает вопрос **выбора средств обеспечения отказоустойчивости** (средств обнаружения, локализации и коррекции возникающих ошибок).

Состояние исследуемого вопроса.

Структурные методы резервирования дискретных вычислительных устройств в Сирии специально не изучались, но они известны из мирового опыта и подробно рассмотрены в работах российских ученых С.М. Доманицкого, В.Е. Обухова, В.В. Сапожникова, Б.М. Каган и др.

В настоящее время для выявления ошибок КС широко используются коды, обнаруживающие ошибки.

Наиболее часто для обнаружения ошибок используется контроль по mod 2.

Недостатком используемого метода обнаружения ошибок (контроля по mod 2) является низкая обнаруживающая способность, так как при его использовании обнаруживается только 50% ошибок.

Для повышения процента обнаруживаемых ошибок данным методом информационные разряды разбиваются на отдельные блоки информации с организацией контроля по mod 2.

Следствием данного способа повышения процента обнаруживаемых ошибок является увеличение числа контрольных разрядов и аппаратурных затрат.

Организация контроля информации на основе кодов, обнаруживающих ошибки, позволяет повысить достоверность функционирования вычислителя, но не позволяет обеспечить отказоустойчивость дискретных устройств КС к постоянным отказам.

Для обеспечения отказоустойчивости могут использоваться три идентичные вычислительные машины, при возникновении отказа (обнаружения ошибки) в одном вычислителе его функции возлагаются на исправный вычислитель.

Недостатком данного подхода обеспечения отказоустойчивости является большая аппаратурная избыточность.

Наиболее **эффективным средством** достижения отказоустойчивости дискретных устройств являются **корректирующие коды**, позволяющие, в отличие от структурных методов резервирования, решать данную задачу при минимальных аппаратурных затратах резервного оборудования.

Предметом исследования является теория и методики обеспечения отказоустойчивости функциональных устройств КС на основе корректирующих кодов.

Вопросам использования корректирующих кодов для построения отказоустойчивых вычислительных систем посвящены российские работы А.М. Гаврилова, Н.Д. Путинцева, Ю.Л. Сагаловича, Е.С. Согомоняна, Я.А. Хетагурова, Н.С. Щербакова, А.А. Павлова и других ученых.

Среди зарубежных работ в области использования корректирующих кодов для решения вопросов обеспечения надёжности дискретных устройств

большое значение имеют труды фон Неймана, Мура и Шеннона, Ф.Дж. Мак-Вильямс, Э. Берлекэмп, У. Питерсон.

Анализ данных работ позволяет сделать вывод, что для обеспечения отказоустойчивости рассматриваемых вычислительных систем наиболее целесообразно использовать **линейные коды**.

Применение *циклических кодов* нежелательно, так как они реализуют последовательный метод декодирования, требующий больших временных затрат и, кроме этого, для исправления кратных ошибок требуется большое число информационных разрядов, что исключает возможность его использования для обеспечения отказоустойчивости мало разрядных вычислителей. Например, при исправлении ошибки в восьми разрядном модуле информации код Рида-Соломона требует 2040 информационных разрядов.

В настоящее время для обеспечения отказоустойчивости функциональных узлов КС наиболее широко используются корректирующие линейные коды, исправляющие одиночную ошибку, реализация которых требует минимальных аппаратных затрат на кодирование и декодирование информации, составляющих 30-40 % относительно резервируемого устройства.

В этом случае предполагается, что в дискретных устройствах наиболее вероятно возникновение одиночных ошибок, так как в нормальных условиях эксплуатации радиоэлектронной аппаратуры до 75% составляют одиночные ошибки, а 25% составляют ошибки большей кратности.

На практике данное ограничение является не всегда оправданным, так как с увеличением сложности современных КС, а также в экстремальных условиях работы, (при воздействии электромагнитных или каких либо помеховых излучений и т.п.), возрастает вероятность неправильной коррекции из-за появления ошибок произвольной кратности, имеющих такой

же синдром ошибки, как и корректируемая (появления кратных ошибок, корректируемых как одиночная ошибка).

Поэтому, при построении отказоустойчивых вычислителей КС возникает необходимость использования корректирующих кодов, обнаруживающих и исправляющих **кратные ошибки**.

Однако в настоящее время *мало эффективных методик построения линейных кодов исправляющих ошибки больше двукратной*.

С другой стороны, использование линейных кодов исправляющих ошибки, позволяет обеспечить отказоустойчивость и высокую достоверность *только устройств хранения информации*, в то же время *неизвестны эффективные методы* использования корректирующих кодов для обеспечения отказоустойчивости *преобразователей информации* (сумматоров, регистров сдвига, логических операций И, ИЛИ, НЕ, суммирования по mod2), что является наиболее опасным, так как ошибка при расчетах начинает распространяться в вычислительном процессе.

Например, при контроле арифметических операций наиболее широко используется контроль по модулю (контроль по остаткам, т.е. остаток от результата суммы должен быть равен сумме остатков слагаемых), который требует больших временных и аппаратных затрат и не позволяет исправлять возникающие ошибки, что является характерным и для других видов контроля (контроль по четности суммы, слагаемых, переносов; с использованием дублирования или пара фазной логики и т.д.).

Для контроля большинства логических операций *невозможно сформировать контрольные разряды*, которые оказались бы *совместимыми с данными операциями*, по этой причине наиболее широко используются метод повторения, который также требует временных затрат и не обеспечивает требуемую отказоустойчивость и достоверность функционирования преобразователей информации.

Кроме этого, при использовании корректирующих линейных кодов для обеспечения отказоустойчивости вычислителей не учитываются функциональные особенности рассматриваемой аппаратуры (например, по приспособленности к техническому диагностированию).

Таким образом, **проблемы использования корректирующих кодов** заключается в следующем:

- для защиты памяти КС, работающих в реальном масштабе времени, могут быть использованы **только линейные коды**, при этом **не известны методы** построения линейных кодов, корректирующих больше двукратных ошибок;

- аппаратные затраты на коррекцию одиночной ошибки составляют 30% относительно исходного ЗУ, двукратной -100%, при коррекции ошибки большей кратности $C_{ДЕК} \gg C_{ИСХ}$ (возникает проблема «сторожа над сторожем»);

- возникновение ошибок, кратность которых превышает корректирующие возможности кода, приводит к ошибочной коррекции;

- в настоящее время не известны методы построения корректирующих кодов, обеспечивающих коррекцию ошибок заданной кратности при условии обнаружения максимального количества некорректируемых ошибок.

Кроме этого, осуществление коррекции кратных ошибок на основе линейных кодов приводит к резкому увеличению избыточности кода и большим аппаратным затратам на кодирование и декодирование информации, что не только не позволяет получить требуемый уровень достоверности функционирования отказоустойчивого устройства, но и приводит к снижению данного показателя, т.е. существует **противоречие** между необходимостью обнаружения и коррекции кратных ошибок специализированных КС и большими аппаратными и временными затратами, связанными с обнаружением и исправлением кратных ошибок.

Концептуальным подходом, разрешающим данное противоречие, является разработка корректирующих линейных кодов, исправляющих ошибки *заданной кратности* при условии обнаружения *максимального количества* некорректируемых ошибок и требующих *минимальных* временных и *аппаратурных* затрат на их реализацию для обеспечения *отказоустойчивости* и высокой достоверности функционирования *запоминающих устройств*, с *адаптацией* данных методов кодирования для обеспечения *отказоустойчивости* и достоверности функционирования *узлов* (преобразователей информации) *процессора*.

Цель диссертации: разработка методического аппарата повышения отказоустойчивости функциональных узлов процессоров телекоммуникационных КС для обычных и экстремальных условий работы.

Научная задача: Разработка методического аппарата обеспечения отказоустойчивости процессора телекоммуникационных сетей на основе коррекции ошибок требуемой кратности при заданных временных и аппаратурных затратах на средства обнаружения и коррекции ошибок:

$$O(t) = O_{\text{треб}}(t) / C_K^O = C_{\text{ЗД}}; \quad t_K^O = t_{\text{ЗД}},$$

где $O(t)$ - отказоустойчивость функционального узла КС;

$C_K^O, C_{\text{ЗД}}$ - аппаратурные затраты на обеспечение отказоустойчивости устройства и заданная аппаратурная избыточность;

$t_K^O, t_{\text{ЗД}}$ - временные затраты, соответственно, на коррекцию кратной ошибки и заданное время на обнаружение и коррекцию ошибки.

Методы исследования. При решении научной задачи использованы теоретические методы исследований, основанные на научных положениях: теории линейных корректирующих кодов, теории множеств, теории дискретных автоматов, теории надежности и элементов нечеткой логики.

Положения выносимые на защиту:

1. Модифицированный итеративный код повышенной корректирующей способности.
2. Методический аппарат функционально-кодовой защиты процессора при выполнении арифметических и логических операций.
3. Методика построения структур защищенных вычислителей КС.

Новизна научных исследований заключается в:

1. Предложен модифицированный итеративный линейный код повышенной обнаруживающей и корректирующей способности, адаптированный для защиты преобразователей информации КС.
2. Предложен методический аппарат функционально-кодовой защиты процессора при выполнении арифметических и логических операций.
3. Разработана методика построения структур защищенных вычислителей КС.

Практическая значимость результатов работы состоит в следующем:

- решение рассматриваемой задачи имеет важное значение для Сирии, ожидаемые научные результаты позволяют создать качественно новый уровень отказоустойчивости КС и обычных и в экстремальных условиях работы;
- в зависимости от правила проведения дополнительных проверок, предлагаемые методики позволяют корректировать от 50% до 94% обнаруживаемых ошибок, обеспечить отказоустойчивость и достоверность функционирования компьютерных сетей в реальном масштабе времени, практически без снижения быстродействия исходного устройства.

Достоверность полученных результатов подтверждается использованием математической модели, адекватно отображающей реальные процессы, протекающие в дискретных устройствах, обоснованием и доказательством впервые полученных научных результатов и выводов,

применением широко известных частных научных результатов, результатами внедрения разработок, ясной физической интерпретацией полученных результатов и их непротиворечивостью с существующими методами коррекции ошибок отказоустойчивых вычислителей.

Апробация работы.

Основные положения диссертационной работы докладывались и обсуждались на 4-х международных конференциях: 8-й Международной НТК «Перспективные технологии в средствах передачи информации», Владимир, РФ, 2009г.; Международной НК «Экономическая проблемы ресурсного обеспечения инновационного развития региона» Владимир, РФ, 2009г.; Международной НПК «Факторы развития региональных рынков» Владимир, РФ, 2001г.; 9-й Международной НТК «Перспективные технологии в средствах передачи информации», Владимир, РФ, 2011г.

Публикации. Результаты работы отражены в 11-ти научных трудах, в том числе в 3-х статьях Всероссийского издания из перечня ВАК («Известия института инженерной физики»); в 2-х отчетах о НИР (ГБ №118, ВлГУ).

Результаты научных исследований реализованы:

при разработке отказоустойчивого вычислителя для КС (акты внедрения см. приложения);

Диссертационная работа состоит из введения, трех глав и приложений.

Во введении проводится обоснование актуальности и научной значимости поставленной научной задачи, исследование состояния вопроса и постановка цели научных исследований, применительно, в частности, к Сирии.

В первой главе проводится анализ методик построения отказоустойчивых автоматизированных систем контроля, выбрана модель исследований, введены основные понятия, приняты ограничения и допущения, выявление существующих противоречий.

Обоснована целесообразность и выявлены особенности использования корректирующих кодов для обеспечения отказоустойчивости систем памяти КС. Определен класс корректирующих линейных кодов, требующих минимальных аппаратурных затрат на кодирование и декодирование информации.

Определена проблема использования корректирующих кодов для обеспечения отказоустойчивости преобразователей информации вычислителей КС.

Вторая глава посвящена разработке модифицированного итеративного кода повышенной обнаруживающей и корректирующей способности.

В результате проведенных исследований в работе были предложены шесть подходов построения модифицированных итеративных кодов.

Предлагаемые методы кодирования включают следующие основные положения: информация представляется в две строки, в каждой строке проводится проверка на четность, организуются диагональные проверки с участием, либо без участия контрольных разрядов.

Существенным отличием построения предлагаемых модифицированных итеративных кодов от известных является организация *дополнительных диагональных проверок при вычислении синдрома ошибки*, относительно двух строчной матрицы построенной на основе переданных значений контрольных разрядов и значений контрольных разрядов, сформулированных на основе значений полученных информационных разрядов, что позволило в два раза повысить корректирующие возможности итеративного кода.

Проведены исследования по расчету аппаратурных затрат на реализацию кодирующего и декодирующего устройства при использовании предлагаемых подходов, обоснована кратность исправляемой ошибки.

Установлено, что наименьшие аппаратные затраты соответствуют модифицированному итеративному коду, исправляющему трехкратные ошибки в полубайте информации при этом, наибольшей обнаруживающей и корректирующей способностью обладает шестой подход (корректирует 94 % от общего количества возможных ошибок) однако, наименьшее количество контрольных разрядов, наименьшие аппаратные затраты соответствуют модифицированному итеративному коду при использовании первого подхода кодирования, который был принят для обеспечения отказоустойчивости устройств хранения и передачи информации.

Защищать информационную систему имеет смысл только комплексно.

Предлагаемые методики модифицированных итеративных кодов позволяют:

1. корректировать ошибки трехкратные ошибки в полубайте информации (в настоящее время *неизвестны эффективные методы построения линейных кодов исправляющих больше двух - кратной ошибки*), при условии обнаружения ошибок в остальных разрядах кодового набора, за исключением ошибок трансформируемых в разрешенные кодовые наборы (*новое свойство линейного кода - коррекция ошибок заданной кратности при условии обнаружения максимального количества некорректируемых ошибок*), при этом обеспечивается возможность:

2. исправлять ошибки различной конфигурации (имеет свойства *нелинейного* кода) при условии обнаружения некорректируемых ошибок;

3. осуществлять коррекцию модульных ошибок при малом числе информационных разрядов, т.е. исключить основной недостаток кода Рида-Соломона (при исправлении ошибки в восьми разрядном модуле информации код Рида-Соломона требует 2040 информационных разрядов - поэтому исключается возможность его использования для обеспечения отказоустойчивости мало разрядных специализированных вычислителей КС);

4. иметь минимальные временные затраты на декодирование (в отличие от кодов Рида-Соломона реализующих процедуру циклического декодирования);

5. исключить влияние неисправного резервного оборудования на работу устройств КС при наличии ошибок в контрольных разрядах и отсутствии ошибок;

6. сигнализировать о неисправности устройства памяти при возникновении некорректируемой ошибки.

Третья глава посвящена разработке функционально-кодовой защиты процессора при выполнении арифметических и логических операций (адаптации предлагаемого модифицированного кода для защиты данных операций).

Для формирования “правильных” значений контрольных разрядов возникает необходимость определения правил формирования поправки к значению контрольных разрядов, полученных в результате выполнения арифметической операции S_{k+} .

Правило формирования поправки, при выполнении операции сложения основано построения матрицы поправок, учитывающих перенос единицы в старший разряд, при наличии единиц в одноименных разрядах.

Выявлены свойства корректирующих кодов, позволяющие сформулировать правила *формирования контрольных разрядов для логических операций*.

Основные результаты научных исследований

В результате проведенных исследований разработан научно-методический аппарат функционально-кодовой защиты КС, позволяющий решить задачу обеспечения отказоустойчивости функционирования рассматриваемых систем в экстремальных условиях работы при выполнении ограничений на временные и аппаратурные затраты.

При решении рассматриваемой научной проблемы получены следующие основные результаты:

1. Сформулирована концепция обеспечения отказоустойчивости вычислителей КС для экстремальных условий работы.

2. Разработаны правила построения **модифицированного итеративного линейного кода** повышенной обнаруживающей и корректирующей способности, **отличающегося** от известных методов *организацией дополнительных проверок при формировании синдрома ошибки*, позволяющих существенно повысить корректирующие способности итеративного кода. Предлагаемая методика применения модифицированных кодов, в отличие от существующих методов, позволяет:

осуществлять построение отказоустойчивых ЗУ при малом числе информационных разрядов;

корректировать трехкратные ошибки в полубайте информации при условии обнаружения некорректируемых ошибок;

исправлять ошибки заданной конфигурации.

3. **Выявлены свойства**, и разработаны теоретические положения, позволяющие создать **методический аппарат** функционально-кодовой защиты процессора при выполнении арифметических и логических операций (*впервые разработана процедура адаптации линейных кодов для защиты преобразователей информации*).

4. Разработана **функциональная модель**, отказоустойчивого процессора, реализующего предлагаемый методический аппарат.

5. Получено выражение для оценки аппаратных затрат, вводимых для обеспечения отказоустойчивости при использовании предлагаемой методики кодирования информации, и проведен анализ избыточности в зависимости от разрядности корректируемой модульной ошибки, проведена оценка выигрыша в достоверности функционирования отказоустойчивого устройства, по сравнению с существующими методами.

Полученные научные результаты свидетельствуют о решении крупной научной проблемы, связанной с достижением качественно нового уровня обеспечения отказоустойчивости КС, необходимого для поддержания их живучести в экстремальных условиях работы.

Решение рассматриваемой научной задачи имеет важное значение для Сирии, которое выходит за пределы обеспечения достоверности функционирования компьютерных телекоммуникационных сетей. Разработанный методический аппарат позволяет обеспечить комплексное решение научной задачи повышения вероятности безотказной работы и достоверности функционирования телекоммуникационных устройств, работающих в реальном масштабе времени.

ГЛАВА 1. ЗАЩИТА ТЕЛЕКОММУНИКАЦИОННЫХ КОРПОРАТИВНЫХ КОМПЬЮТЕРНЫХ СЕТЕЙ

1.1. Основные понятия и допущения

Высокая эксплуатационная надежность КС требует включения в своем составе функциональное ядро (специализированную ЭВМ) имеющую, в свою очередь, средства (аппаратурные, микропрограммные, программные), позволяющие автоматически восстанавливать процесс выполнения программы после значительной части возможных ошибок в ее работе.

Сохранение работоспособности функционального ядра при отказах имеет особенно большое значение при эксплуатации систем, работающих в реальном масштабе времени.

Автоматическое восстановление вычислительного процесса при отказах может быть достигнуто путем введения в ЭВМ свойств отказоустойчивости [18].

Отказоустойчивыми будем называть системы, обеспечивающие автоматическое обнаружение ошибки, выявление ее характера (случайная или постоянная), изоляцию неисправности, реконфигурацию системы и восстановление вычислительного процесса [27].

В упрощенном виде вычислитель телекоммуникационных компьютерных сетей может быть представлена одноадресной ЭВМ, как правило, с микропрограммным управляющим устройством, работающей с числами с фиксированной запятой в не модифицированном дополнительном коде (рис. 1.1.1).

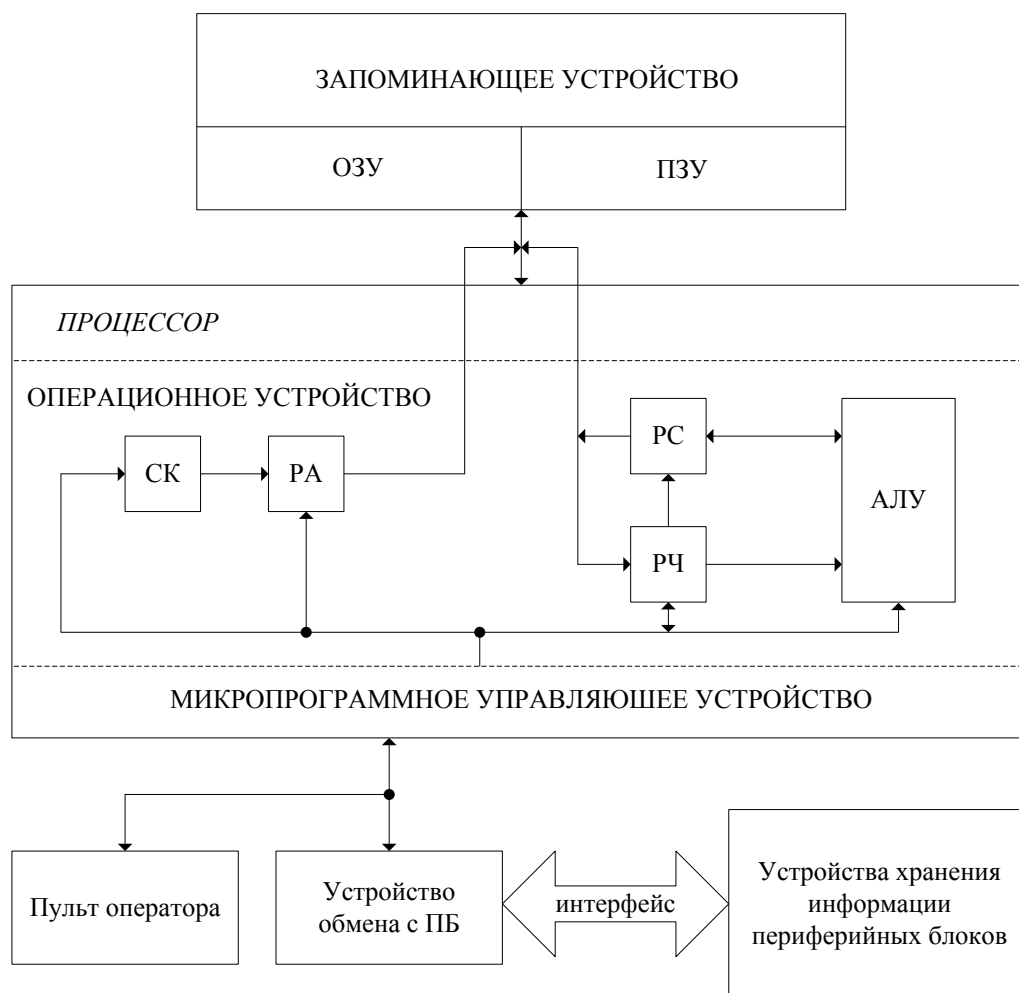


Рис. 1.1.1. Упрощенная структурная схема вычислителя

Основу вычислителя составляет процессор, состоящий из двух основных устройств - операционного и управляющего. Операционное устройство включает в свой состав: арифметико-логическое устройство (АЛУ, использующее k -разрядный комбинационный сумматор параллельного типа);

СК- счетчик команд;

РА-регистр адреса;

РС- регистр сумматора (аккумулятора);

РЧ – регистр числа.

Устройство управления микропрограммного типа, конструктивно выполнено в виде постоянного запоминающего устройства.

Запоминающее устройство состоит из постоянного (ПЗУ) и оперативного (ОЗУ) полупроводниковых запоминающих устройств.

Обмен информацией между вычислителем и периферийными устройствами осуществляется по цифровому интерфейсу с помощью устройства обмена [13,14].

Для ТККС в состав вычислителя входит пульт оператора, предназначенный для ввода информации оператором, вывода информации и признаков состояния системы на панель индикации.

Допущения, принятые при исследовании функциональной модели дискретных устройств:

сигналы, поступающие на вход дискретного устройства идеальны;

дискретное устройство состоит из блоков одинаковой сложности (т.е. блоки содержат равное количество логических элементов), и их вероятности безотказной работы равны;

блоки построены из однотипных логических элементов (элементы, выполняющие вспомогательные функции, такие, как формирователи и т.п. при рассмотрении не учитываются).

Под отказом будем понимать событие, заключающееся в нарушении работоспособного состояния соответствующего дискретного устройства [41].

Несоответствие выходной информации функционального узла вычислителя относительно рассматриваемых входных воздействий (вызванное отказом) будем считать ошибкой [42].

Ограничимся рассмотрением ошибок дискретных устройств, которые проявляются в виде постоянных значений логических сигналов на его выходах типа *const0* и *const1*.

Если ошибка проявляется на одном выходе устройства, то ее будем называть одиночной ошибкой, на двух и более выходах устройства – кратной ошибкой.

Допустим, что рассматриваемые ошибки носят групповой (алгебраический) характер, могут быть одиночными, кратными и являются симметричными.

Предположим, что возникающие неисправности являются правильными, т.е. дискретное устройство, имеющее некоторую неисправность, описывается моделью, принятой для описания исправного устройства.

К возникновению ошибок в полупроводниковых интегральных микросхемах приводят различные физико-химические факторы [63].

Так, например, при эксплуатации БИС ЗУ основной причиной сбоев является наличие в материале полупроводника молекул урана и тория, распад которых вызывает появление альфа-частиц, причём, чем выше плотность упаковки элементов в кристалле, тем сильнее влияние этого фактора, а также, солнечная радиация.

Второй причиной отказа являются космические лучи. Если на уровне моря доля сбоев, вызванных космическими лучами, составляет 10%, то на высотах 10 - 15км интенсивность космических лучей возрастает в 150 раз. Так при исследовании воздействия альфа частиц на модуль памяти фирмы HARRIS NM 6508 ёмкостью 16 килобайт за 371 день полёта спутника наблюдалось 72 сбоя [98].

Кроме этого, в процессе эксплуатации на аппаратуру действует ряд независимых один от другого факторов. Прежде всего, происходит старение элементов схемы. На работу системы также влияют флуктуации, связанные с изменением температуры, величин нагрузки, электромагнитные наводки, изменение значений питающих напряжений и другие внешние факторы.

Многочисленные исследования и опыт эксплуатации дискретных устройств в нормальных условиях позволяют определить параметры возникающих ошибок [42]:

однократные ошибки.....75...80% ;

двукратные ошибки.....15...20%;

трехкратные ошибки.....2...5%;

ошибки прочей кратности.....не более 1%.

В экстремальных условиях работы ТККС (в условиях воздействия вредных факторов), учитывая, что вредные факторы воздействуют одновременно на все элементы системы, возникновение ошибок произвольной кратности следует считать равновероятным событием. Особую актуальность приобретает решение задачи по обеспечению достоверности функционирования и отказоустойчивости ТККС в экстремальных условиях работы.

1.2. Отказоустойчивые функциональные ядра телекоммуникационных систем

1.2.1. Анализ отказоустойчивых ЭВМ

Отказоустойчивая ЭВМ фирмы Stratus содержащую следующие устройства [4] (рис.1.2.1): центральный процессор (ЦП), устройство управления памятью (УУП), дисковый контроллер (ДК), блок управления (БУ), ленточный контроллер (ЛК) и общую шину.

Каждое устройство системы задублировано и каждое дублированное устройство, в свою очередь, проверяется в процессе функционирования с помощью такого же устройства. Фактически в каждое функциональное устройство (ЦП, УУП, ДК, БУ) реализовано в виде двух самостоятельно проверяемых блоков, в которых для построения средств встроенного контроля используется дублирование.

В случае неисправности какого-либо блока системы производится переключение на исправную пару устройств. Факт неправильной работы фиксируется несовпадением выходных результатов устройств, образующих самопроверяемый блок. Поскольку общая шина также задублирована, то при отказе одной производится переключение на исправную шину.

Таким образом полная отказоустойчивость в системе Stratus достигается за счет значительных аппаратных затрат, но при этом гарантируется высокая полнота обнаружения неисправностей (за счет контроля дублированием).

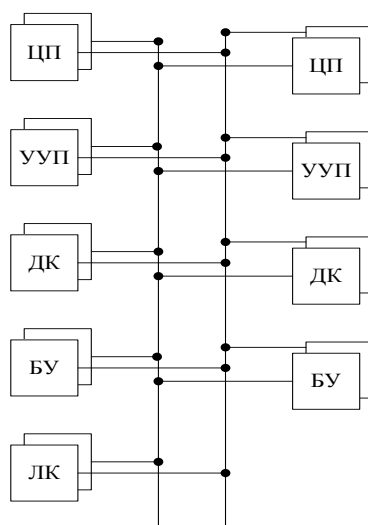


Рис. 1.2.1. Отказоустойчивая ЭВМ фирмы Stratus

Интересной особенностью рассматриваемой системы является исключение временных затрат для восстановления по сбоям. При анализе выходных результатов какой-либо пары устройств эта пара исключается из рабочей конфигурации и подключается исправная пара. Затем с помощью тестовых средств определяется вид неисправности (устойчивая или неустойчивая) и в случае устойчивой неисправности производится замена неисправного устройства пары. После этого самопроверяемый блок вводится в систему. Поскольку в системе восстановление выполняется на уровне аппаратуры, то никакие программные способы образования контрольных точек не используются. Кроме того, в системе не предусмотрена возможность рассылки между блоками информации об их состоянии. Как следствие, все это существенно упростило систему.

Отказоустойчивая вычислительная система фирмы Intel. Система Intel, была задумана и спроектирована как некоторая универсальная система, обладающая двумя основными свойствами: расширяемостью и

отказоустойчивостью, реализованными аппаратурной [7]. В системе на аппаратурном уровне выполняется восстановление по отношению к постоянным отказам и к кратковременным сбоям, причем не только процессоров, но и блоков памяти и интерфейсных шин.

Основная базовая конфигурация системы содержит три 32-разрядных модуля (рис.1.2.2); процессорный модуль, модуль памяти, интерфейсный модуль, а также системную шину. Система построена на пяти СБИС: процессор данных (ПД), который состоит из двух СБИС; интерфейсный процессор (ИП); устройство управления памятью (УУП); шинное интерфейсное устройство (ШИУ).

Система является проблемно-ориентированной с архитектурой, ориентированной на язык программирования АДА [4].

Шинное интерфейсное устройство и УУП позволяют расширить систему; УУП обеспечивает работу с динамической оперативной памятью.

Разделение ресурсов, межпроцессорное взаимодействие в системе, доступ к памяти выполнены посредством введения специальных сигнальных шин, что обеспечивает модульную наращиваемость системы и дает возможность восстанавливать систему, не прерывая процесса ее функционирования.

Поскольку в системе все сигналы формируются независимо от ресурсов системы, наличие или отсутствие кого-либо модуля не влияет на процесс взаимодействия между остальными модулями. В системе отсутствует централизованное устройство арбитража шин, что позволяет всем ШИУ и УУП выполнять свой собственный доступ к системной шине.

Системная шина представляет собой многократно повторяемые 1-разрядные адресные шины и 16-разрядные шины данных, поддерживающие 24-разрядное адресное пространство оперативной памяти. Любая одна передача по системной шине может содержать 1 ... 16 байт данных.

С целью оптимизации пропускной способности системной шины каждая передача делится на отдельные пакеты запросов и ответов. Запросы и ответы могут проходить по (системной шине в то время, как УУП обрабатывает запрос.

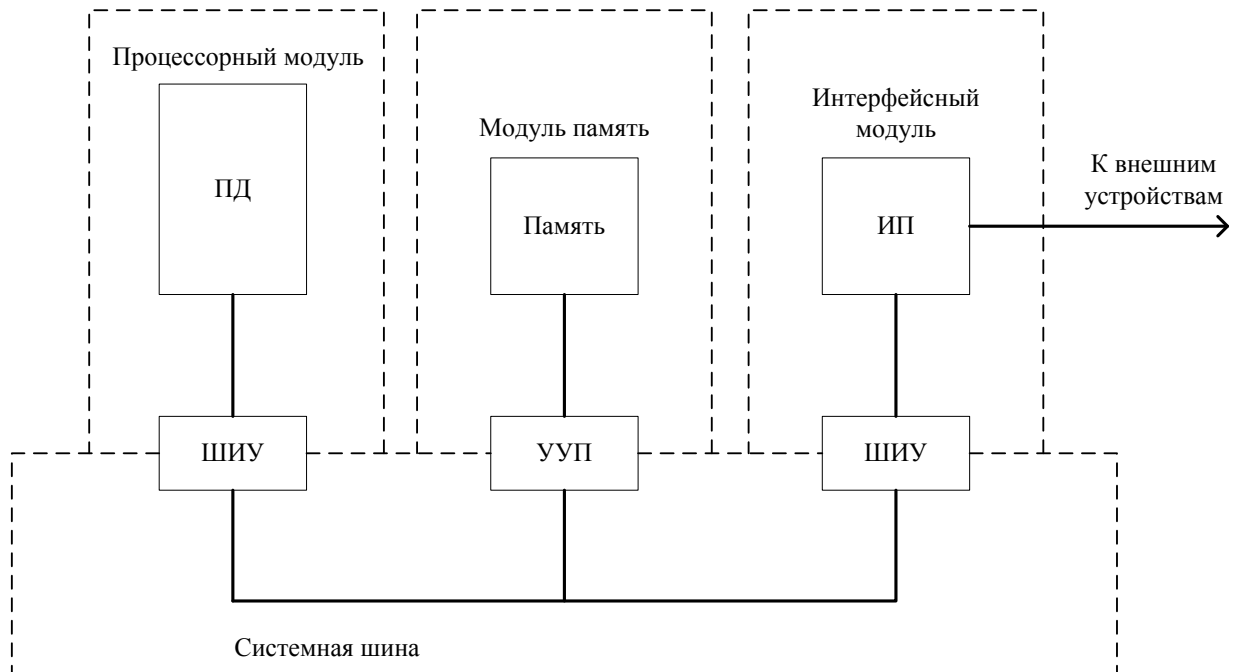


Рис. 1.2.2. Базовая конфигурация системы Intel

Представленная система обладает свойствами отказоустойчивости. Системная шина защищена кодом с проверкой на нечетность, причем для шин данных и управления вводятся свои независимые разряды четности. Таким образом, на системной шине контролируются все одиночные неисправности и многие кратные.

Каждый модуль памяти защищен корректирующим кодом, обеспечивающим коррекцию одиночных ошибок и обнаружение двойных. Адресные ошибки также контролируются, поскольку контрольные разряды при записи в память формируются на основании как данных, которые будут записаны в память, так и адреса, по которому должны быть данные записаны.

УУП обеспечивает возможность программным способом использовать имеющийся в памяти один резервный разряд. Вся оперативная память содержит 40 кристаллов: 32 разряда данных, 7 контрольных разрядов корректирующего кода и 1 резервный разряд. Резервный разряд может подключаться взамен одного неисправного из 39 разрядов.

Для обеспечения отказоустойчивости в системе применяются самопроверяемые модули. Важным понятием в отказоустойчивой системе является понятие зоны распространения ошибки. Зона определяется как модуль или системная шина, которые ограничены контролируруемыми интерфейсами.

Средства обнаружения ошибок располагаются в каждом контролируемом интерфейсе. Такое разбиение системы на зоны позволило существенно упростить алгоритм обнаружения и изоляции ошибок. На рис. 1.2.2 штриховыми линиями выделены 4 зоны распространения ошибок.

Зона распространения ошибок ПД включает в себя ПД, его ШИУ, процессорную шину и вспомогательную логику. Ошибки ПД контролируются путем дублирования СБИС (кристаллов), из которых состоит ПД, и сравнения результатов их работы.

Схема самопроверяемого модуля, состоящего из двух ПД, каждый из которых может быть как основным, так и контролирующим. Выбор статуса ПД (основной или контролирующий) производится при инициализации системы. Оба ПД работают синхронно, параллельно и с одинаковыми данными. Результаты их работы сравниваются схемой сравнения (СС). Например, основным по статусу может являться верхний ПД и результаты работы двух ПД сравниваются на схеме сравнения. При этом сигнал запрета поступает на нижнюю схему с тремя состояниями.

В результате выходная информация основного ПД будет поступать как на выход пары, так и на вход схемы сравнения, на второй вход которой поступает выходная информация контрольного ПД. При неисправности,

возникшей в основном или контрольном ПД, на выходе схемы сравнения появится сигнал ошибки. Если поменять статусы ПД (основной, контрольный), то сравнение будет выполнять вторая СС и сигнал запрета будет поступать на верхнюю схему с тремя состояниями.

Интерфейсный модуль, УУП, ШИУ также выполнены в виде самопроверяемых модулей, контроль в которых осуществляется дублированием со сравнением.

После того, как в системе будет обнаружена ошибка, выполняется восстановление путем повторения. В течение того временного цикла, когда была зафиксирована ошибка, все обмены по системной шине приостанавливаются и отменяются все запросы на обмен по системной шине. Система входит в состояние ожидания, что позволяет устранить влияние кратковременных сбоев. Состояние ожидания может длиться от 16 мкс до 2 с. Время ожидания устанавливается программным способом. По истечении времени ожидания все текущие запросы на обмен (и те, которые были отменены) повторяются ШИУ, что позволяет выполнять восстановление по отношению к сбоям и к одиночным корректируемым отказам в памяти [102,103].

Свойство наращиваемости позволяет организовать систему с непрерывным функционированием при наличии отказов в модулях. Уровень вводимой в систему избыточности устанавливается программно. Восстановление вычислительного процесса выполняют УУП и ШИУ без какой либо программной поддержки.

Система позволяет объединять самопроверяемые модули в пары. При этом один модуль пары является основным, а второй — «теневым». Фактически при такой организации системы все устройства учетверяются по сравнению с неизбыточной конфигурацией.

1.2.2. Анализ построения российских отказоустойчивых ЭВМ

Большое внимание вопросам отказоустойчивости уделялось при производстве вычислительных систем на базе ЕС-ЭВМ [4].

Так, в процессоре этих машин предусматривалось аппаратурно-микропрограммное повторение относительно сбоя. В этом случае, при принятом совмещении инструкций по нескольким уровням предусмотрена организация повторения инструкции или ее части. В аппаратуру повторения входят дополнительные регистры для сохранения информации, которая может быть изменена при выполнении операций процессора.

В дополнительных регистрах сохраняется состояние начальной выборки инструкции, код операции, адрес инструкции, содержимое регистра состояний и т. д. Дополнительные регистры устанавливаются микропрограммой (во время выполнения инструкции). Среди них предусмотрен регистр повторения, который указывает тип выполняемой операции и этап ее выполнения и обеспечивает повторение с требуемого места.

Взаимодействие между аппаратурой и микропрограммой повторения по сбою осуществляется блоком восстановления, состоящим из регистра ошибок (РОш) и комбинационной части. При обнаружении ошибки схемами контроля производится блокировка синхронизации процессора, т. е. прекращение выполнения инструкции, сигнал ошибки запоминается в РОш. По сигналу от блока восстановления (БВ) производится запись ситуации и запуск микропрограммы повторения. Восстанавливается инструкция, выполненная с ошибкой, она повторяется сначала или с заранее определенной контрольной точки.

При успешном повторении осуществляется переход к продолжению нормальной работы процессора. При неуспешном выполнении всех попыток повторения БВ выдает сигнал прерывания от схем контроля, который может вызвать попытку обойти сбой программными методами восстановления,

включенными в программу потребителей. При безуспешных попытках неисправность классифицируется как отказ.

Восстановление в случае возникновения отказа производится обслуживающим персоналом с помощью системы диагностирования. Время поиска сокращено возможностью входа в любую зону диагностических тестов. Ошибки обнаруживаются размещенными в контрольных точках схемами контроля, которые сами могут оказаться неисправными. Наложение неисправностей схем контроля и БВ на неисправности основного оборудования снижает эффективность контроля. Ошибка, вызванная одной неисправностью основного оборудования, обычно выполняется несколькими схемами контроля, т. е. происходит распространение сигнала об ошибке. Однако существуют узлы в ЭВМ, пропадание сигнала ошибки в которых может привести к следующим последствиям:

данная ошибка не будет обнаружена вообще;

нарушатся условия повторения по сбою (в частности, адрес инструкции в дополнительном регистре хранения может содержать ошибку);

будет неверно определена область неисправности;

будет неверно определен тип неисправности (сбой может быть классифицирован как отказ).

Ясно, что такие узлы являются контрольными точками и в них обязательно должны быть размещены схемы контроля. Назовем контрольные точки, которые критичны к пропаданию сигнала ошибки, особыми. Для определения особых контрольных точек необходимо для каждой из них провести анализ на критичность пропадания сигнала ошибки. При этом следует учесть, что заведомо особыми являются такие контрольные точки, которые расположены в местах перехода от одного метода контроля к другому. Например, особыми являются точки перехода от контроля по четности к контролю дублированием и опять к контролю по четности в сумматорах.

Особыми являются схемы контроля дешифраторов и узлов, информация которых сохраняется для повторения. Очевидно, что БВ может быть рассмотрен как особая контрольная точка.

Таким образом, схемы контроля должны обнаруживать также сбои основного оборудования, интенсивность отказов которых на порядок выше интенсивности отказов, по которой определяется время T , не обнаруживаемые отказы и систематические сбои схем контроля особых контрольных точек могут привести к потере сигнала ошибки основного оборудования.

При ошибке единичные сигналы могут поступить на несколько входов БВ.

Блок восстановления рассматриваемого ЦП содержит регистр ошибок (РОш), схемы обработки ошибок и организации повторения, схемы управления синхросигналами и при использовании микросхем средней интеграции состоит из 8—10 ТЭЭ.

Последовательность обработки сигнала ошибки в БВ показана на рис.

1.2.3. Сигналы ошибок основного оборудования поступают на первый ряд триггеров РОш, содержимое которых обновляется в каждом машинном такте. Регистр ошибок разделен на группы по времени поступления сигналов ошибок. Прием ошибок в группе производится одним и тем же синхросигналом. Регистр занимает 100 логических элементов для 80 разрядов.

Схема сжатия и уравнивания времен содержит собирательные схемы и ряд триггеров, осуществляющих задержку сигналов ошибок.

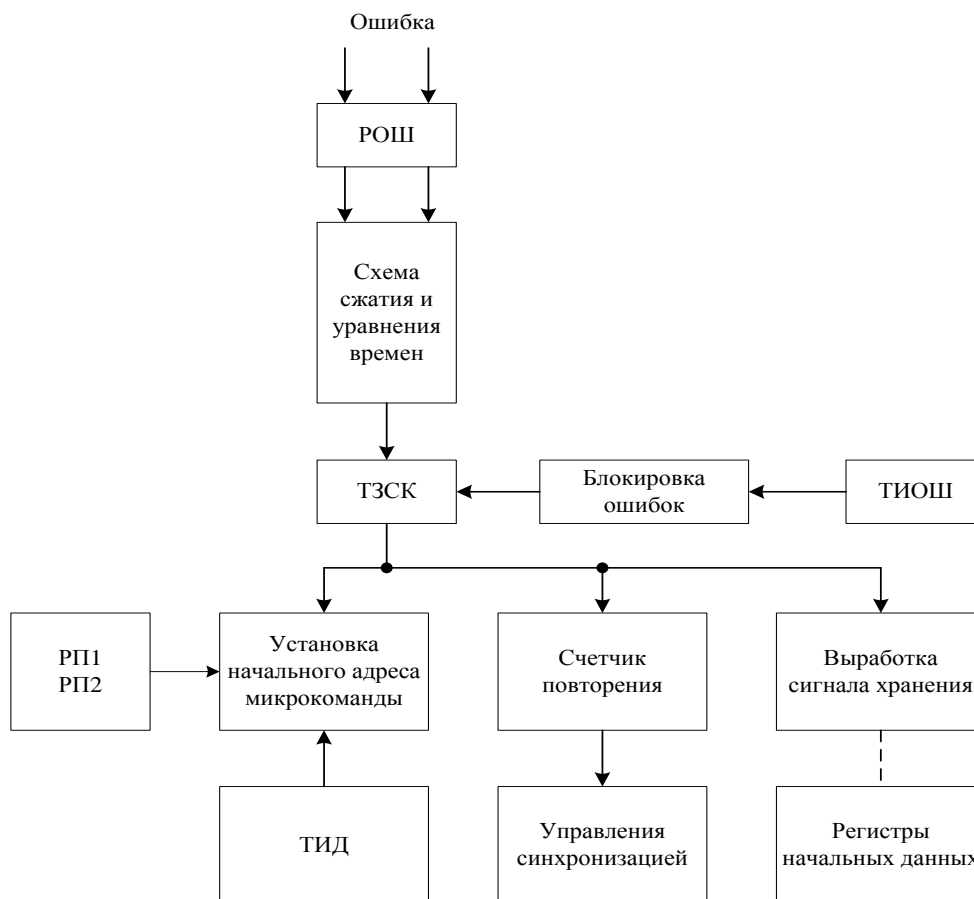


Рис. 1.2.3. Последовательность обработки сигналов ошибки в блоке восстановления

На выходе первого ряда РОШ сигналы ошибок объединяются по времени их возникновения. Уравнивание времен позволяет при блокировке синхросигналов иметь адрес микрокоманды, при выполнении которого возникла ошибка.

При отсутствии блокировки сигнал ошибки устанавливает триггер захвата от схем контроля (ТЗСК), что обеспечивает переход к микропрограмме обработки ошибок. Триггер игнорирования ошибок (ТИОш) блокирует установку ТЗСК от ошибок данных, поступающих при выполнении микропрограммы обработки ошибок. Схемы ТЗСК, ТИОш и блокировки занимают 15 логических элементов.

Единичное состояние ТЗСК блокирует сигнал «хранение», которым разрешается обновление начальных данных повторения, производимого на

основании анализа содержимого регистров повторения $РП_1$ и $РП_2$, занимающих 30 логических элементов. Триггер исходных данных (ТИД) занимает 3 логических элемента, его состояние изменяется при изменении исходных данных команды.

1.2.3. Построение отказоустойчивого ядра ТККС

Проведенный аналитический обзор построения отказоустойчивых функциональных узлов ЭВМ показывает что, при обеспечении отказоустойчивости данных систем необходимо использовать следующие принципы:

обеспечение отказоустойчивости к ошибкам типа “сбой” достигается *повторением операции, возвратом к контрольной точке и программным рестартом;*

обеспечение отказоустойчивости к константным ошибкам типа “const0 или const1” достигается на основе использования *корректирующих кодов и структурных методов резервирования;*

Модульность. Специализированная ЭВМ разбивается на отдельные функциональные блоки “модули” Для каждого модуля выбирается наиболее эффективный метод обеспечения отказоустойчивости;

Быстрое проявление неисправности. При возникновении ошибки модуль должен либо работать правильно (ошибка сразу корректируется), либо немедленно останавливаться (для предотвращения распространения ошибки).

Независимость отказов. Модули и связи между ними должны быть разработаны так, что отказ одного из модулей никак не влияет на работу остальных.

Избыточность. В систему должны быть заранее установлены или сконфигурированы запасные модули, так что при отказе одного из модулей, запасной модуль может заменить его практически немедленно. Отказавший

модуль может ремонтироваться автономно, в то время как система продолжает работать.

Для систем, у которых недопустимо даже кратковременное прерывание вычислительного процесса, требуется многократное резервирование.

Принципиальными требованиями к методам обеспечения отказоустойчивости систем, работающих в реальном масштабе времени, являются:

- высокая обнаруживающая способность (обнаруживать максимальное количество ошибок различной кратности);
- минимальное время обнаружения отказа;
- минимальные аппаратурные затраты на средства контроля;
- возможность обнаружения и определения места возникновения отказа;
- обнаружение и по возможности исправление ошибок, вызванных отказами;
- возможность замены неисправных блоков в процессе выполнения системой основных функций;
- отказоустойчивость системы не должна быть заметной для программиста пользователя.

В связи с тем, что ТККС, как правило, работают в реальном масштабе времени, для них могут быть использованы следующие методы обеспечения отказоустойчивости [42]:

- методы простого резервирования;
- методы избыточной логики с перекрестными соединениями;
- мажоритарные методы резервирования;
- методы, использующие корректирующие коды.

Из всех известных методов повышения достоверности функционирования в настоящее время широко применяется метод контроля по модулю, использующий контрольные символы, являющиеся остатками от деления чисел на некоторый модуль.

Однако определение контрольного остатка требует больших аппаратных затрат и приводит к заметной потере быстродействия аппаратуры [9].

В связи с этим наиболее широко используется метод контроля на чётность (контроль по $\text{mod } 2$), требующий при реализации один контрольный разряд и минимальные аппаратные затраты.

Недостатком метода контроля на четность является низкая обнаруживающая способность.

Существенным недостатком данного метода обнаружения ошибок является то, что он не обеспечивает отказоустойчивость рассматриваемой аппаратуры.

Недостатками метода поэлементного резервирования являются [12]:

- резкое увеличение резервного оборудования;
- увеличение нагрузки на отдельные компоненты, что приводит к снижению нагрузочной способности схем;
- повышение требований к допускам составляющих элементов, поскольку при выходе из строя некоторых элементов выходной сигнал должен оставаться в необходимых допусках.

Методы простого резервирования применительно к схемам, выполненным с высокой степенью интеграции, нецелесообразно использовать ещё и потому, что существенно возрастает число дополнительных связей, что приводит к снижению общей надёжности и существенно усложняет регулировку схем [73].

Избыточная логика с перекрёстными соединениями усложняет построение дискретных устройств, затрудняет их регулировку и диагностирование. Высокая надёжность достигается высокой ценой так, исправление одиночной ошибки требует увеличения аппаратуры в четыре раза [1].

В настоящее время наиболее часто используется мажоритарный метод резервирования. Достоинством метода является большая универсальность,

позволяющая использовать его на различных уровнях разделения дискретных устройств от отдельных узлов до резервирования устройства в целом.

Наиболее эффективным способом повышения отказоустойчивости и достоверности функционирования дискретных устройств является использование корректирующих кодов, позволяющих обеспечить аналогичный эффект при увеличении количества оборудования на 10...30% [9,42,58,63].

1.3. Анализ резервирования на основе линейных кодов

1.3.1. Обоснование требований к методам кодирования информации функционального ядра ТККС

Появление корректирующих кодов связано с созданием помехоустойчивых каналов связи.

В основу построения корректирующих кодов для каналов связи положены следующие допущения [47,48]:

- надежность кодирующих и декодирующих устройств предполагается несравнимо более высокой, чем надежность каналов связи, поэтому проблемы, "кто будет сторожить сторожа", для каналов связи не существует, а сложность кодирующих и декодирующих устройств не является определяющим параметром;

- ошибки в каналах связи взаимонезависимы (даже в случае пакета ошибок, внутри пакета ошибки предполагаются независимыми) и, как следствие, вероятность появления ошибок с увеличением их кратности резко падает.

Основным критерием эффективности применения корректирующих кодов в каналах связи является получение максимальной скорости передачи информации при заданной корректирующей способности кода. Это означает, что скорость тем выше, чем больше отношение k/n (где k — число информационных символов кода, n - общая длина кода) при заданной

кратности d исправляемых ошибок. Поэтому наиболее эффективным с указанной точки зрения являются плотноупакованные коды (совершенные коды). К таким кодам относятся коды Хемминга, коды Голея и код с повторением.

При использовании корректирующих кодов в вычислительных каналах к ним предъявляются требования, отличные от требований, предъявляемых к кодам для каналов связи.

Прежде всего, надежность кодирующих и декодирующих устройств соизмерима с надежностью самого вычислительного канала. Отсюда вытекает необходимость повышения надежности данных устройств. Это можно осуществить, во-первых, применением низкоплотных кодов, проверочные (порождающие) матрицы которых содержат существенно меньшее число единиц по сравнению с плотноупакованными кодами, что упрощает реализацию кодирующих и декодирующих устройств, во-вторых, организацией защиты последних от собственных ошибок.

Критерием эффективности применения корректирующих кодов в вычислительных каналах служит не максимизация отношения k/n , а получение максимума надежности вычислительного канала с коррекцией ошибок при минимуме аппаратных затрат.

В вычислительных каналах, как правило, проводится параллельная обработка информации. Скорость обработки определяется как сумма задержек в самом канале и в кодирующем, и особенно в декодирующем устройствах. Поэтому корректирующие коды, используемые в вычислительном канале, и, прежде всего, в запоминающих устройствах (ЗУ), должны обеспечивать минимальную глубину кодирования и декодирования. Наиболее полно этому требованию удовлетворяют низкоплотные коды.

В вычислительном канале кратность ошибок во многом определяется структурой устройств, так как однократный отказ в результате имеющихся в схеме разветвлений может привести к многократной ошибке на выходе.

Этим, в частности, объясняется сравнительно слабое использование корректирующих кодов в цифровых устройствах с произвольной структурой. Кроме того, поскольку устройства обработки и хранения информации реализуются на интегральных микросхемах (ИМС), в том числе БИС, имеющих от единиц до нескольких десятков выводов, нельзя утверждать, что ошибки на отдельных выводах одной ИМС взаимонезависимы. Дефект в кристалле или подложке ИМС может привести к появлению пачки ошибок на ее выходах. Очевидно, что при формировании кодовых слов пачки ошибок на выходах отдельных ИМС будут занимать строго определенное положение. Такие фиксированные (фазированные) пачки ошибок получили название байтов ошибок. Этот класс ошибок свойственен только вычислительному каналу (не следует смешивать понятие байта ошибок, размерность которого произвольна, и байта информации, под которым в вычислительной технике понимается восьмиразрядное двоичное слово).

В целях повышения достоверности работы цифровых устройств и улучшения их эксплуатационных характеристик целесообразно наряду с исправлением ошибок осуществлять и обнаружение ошибок. Корректирующий код (линейный или нелинейный) с минимальным кодовым расстоянием D_0 исправляет ошибки кратности $d \leq [(D_0 - 1)/2]$ или обнаруживает ошибки кратности $d \leq D_0 - 1$ (знак $[47]$ означает округление до ближайшего меньшего целого числа).

Для того чтобы с помощью кода можно было одновременно исправлять ошибки кратности d_1 (и менее) и обнаруживать ошибки кратности d (и менее), причем $d_1 > d$, необходимо выполнять условие $D_0 \geq d + d_1 + 1$. В этом случае сигнал ошибки кратности d означает, что ошибка была, но исправлена, и дефект должен быть устранен в период проведения ремонта устройства. Сигнал ошибки кратности d_1 показывает, что с декодирующего устройства снимается неправильная информация, при этом сигнал ошибки может быть использован для блокирования этой информации.

Таким образом, корректирующие коды, предназначенные для устройств обработки и хранения информации, должны:

иметь кодирующие и декодирующие устройства минимальной сложности;
 иметь декодирующие устройства, защищенные от собственных отказов;
 иметь минимальные задержки в кодирующем и декодирующем устройствах (минимальную глубину кодирования и декодирования), что требует реализации параллельного способа декодирования;

обеспечивать как коррекции случайных независимых ошибок кратности d , так и коррекцию байтов ошибок длины b ($b > d$);

позволять осуществлять раздельное обнаружение ошибок кратности d и $d + 1$ (b и $b + 1$).

Исходя из перечисленных требований в вычислительных каналах могут быть использованы блочные линейные и нелинейные коды. Применение циклических кодов нецелесообразно, так как они реализуют последовательный метод декодирования, требующий существенно большего времени, чем метод параллельного декодирования.

1.3.2. Выбор метода кодирования, обеспечивающего минимальную сложность декодирующего устройства

Линейные коды допускают три метода параллельного декодирования:
 декодирование методом дизъюнктивных сфер;
 синдромное декодирование;
 мажоритарное декодирование.

Первый метод декодирования требует существенно больших аппаратных затрат, по сравнению с двумя другими, и поэтому его использование является нецелесообразным [42].

Декодирование, использующее стандартное расположение, является декодированием по методу максимума правдоподобия.

В этом случае предполагается, что кратность ошибки не превышает корректирующую способность выбранного кода.

На практике данное ограничение является не всегда оправданным, так как под воздействием внешних воздействий, влияющих одновременно на все

элементы дискретного устройства, возможно возникновение ошибки, превышающей кратность корректируемой. Однако в данном случае лидер смежного класса будет выбран неправильно, что приведет к неправильной коррекции ошибки.

В связи с этим при построении отказоустойчивых устройств необходимо использовать корректирующие коды, исправляющие ошибки заданной кратности при условии обнаружения максимального количества некорректируемых ошибок.

Достоверность функционирования отказоустойчивого дискретного устройства во многом определяется аппаратурными затратами (надежностью) декодирующих устройств обнаружения и коррекции ошибок.

Таким образом, для построения отказоустойчивых дискретных устройств необходимо использовать корректирующие коды:

с минимальной сложностью декодирующего устройства;
 повышенной корректирующей способности ($v = 1 \div k$);
 повышенной обнаруживающей способности (обнаруживать некорректируемые ошибки).

Необходимость использования корректирующих кодов, требующих минимальных аппаратурных затрат на построение декодирующего устройства, заставляет искать специальные классы кодов, для которых существует простой метод декодирования. К таким кодам относят низкоплотные коды [9].

Наименьшую сложность декодирующего устройства ($P_{ДЕК} \rightarrow \max$) и минимальную глубину декодирования (t_{\min}) имеют низкоплотные линейные коды, у которых $k = r$.

Низкоплотные коды описываются матрицей, содержащей преимущественно нули и сравнительно небольшое количество единиц, тем самым уменьшается количество символов, входящих в контрольные соотношения. При увеличении r (уменьшении j) увеличивается число

контрольных соотношений, что приводит к увеличению аппаратных затрат на декодирование. Минимальное количество символов, входящих в контрольные соотношения низкоплотных кодов, позволяют обеспечить минимальные временные затраты на декодирование, так как на пути прохождения сигнала находится минимальное количество сумматоров по mod2.

1.3.3. Исследование коррекции ошибок заданной кратности

При обосновании кратности корректируемых и обнаруживаемых ошибок необходимо учитывать особенности конструктивного исполнения функциональных узлов вычислителя.

В связи с тем, что многие дискретные узлы вычислителей реализуются на отдельных интегральных микросхемах, имеющих от единицы до нескольких десятков выходов, нельзя утверждать, что ошибки на отдельных выводах ИМС взаимно независимы. Отказ в кристалле или подложке ИМС может привести к пачке ошибок (ошибкам модульного типа на её выходах). Поэтому при резервировании дискретных узлов особую актуальность приобретает задача коррекции ошибок заданной кратности и ошибок модульного типа.

Обычно в качестве кодов, исправляющих любые две и обнаруживающих любые три ошибки, применяются коды Боуза-Чаудхури-Хоквингема (БЧХ-коды) с расстоянием $d=6$ [71]. Порождающим многочленом такого кода является многочлен $g(x)=(x+1)p_1(x)p_3(x)$, где неприводимый многочлен $p_3(x)$ имеет своим корнем элемент a^3 . Длина кода является наименьшим общим кратным порядка элементов a и a^3 в мультипликативной группе поля $GF(2^m)$. И если хотя бы один из многочленов $p_1(x)$ и $p_3(x)$ примитивный, то длина кода $n=2^m-1$, однако чаще всего она связана с различными техническими ограничениями, в силу которых число информационных символов равно

степени двойки. Поэтому на практике применяются укороченные коды БЧХ [50].

Частным видом кодов БЧХ являются коды Рида-Соломона, используемые для исправления ошибок модульного типа.

Однако использование данных кодов для резервирования дискретных устройств, работающих в реальном масштабе времени, практически неприемлемо из-за больших аппаратных затрат на кодирование и декодирование и большой глубины декодирования (низкое быстродействие, так как реализуется процедура декодирования циклических кодов).

Заметим, что коды РС допускают мажоритарный метод декодирования, но, как отмечается в [5,71,50] практически это неприемлемо.

Большой практический интерес для коррекции ошибок байтового типа представляют коды с коррекцией ошибок в одной группе бит и обнаружения ошибок в двух группах, являющихся модификацией кода Хемминга [65].

Недостатком данных кодов является сложность (аппаратурные затраты) алгоритма декодирования и, следовательно, низкая надёжность (достоверность функционирования) резервированного устройства.

1.3.4. Проблема использования корректирующих кодов для обеспечения отказоустойчивости функционального ядра

Контроль арифметических операций функционального ядра системы управления

Все рассмотренные выше коды использовались для контроля передачи информации. Особенностью подобного вида контроля является то, что с его помощью решается сравнительно несложная задача — убедиться в неизменности передаваемой информационной комбинации или восстановить эту информацию, если в ней произошли искажения. Совсем другие требования возникают при контроле обрабатываемой информации, которая не остается постоянной, а все время изменяется в процессе тех или иных

операций. Следовательно, в этом случае необходимо обеспечить контроль правильности ее преобразования, т.е. правильности выполнения этих операций. И если возникшая ошибка при передаче информации искажает одно число или отдельные числа, не связанные друг с другом, то та же ошибка при расчетах начинает распространяться в вычислительном процессе, поскольку исходные данные одной операции являются результатом предшествующих операций.

Из множества разработанных методов *контроля арифметических операций* наибольшее распространение получил *контроль по модулю*, который называют также контролем по остаткам или наименьшим вычетам. Суть организации такого контроля заключается в том, что каждому числу, участвующему в операции, ставится в соответствие контрольный код, который представляет собой остаток от деления контролируемого числа на некоторое заранее заданное целое число q , называемое модулем. Использование остатка в качестве контрольного кода возможно по той причине, что любое число A сравнимо с этим остатком.

При выполнении операции над числами та же операция выполняется над их контрольными кодами, после чего контрольный код результата основной операции сравнивается с результатом аналогичной операции над контрольными кодами исходных чисел.

Для нахождения остатка от деления двоичного числа на модуль 3 достаточно просуммировать цифры разрядов контролируемого числа по модулю 3 с учетом знаков четных и нечетных разрядов, что удовлетворяет второму условию.

Одиночная ошибка в одном из разрядов двоичного числа соответствует его изменению на $\pm 2^i$. Для возможности обнаружения ошибки необходимо, чтобы контрольные коды чисел A и $A \pm 2^i$ не совпадали, т.е. нужно выполнить условие

$$R(A) \neq R(A \pm 2^i)$$

Поскольку 2^i не делится на три без остатка, последнее условие всегда выполняется. Кроме одиночных ошибок, контроль по модулю 3 выявляет все двойные ошибки, для которых справедливо указанное условие.

Контроль логических операций функционального ядра

Контроль логических операций, в частности таких поразрядных операций, как логическое сложение (ИЛИ), логическое умножение (И) и исключающее ИЛИ (сложение по модулю 2 или операция неравнозначности), не имеет такой структуры, как контроль арифметических операций. Объясняется это тем, что в отличие от арифметических логические операции выполняются поразрядно и результат операции в каждом конкретном разряде определяется только состоянием соответствующих разрядов операндов, не связанных с другими разрядами чисел. Следовательно, для большинства логических операций невозможно найти общие контрольные разряды, которые оказались бы совместимыми с данной операцией. Реализация же поразрядного схемного контроля в принципе возможна, но неэкономична, так как это потребует резкого увеличения контрольной аппаратуры.

По этой причине наиболее целесообразным является осуществление контроля логических операций путем их повторения, т.е. путем использования временной избыточности. При этом увеличение времени исполнения таких операций не является столь критичным, поскольку в целом выполнение логических операций занимает сравнительно небольшую часть общего времени выполнения программы.

Другим методом контроля является одновременное выполнение двух разных логических операций с последующим сравнением результатов по модулю 2. Для этих целей используются некоторые соотношения между результатами двух различных логических операций над операндами A и B и их алгебраической суммой $A+B$.

1.4. Финансовая и информационная безопасность и риски при проектировании

Управление рисками становится, по сути, повседневной деятельностью для многих участников инновационного ИТ-проекта сразу после его инициации и вплоть до завершения. Вместе с тем классификация рисков и соответственно оценка значимости каждой из категорий рисков в контексте конкретного проекта - это работы, выполняемые на предпроектной стадии[23].

Опытные руководители проектов знают, что наступление многих рисков предсказуемо, а от остальных необходимо страховаться. Наиболее распространенное заблуждение, встречающееся на практике, отождествляет управление рисками и борьбу с последствиями уже возникших проблем. Так вот, цель управления рисками состоит в том, чтобы а) в идеале избежать возникновения проблем или б) минимизировать возможный ущерб для проекта, если избежать проблемы не представляется возможным (например, смена руководства компании и все связанные с этим последствия).

Реагирование на риски - одна из последних стадий функции управления рисками. Поэтому прежде чем перейти к рассмотрению возможных способов реагирования, предлагается определиться с источником возникновения рисков по отношению к проекту и дать им самую общую классификацию.

- **Внутренние риски.** Эти риски непосредственно зависят от деятельности руководителя проекта, команды проекта и других участников, которые могут активно управлять рисковыми ситуациями. Поэтому внутренние риски в достаточной степени управляемы.

- **Внешние риски.** Они возникают вне зависимости от проектной деятельности и порождаются окружением проекта. Они могут учитываться участниками проекта и в некоторой степени быть управляемыми. Важно заметить, что внешние риски можно разделить на две качественные группы: предсказуемые (но неопределенные) - изменение цен, усиление конкуренции на рынке, изменение курсов валют, изменения в налогообложении и т. п.;

непредсказуемые - природные катастрофы, срывы в финансировании из-за смены руководства, неожиданные внешние социальные эффекты и т. п.

Природа возникновения рисков во многом является определяющим фактором при выборе методов реагирования, поэтому в таблице 1.1 мы приводим классификацию рисков более подробно. Кроме того, методы реагирования на риски, в общем случае помимо источника возникновения, определяют следующие факторы:

- принадлежность риска (риск заказчика, исполнителя или обоюдный риск);
- возможные последствия для проекта;
- стадия проекта, на которой выявлен риск;
- возможности участников проекта по предотвращению рисков (достаточность бюджета управления рисками, необходимое время реакции, возможности лоббирования интересов и пр.).

Очевидно, что с учетом такого многообразия факторов дать универсальные рекомендации можно только с учетом специфики конкретного проекта. Однако если сделать определенные допущения, то можно попытаться предложить ряд универсальных способов реагирования на риски с учетом природы их возникновения.

Таблица 1.1 - Пример классификации рисков

Классификация рисков	Характеристика рисков	Примеры рисков
Внутренние риски		
Проектные	Риски возникновения ошибок в проектных разработках, проектной документации, несанкционированный доступ к информации	В проектную документацию закралась ошибка, которая выявилась только на поздней стадии проекта. Результат - перерасход средств и увеличение сроков выполнения проекта
Технические	Риски неправильных технических решений и неправильного использования технических устройств, несанкционированный доступ к информации	Приобретенное по лизингу оборудование оказалось ненадежным и постоянно отказывает в работе. Результат - простои в работе, увеличение сроков проекта, затраты на ремонт оборудования

Классификация рисков	Характеристика рисков	Примеры рисков
Технологические	Риски применения непроверенных технологий и методик, несоблюдения установленных норм и правил, несанкционированный доступ к информации	В результате того, что методология выполнения проекта не предусматривала подготовку и утверждение документа "Отчет о предпроектном обследовании", возникли разногласия в ходе согласования технического проекта
Организационные	Риски возникновения ошибки планирования, неэффективной координации работ, несанкционированный доступ к информации и т. п.	При формировании команды проекта не был назначен ответственный за контроль качества. В результате проект выполнен с большими претензиями со стороны заказчика
Финансовые	Риски перерасхода бюджета проекта из-за неправильных оценок, срывов сроков выполнения работ, ошибок исполнителя и т. п.	При оценке бюджета проекта не было четко определено распределение обязанностей заказчика и исполнителя. В результате проект выполнен с превышением бюджета в несколько раз
Внешние риски		
Природные	Риски, связанные с природными или социальными явлениями (форс-мажор)	В сервер БД ударила молния
Политические	Риски, связанные с нестабильностью деятельности органов власти, несанкционированный доступ к информации	Неожиданные государственные меры регулирования в сферах ценообразования, налогообложения, проектных нормативов и т. п.
Социальные	Риски, связанные с разделением интересов разных социальных групп и ростом социальной активности населения	Вандализм, саботаж, забастовки и пр.
Экономические	Риски, связанные с экономической политикой государства; финансовые риски, связанные с кризисом денежно-кредитной системы, инфляцией; валютные риски, связанные с изменением курсов валют	В результате кризиса 2008 г. многие проекты были закрыты или приостановлены

Можно предложить базовый набор антирисковых мероприятий, применение которых во многом обеспечит эффективное выполнение проекта сразу в нескольких направлениях [78].

· **Учет опыта аналогичных проектов.** Наверное, это один из самых эффективных подходов к оценке, планированию и разработке методов реагирования на риски на начальных стадиях проекта. Главная сложность состоит в том, что в российской практике по результатам выполнения проектов крайне редко подготавливается отчет о закрытии проекта, да и с ведением оперативной документации по управлению проектом тоже не все гладко. Вторая сложность - как эту информацию получить заказчику, так как исполнитель не всегда занимает открытую позицию и не готов проливать свет на все условия и тонкости выполнения проекта. На практике систематизированную информацию об опыте выполнения аналогичных проектов можно получить только у независимых экспертов.

· **Распределение рисков между участниками проекта.** Этот вопрос должен решаться еще на этапе организации проекта в ходе заключения договоров с внешними исполнителями. На данном этапе необходимо максимально предусмотреть соблюдение интересов сторон в случае возникновения как внутренних, так и внешних рисков. Как правило, заказчик привлекает внешних исполнителей, если у него недостаточно опыта выполнения комплексных проектов (реже - в случае нехватки собственных ресурсов). Следовательно, вероятность недостаточной проработки условий договора с исполнителем многократно возрастает, так как выполнение подобных проектов не есть суть бизнеса заказчика.

· **Выделение ресурсов для управления проектом.** Очевидно, что для эффективного выполнения проекта им необходимо эффективно управлять (в том числе рисками), для чего необходимы соответствующие ресурсы: команда менеджмента проекта и бюджет.

· **Планирование резервов.** Международная статистика выполнения проектов свидетельствует о том, что только 16% всех проектов завершаются вовремя и в срок. Очевидно, что помимо просчетов в управлении данная статистика также отражает влияние факторов, предвидеть или воздействовать на которые практически не представлялось возможным. В силу данных причин базовые параметры любого проекта необходимо планировать "с запасом" примерно в 20% (экспертные оценки).

· **Внесение изменений в проектные решения.** Смысл данного пункта поясним на примере: допустим, по календарному плану проекта начало тестовой эксплуатации запланировано на 1 апреля. Результаты аттестации конечных пользователей системы после прохождения обучения показали недостаточность знаний и навыков для работы в системе. Очевидно, в таких условиях необходимо пересмотреть сроки начала тестовой эксплуатации, так как в противном случае количество допущенных ошибок сведет ее результаты «на нет».

· **Привлечение независимых экспертов.** Чтобы компенсировать недостаток знаний заказчика в предметной области выполнения проекта, недостаток человеческих и временных ресурсов, целесообразно рассмотреть вариант привлечения независимых экспертов. На практике хорошо себя зарекомендовали следующие варианты сотрудничества:

независимые эксперты от лица заказчика осуществляют управление проектом, представляют его интересы в ходе всего проекта; заказчик привлекает независимых экспертов к "точечному" контролю качества результатов проекта.

· **Страхование рисков.** Переходя в предметную область информационных технологий, можно констатировать, что внедрение в настоящее время рассматривается как инвестиционный проект, и желание застраховать в страховой компании связанные с внедрением риски - уже не экзотическая ситуация в мировой практике. Подобные прецеденты стали возникать после

нашумевшей истории с банкротством компании FoxMeuer, которая подала в суд и на разработчика, и на ее «внедренца». В российской практике комплексное страхование рисков проектов внедрения пока не распространено. Причин тому несколько: здесь мы можем назвать и сильное удорожание проекта, отсутствие правовой базы и опыта у страховщиков, невозможность оценить экономический эффект от внедрения на стадии начала проекта. Однако не все так плохо - некоторые российские страховщики уже предлагают программы страхования технических рисков (сбой сервера, обрыв линий связи и т. п.) [78].

Таблица 1.2 - Типовые подходы к реагированию на риски

Классификация рисков	Подходы к реагированию
Внутренние риски	
Проектные	Вовлечение в проект ключевых специалистов заказчика как носителей знаний о предметной области проекта Тщательный выбор исполнителя и инструментов выполнения проекта Контроль качества разработанных проектных решений независимыми экспертами Апробация проектных решений на прототипах Выполнение пилотных проектов Улучшение информационной безопасности
Технические	Использование высококачественных технических решений в соответствии с текущими требованиями проекта и перспективами его развития Создание необходимых условий для эффективной эксплуатации технических средств Повышение квалификации персонала, отвечающего за эксплуатацию технической инфраструктуры Улучшение информационной безопасности
Технологические	Контроль соблюдения установленных норм, правил и методик выполнения проекта Внедрение международных стандартов (управления проектами, качества и т. п.) Независимая экспертиза проекта Повышение квалификации персонала Улучшение информационной безопасности
Организационные	Использование передового опыта управления проектами Принятие мер по уменьшению последствий возможных ошибок (регламентация проекта, дробление проекта на этапы и стадии и т. п.) Привлечение внешнего независимого управляющего проектом

Классификация рисков	Подходы к реагированию
	Встречи и переговоры для решения возникающих вопросов и проблем Улучшение информационной безопасности
Финансовые	Особые условия в договоре (условия оплаты, штрафные санкции и т. п.) Исключительное внимание к планированию и контролю исполнению бюджета
Внешние риски	
Природные	Данные риски неуправляемы в том смысле, что их нельзя предотвратить. Однако можно считать их частично управляемыми, так как можно уменьшить последствия и ущерб от их наступления за счет превентивных мер
Политические	Управлять такими рисками значит объективно оценивать связку "бизнес-власть", а для больших проектов и программ целесообразно использовать процедуру лоббирования
Социальные	Обоснование социальной направленности проекта и проведение эффективных PR-компаний Улучшение информационной безопасности
Экономические	Управлять такими рисками можно только на макроуровне, а на уровне проекта эти риски необходимо анализировать и учитывать, чтобы минимизировать возможный ущерб от их наступления

Суммируя вышеизложенное, можно сказать, что управление рисками и управление проектами по смыслу являются схожими понятиями, так как они предназначены для достижения проектом своей цели в рамках заданных ограничений "сроки-бюджет-качество". Однако необходимо четко понимать, что, управляя рисками, можно достигнуть лишь формальных результатов, а управляя проектом - помимо достижения результата по формальным показателям, также создать в компании все необходимые условия для их практического использования. И все это возможно только при хорошей защите от несанкционированного доступа к информации [3,20].

Выводы к главе 1

1. В настоящее время для обеспечения отказоустойчивости функциональных узлов ЭВМ наиболее широко используются корректирующие линейные коды, исправляющие одиночную ошибку, реализация которых требует минимальных аппаратурных затрат на

кодирование и декодирование информации, составляющих 30-40 % относительно резервируемого устройства.

В этом случае предполагается, что в дискретных устройствах наиболее вероятно возникновение одиночных ошибок, так как в нормальных условиях эксплуатации радиоэлектронной аппаратуры до 75% составляют одиночные ошибки, а 25% составляют ошибки большей кратности.

2. Однако в настоящее время *неизвестны эффективные методы построения линейных кодов исправляющих больше двух кратной ошибки.*

С другой стороны, использование линейных кодов исправляющих кратные ошибки, позволяет обеспечить отказоустойчивость и высокую достоверность *только устройств хранения информации ЭВМ*, в то же время *неизвестны эффективные методы* использования корректирующих кодов для обеспечения отказоустойчивости *преобразователей информации* (сумматоров, регистров сдвига, логических операций И, ИЛИ, НЕ, суммирования по mod2), что является наиболее опасным, так как ошибка при расчетах начинает распространяться в вычислительном процессе.

3. Для контроля большинства логических операций *невозможно сформировать контрольные разряды*, которые оказались бы *совместимыми с данными операциями*, по этой причине наиболее широко используются метод повторения, который также требует временных затрат и не обеспечивает требуемую отказоустойчивость и достоверность функционирования преобразователей информации.

4. Таким образом, ***проблема*** использования корректирующих кодов заключается в следующем:

4.1. Для обеспечения отказоустойчивости ЗУ специализированных ЭВМ наиболее целесообразно использовать корректирующие коды, позволяющие (по сравнению со структурными методами резервирования) обеспечить заданную отказоустойчивость при минимальных аппаратурных затратах;

4.2. Для защиты ТККС, работающих в реальном масштабе времени, могут быть использованы *только линейные коды* (циклические коды, турбо коды и т.д. для исправления кратных ошибок требуют большого числа информационных разрядов, что не приемлемо при резервировании мало разрядных ЭВМ, и кроме этого требуют больших временных затрат на декодирование).

4.3. Минимальные аппаратурные затраты на построение декодирующего устройства достигаются при использовании низкоплотных линейных кодов;

4.4. В настоящее время *не известны* эффективные методы построения линейных кодов, корректирующих больше двукратных ошибок.

4.5. При использовании корректирующих линейных кодов аппаратурные затраты на коррекцию одиночной ошибки составляют 30% относительно исходного ЗУ, двукратной – 100%, при коррекции ошибки большей кратности $C_{\text{ДЕК}} \gg C_{\text{ИСХ}}$ (возникает проблема “сторожа над сторожем”).

4.6. Возникновение ошибок, кратность которых превышает корректирующие возможности кода, приводит к ошибочной коррекции.

4.7. Существующие корректирующие коды используются, как правило, только для защиты устройств хранения и передачи информации (ОЗУ, ПЗУ) и *не адаптированы* для исправления ошибок преобразователей информации (для выполнения арифметических и логических операций процессором: сумматора, регистров сдвига, блока логических операций и т.д.).

ГЛАВА 2. ОБНАРУЖЕНИЕ И КОРРЕКЦИЯ ПРИ ФУНКЦИОНАЛЬНО-КОВОДОЙ ЗАЩИТЕ ЯДРА ТККС

2.1. Поэлементная и комплексная информационная защита ТККС

Современные ТККС являются сложными системами с иерархической информационно-управляющей структурой. Большое количество

интеллектуальных элементов и многоуровневые связи между ними, а также соединение с внешними, в том числе глобальными сетями, выводят защиту информации на уровень задач, определяющих жизнеспособность систем, которые они обслуживают.

В большинстве современных систем (покажем на примере АСУ ТП) можно выделить три уровня. На нижнем уровне располагаются аппаратные средства – датчики, исполнительные механизмы, управляющие контроллеры. На среднем уровне находятся групповые контроллеры, устройства сопряжения с объектами. Верхний уровень реализуется на персональных компьютерах, где с помощью специальных пакетов реализуется интерфейс с оператором-технологом, выполняющим супервизорное управление технологическим процессом.

Пример трёхуровневой системы управления сложным технологическим процессом показывает рис.2.1.1. Количество контролируемых параметров в таких системах измеряется сотнями, управляемых параметров – десятками.

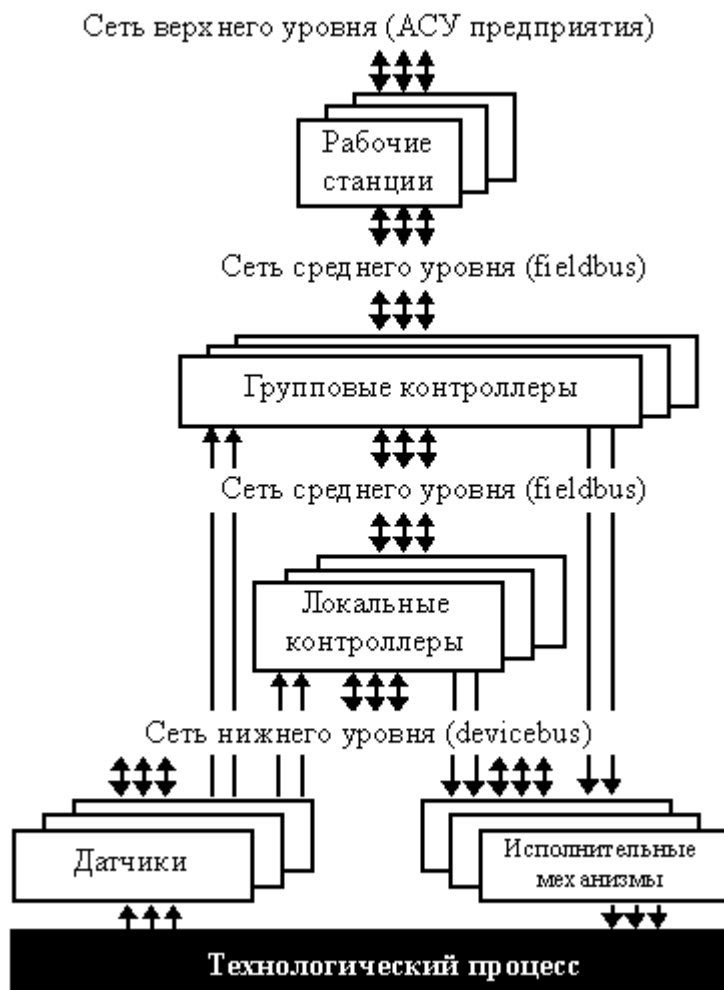


Рис.2.1.1. Типовая структура трёхуровневой АСУ ТП

Технологический процесс управляется с помощью исполнительных механизмов, а информация о параметрах технологического процесса снимается с помощью датчиков. Ввод информации о параметрах, вычисление управляющих воздействий и их выдача на исполнительные механизмы осуществляется локальными контроллерами (ЛК) или групповыми контроллерами (ГК). Координация управления процессом осуществляется рабочими (операторскими) станциями (РабС) с помощью локальной сети. Количество РабС определяется в зависимости от сложности процесса, но их всегда больше одной, так как применяется дублирование для повышения надежности. В свою очередь, РабС объединены как между собой, так и с другими абонентами цеха (предприятия), с помощью локальной сети верхнего уровня типа Ethernet. Необходимо подбирать тип каждой локальной

сети и её параметры под конкретные требования: гарантированное время доставки; скорость передачи; объём пакетов; количество аппаратно реализованных функций в контроллерах сети и т.д. Особо надо отметить, что сеть верхнего уровня АСУ ТП чаще всего имеет контакт с глобальной сетью.

В глобальной сети профессионалы могут нанести значительный ущерб информационной безопасности из любой точки земного шара. Конечно, изменились и механизмы обеспечения информационной безопасности: возникли межсетевые экраны, системы обнаружения вторжений и др. Как только накопилась критическая масса разнообразных средств защиты, появились компании, которые стали называть себя системными интеграторами в области информационной безопасности. Речь пошла о комплексном подходе к информационной безопасности, когда все элементы системы безопасности взаимодействуют между собой.

Однако сейчас тенденции развития информационных технологий таковы, что комплексного подхода к обеспечению информационной безопасности уже недостаточно. Комплексные системы, в существующем понимании, остаются по отношению к самой информационной системе внешними. Большинство систем такого рода строятся сегодня исключительно как системы защиты периметра.

Чаще всего проблему ассоциируют с «защитой от несанкционированного доступа» (ЗНСД), то есть с деятельностью, определяемой как «предотвращение получения защищаемой информации заинтересованным субъектом с нарушением прав или правил доступа к защищаемой информации, установленных правовыми документами или собственником, владельцем информации». Однако данный подход не отвечает всем требованиям по обеспечению безопасности эксплуатируемого объекта. Необходимо включить в определение действия по защите информации от непреднамеренного воздействия (ЗИНВ), призванную предотвратить негативное воздействие на защищаемую информацию от ошибочных

операций пользователя, сбой технических и программных средств информационных систем [88,85].

Нельзя обеспечить безопасность системы управления технологическим процессом, не обеспечивая безопасность компонентов, из которых она состоит. Недостаточно выставить охрану, нужно еще задействовать механизмы безопасности самих объектов. Поэтому многие крупные производители программного обеспечения и технических средств приступили к разработке инструментов обеспечения безопасности своих продуктов. Весь спектр продуктов: от систем и рабочих станций до интеллектуальных датчиков и исполнительных механизмов, - снабжается системой паролей для разграничения доступа и средствами регистрации доступа, защищёнными физическими каналами и профилями локальных сетей, дублированными узлами связи и бесперебойным электропитанием. Раньше внутренние механизмы собственной безопасности программных и технических продуктов практически отсутствовали. Сейчас они становятся неотъемлемой частью любого продукта. Причём эти механизмы могут быть задействованы на том уровне, который нужен пользователю. Можно вообще не пользоваться встроенным механизмом безопасности, можно задействовать его частично, обеспечивая только необходимые функции защиты информации, а можно использовать полностью. Но если такого механизма в продукте нет, то обеспечить его безопасность только внешними средствами становится невозможно.

Именно поэтому сейчас к построению систем информационной безопасности нужно подходить уже даже не комплексно, а **системно**. В этом случае, с одной стороны, информационная безопасность будет представлять собой включение в работу встроенных механизмов защиты в программных и технических компонентах информационной системы. С другой стороны, внешнюю защитную оболочку будет создавать комплексная система информационной безопасности (включая системы мониторинга и управления

информационной безопасностью, интегрированные с системами мониторинга и управления информационной системой). О том, что подобный подход является верным, свидетельствуют активные усилия по обеспечению безопасности своих продуктов ведущих зарубежных (Microsoft, SAP, HP, IBM) и отечественных производителей («Овен», «Элемер», НПП «Автоматика» и других), — а также развитие линеек взаимоувязанных продуктов комплексного обеспечения информационной безопасности компаний Symantec, ISS, IBM, Computer Associates, Adastra и т. д.

При таком подходе система информационной безопасности становится двухуровневой. Верхний уровень обеспечивает комплексная система защиты, а нижний — собственные встроенные механизмы безопасности программных продуктов и технических средств. Обойти внешнюю защиту можно, внутреннюю — гораздо сложнее. До тех пор, пока не будет решен вопрос о встраивании механизма безопасности в продукты, по-настоящему защищенная система не получится.

Есть базовый, или обязательный уровень обеспечения информационной безопасности, подразумевающий наличие, например, средств антивирусной защиты, средств разграничения доступа к ресурсам, средств аутентификации и механизма администрирования системы. Для реализации внутренних правил и регламентов безопасности можно ограничиться мониторингом системы, чтобы иметь возможность фиксировать «следы» действий как легальных пользователей, так и нелегальных.

С точки зрения внешних воздействий надо постараться поставить барьер для защиты от несанкционированных действий извне. Для этого используется, например, межсетевой экран (firewall), определяющий права внешних пользователей и процессов по отношению к внутренним. Это практически обязательный набор, который присутствует во всех достаточно сложных информационных системах.

Дальше начинается другой уровень, который регулирует наличие и пропорции тех или иных механизмов защиты в зависимости от уязвимостей и угроз конкретному технологическому процессу. Необходимо помнить, что при построении системы информационной безопасности надо двигаться не «снизу», а «сверху». То есть в первую очередь, надо уделять внимание тем угрозам для информационной системы, которые представляют опасность для технологического процесса, а не для информационной системы.

Отличительной особенностью АСУ ТП от бытовых и корпоративных вычислительных сетей является то, что они представляют собой комплекс сильно связанных элементов, выполняющих единые функции по управлению объектом автоматизации. Другое характерное отличие АСУ ТП как сложной системы - это неполная определенность ее поведенческой модели, что затрудняет анализ безопасности системы.

При реализации максимального класса защиты стремятся закрыть возможность доступа со всех прогнозируемых направлений, где может быть осуществлена атака на систему. Типичными мерами «глухой защиты» являются: удаление клавиатуры у компьютеров и запрет сетевого доступа; наложение электронных замков, перемычек на электронные платы в контроллерах; использование механических ключей на шкафы автоматики и т.п. Это эффективный и радикальный способ решения ЗНСД, однако, он превращает программируемую технику АСУ ТП в непрограммируемую.

Другой крайностью является защита с использованием некоторой единичной организационно-технической меры, чаще всего это защита с помощью паролей. Это очень архаичный и ненадежный способ ЗНСД (т.е. не реализованы меры, обеспечивающие персонализацию объекта, связанного с паролем).

В то же время, можно сколько угодно защищать систему внешними устройствами, но оказаться под ударом по вине собственных сотрудников: как свидетельствует официальная статистика, около 80 % всех инцидентов

происходит по вине легальных пользователей. Причем эти удары далеко не всегда вызваны злым умыслом, зачастую это случается из-за рассеянности пользователей, из-за незнания правил работы с системой.

Оцениваются меры обеспечения ЗНСД можно путем рассмотрения угроз безопасности АСУ ТП со стороны различных групп пользователей:

- *разработчики* знают все пароли, в том числе и системный, который открывает доступ ко всему программному обеспечению (ПО). Для них приемлема только глухая защита, а это означает, что их нельзя допускать на объект;

- *наладчики* также обязаны знать все пароли и тоже могут производить вредные модификации ПО. От них не спасет ни один из указанных выше способов;

- *эксплуатационный персонал*, который может быть существенно ограничен в своих действиях, если парольная защита выполнена надлежащим образом. Если при эксплуатации АСУ ТП необходимо производить ввод информации, настройку или иные действия над ПО, то парольная защита не будет обеспечивать ЗНСД;

- *операторы-технологи* - единственная группа персонала, для которой при помощи комбинации глухой защиты и паролей можно добиться хорошей ЗНСД;

- *системные администраторы* на практике выполняют всю совокупность функций обслуживания ПО АСУ ТП, включая его установку, настройку, модификацию и ЗНСД. По отношению к ПО они "всемогущи" и "бесконтрольны".

Таким образом, приведенные выше способы ЗНСД для АСУ ТП приходится признать неэффективными и недостаточными.

Задача ЗНСД сводится к определению факта и места воздействия; проверке, является ли оно санкционированным или нет; принятию мер по устранению вредных последствий в случае возникновения

несанкционированного воздействия. Меры по устранению воздействий должны быть регламентированы на основе анализа возможных атак на безопасность системы.

В современных АСУ ТП применяется в основном распределенная, или поэлементная, модель ЗНСД, в которой каждый элемент реализует свой комплекс мер по защите от НСД. Эта модель скопирована с офисных систем, в которых взлом одного компьютера не затрагивает работу других. Поэтому система только слегка деградирует, а не выходит из строя. Если ее восстановят через сутки, ничего страшного не произойдет.

В АСУ ТП все элементы сильно связаны друг с другом. Поэтому порча одного элемента, например контроллера или сетевого коммутатора, может привести к отказу целого измерительного канала или множества каналов одновременно. Поэтому простои в течение суток недопустимы.

К построению систем информационной безопасности АСУ ТП нужно подходить системно. При таком подходе система информационной безопасности становится двухуровневой. Верхний уровень обеспечивает комплексная система защиты, а нижний — собственные встроенные механизмы безопасности программных продуктов и технических средств.

Централизованные системы ЗНСД должны создаваться на основе анализа угрозы НСД применительно ко всей системе управления, а не по отдельным её компонентам. Необходимо анализировать угрозы, возникающие не только в процессе нормальной эксплуатации системы, а на всех этапах жизненного цикла, включая разработку и наладку

2.2. Разработка модифицированных итеративных кодов повышенной обнаруживающей и корректирующей способности

В результате проведенных исследований (см. приложение 1) были предложены шесть подходов построения итеративных кодов, которые могут быть использованы для защиты постоянной памяти специализированных

ЭВМ.

Алгоритм кодирования информации первым подходом включает следующие положения:

1) информационные разряды делятся на две равных части и представляются в две строки

2) для каждой строки информационной матрицы организуется проверка на четность, т.е. информационная матрица представляется в виде:

$$\begin{array}{l} y_1 \quad y_2 \dots \dots \dots y_{k/2} \quad r_{\times 1} \\ y_{(k/2)+1} \quad y_{(k/2)+2} \dots \dots \dots y_k \quad r_{\times 2} \end{array} \quad (2.1)$$

3) для полученной информационной матрицы организуются правые и левые диагональные проверки. Число диагональных проверок (число контрольных разрядов диагональных проверок) определяется по формуле:

$$R_{\bar{A}} = k + 4 \quad (2.2)$$

4) кодовый набор передается в виде:

$$Y = y_1 y_2 \dots y_k r_1 r_2 \dots r_{k+4} \quad (2.3)$$

5) результат сложения значений сигналов переданных и сформированных контрольных разрядов даст синдром ошибки:

$$E = e_1 e_2 e_3 \dots e_{k+4} \quad (2.4)$$

6) при формировании синдрома ошибки относительно полученных и сформированных значений контрольных разрядов организуются дополнительные диагональные проверки, число которых определяется выражением:

$$R_V = 2(k + 5) \quad (2.5)$$

7) в результате имеем множество ошибок заданной кратности (в данном случае от одиночной до кратности $k-1$, определяемое выражением:

$$N = \sum_{i=1}^{k-1} C_n^i), \quad \text{характеризующихся определенными значениями синдрома}$$

ошибки и дополнительной проверки.

8) множество N разбивается на четыре подмножества $N = n_1 + n_2 + n_3 + n_4$, где n_1 - синдромы, имеющие совпадения по дополнительным проверкам (некорректируемые ошибки, признак отказа устройства);

n_2 -подмножество групп (каждая группа включает 2^k -одинаковых значений синдромов) при наличии ошибок только в информационных разрядах;

n_3 -подмножество групп (каждая группа включает 2^k -одинаковых значений синдромов) при наличии ошибок только в контрольных разрядах;

n_4 -подмножество групп (каждая группа включает 2^k -одинаковых значений синдромов) при наличии ошибок одновременно в информационных и контрольных разрядах.

Заметим, что для ошибок, не превышающих кратность $k-1$ нет ошибочных кодовых наборов, трансформируемых в разрешенные (исправные) кодовые наборы.

Второй подход кодирования полностью включает правила кодирования информации, используемые в первом подходе кодирования, но при этом наряду с контрольными разрядами, сформированными относительно диагональных проверок, передаются контрольные разряды, сформированные относительно проверок на четность столбцов информационной матрицы.

Третий подход основан на следующих правилах кодирования:

1) из прямых инверсных значений информационных разрядов формируется информационная матрица:

$$\begin{array}{cccc} y_1 & y_2 & \dots & y_k \\ \bar{y}_1 & \bar{y}_2 & \dots & \bar{y}_k \end{array}$$

2) для полученной информационной матрицы организуются правые и левые диагональные проверки. Число диагональных проверок (число контрольных разрядов) определяется по формуле:

$$R_d = 2(k + 1)$$

3) кодовый набор передается в виде:

$$Y = y_1 y_2 \dots y_k r_1 r_2 \dots r_{2(k+1)}$$

Процедура получения множеств синдромов ошибок для рассматриваемого кода включает положения, рассмотренные при построении первого и второго методов кодирования.

Четвертая методика кодирования включает следующие положения:

1) из прямых инверсных значений информационных разрядов и полученного значения разряда четности формируется двухстрочная информационная матрица, для каждой строки которой организуется проверка на четность:

$$\begin{array}{c} y_1 y_2 \dots y_k r_{\text{ЧЕТ}} \\ \bar{y}_1 \bar{y}_2 \dots \bar{y}_k \bar{r}_{\text{ЧЕТ}} \end{array}$$

2) для полученной информационной матрицы организуются правые и левые диагональные проверки. Число диагональных проверок определяет число контрольных разрядов (контрольные разряды, соответствующие проверкам на четность не передаются). В этом случае число контрольных разрядов определяется по формуле:

$$R_d = 2(k + 2) .$$

3) кодовый набор передается в виде:

$$Y = y_1 y_2 \dots y_k r_1 r_2 \dots r_{2(k+2)} .$$

Пятый и шестой подходы кодирования включают положения, аналогичные третьему подходу кодирования, но при этом в пятом подходе дополнительно к контрольным разрядам, сформированным относительно диагональных проверок, передается контрольный разряд четности полученный относительно информационных разрядов, а в шестом - дополнительно передаются разряды четности, сформированные относительно прямого и инверсного кодовых наборов.

Таким образом, предлагаемые методы кодирования включают следующие основные положения: информация представляется в две строки, в

каждой строке проводится проверка на четность, организуются диагональные проверки с участием, либо без участия контрольных разрядов.

В таблице 2.1 представлены результаты исследования корректирующих и обнаруживающих способностей предлагаемых методик кодирования информации (число информационных разрядов – 4, кратность ошибок изменяется от 0 до 3-х).

Таблица 2.1 - Обобщенная характеристика предлагаемых подходов

Контролируемый параметр	№ варианта					
	1	2	3	4	5	6
Количество информационных разрядов	4	4	4	4	4	4
Количество контрольных разрядов	8	10	10	12	13	14
Количество разрядов доп. проверки	18	20	22	26	28	30
% коррекции ошибок	46	75	72	88	90	94
Общее количество ошибок	4784	7520	7520	11152	13344	15808
Количество некорректируемых кодов	16	16	16	16	16	16
Количество совпадающих кодов доп. пр.	2544	1824	2096	1272	1280	960
Количество не совпадающих кодов доп. пр.	2224	5680	5408	9864	12048	14832
Ошибки только в информационных разрядах	224	224	224	224	224	0
Ошибки только в контрольных разрядах	592	2064	2016	4416	5680	7312
Ошибки и в информационных и в контрольных разрядах	1408	3392	3168	5224	6144	7520

Анализ табл. 2.1 показывает, что из предлагаемых шести подходов кодирования 4-х информационных разрядов, наибольшей обнаруживающей и корректирующей способностью обладает шестой подход (94 % от общего количества возможных ошибок), а наименьшее количество передаваемых контрольных разрядов (8 разрядов) может быть получено при использовании первого подхода.

2.3. Обоснования рациональной методики построения итеративного кода для обнаружения и коррекции ошибок в устройствах хранения информации функционального ядра ТККС

2.3.1. Оценка аппаратурных затрат на реализацию предлагаемого метода кодирования

Аппаратурные затраты, вводимые для обеспечения отказоустойчивости, выразим через простейшие (двухвходовые) логические элементы. В этом случае сложность одного элемента неравнозначности равна четырем простейшим логическим элементам.

Число диагональных проверок (число контрольных разрядов диагональных проверок) формируется относительно информационных разрядов, дополненных контрольным разрядом на четность определяется по формуле:

$$R_D = k + 4 \quad (2.6)$$

Общие аппаратурные затраты для вычисления значений контрольных разрядов четности информационных разрядов, реализованные на сумматорах по mod 2 составят: $C_{ЧЕТ} = k - 2$.

Аппаратурные затраты кодирующего устройства для вычисления диагональных проверок, относительно информационных разрядов, дополненных контрольным разрядом на четность и выраженные через сумматоры по mod 2 равны:

$$C_{\text{mod}2} = k \cdot \quad (2.7)$$

Тогда аппаратурные затраты кодирующего устройства составят

$$C_{КУ \text{ mod}2} 2k - 2 \text{ сумматоров по mod 2.} \quad (2.8)$$

Аппаратурные затраты кодирующего устройства выраженные через простейшие логические элементы равны:

$$C_{КУ_b} = 8(k - 1). \quad (2.9)$$

Кроме этого, в состав устройства входит схема вычисления синдрома ошибки (схема поразрядного сравнения), аппаратурные затраты которой оценивается выражением:

$$C_{СИН} = 4R_D = 4(k + 4), \quad (2.10)$$

Аппаратурные затраты регистра памяти, при условии что для записи одного разряда потребуется отдельный триггер, выполненный на четырех двухвходовых логических элементах и два элемента И (соответственно для записи и считывания информации) составят:

$$C_{Pez} = 6R_D M = 6 * (k + 4) * M, \quad (2.11)$$

где M - число слов памяти.

Число логических элементов И, разрешающих поступление значений сигналов информационных разрядов на входы кодирующего устройства при записи и считывании информации составит:

$$C_{И1} = 2k. \quad (2.12)$$

Аппаратурные затраты элемента ИЛИ, обеспечивающего поступление сигналов на вход кодирующего устройства при записи или при считывании информации составят:

$$C_{ИЛИ1} = k. \quad (2.13)$$

Число логических элементов И, разрешающих поступление значений сигналов контрольных разрядов при вычислении синдрома ошибок составят:

$$C_{ИСИНДР} = k + 4. \quad (2.14)$$

Аппаратурные затраты дешифратора, определим учитывая что множество ошибок заданной кратности (в данном случае от одиночной до

кратности $k-1$) определяется выражением $N = \sum_{i=1}^{k-1} C_n^i$ и при этом данное

множество разбивается на четыре подмножества $N = n_1 + n_2 + n_3 + n_4$,

где n_1 - ошибки, синдромы которых имеют одинаковые дополнительные проверки (некорректируемые ошибки, признак отказа устройства);

n_2 - подмножество ошибок только в информационных разрядах (каждая группа включает 2^k - одинаковых значений синдромов);

n_3 - подмножество ошибок только в контрольных разрядах (каждая группа включает 2^k - одинаковых значений синдромов);

n_4 - подмножество ошибок одновременно в информационных и контрольных разрядах.

Каждое n_i - подмножество ошибок включает l_j - групп, а каждая группа включает 2^k - одинаковых значений синдромов.

В этом случае группа:

l_1 - число групп синдромов для ошибок только в информационных разрядах;

l_2 - число групп синдромов для ошибок только в контрольных разрядах;

l_3 - число групп синдромов для ошибок, возникающих одновременно в информационных и контрольных разрядах.

Тогда аппаратурные затраты дешифратора составят:
 $C_{ДЕШ} = 16(l_1 + l_2 + l_3)$ r -разрядных элементов И, или для двухвходовых элементов:

$$C_{ДЕШ} = 16(l_1 + l_2 + l_3)(R_D - 1) = 16(l_1 + l_2 + l_3)(k + 3). \quad (2.15)$$

Выходы дешифратора соответствующие синдромам, входящих в n_3 , объединим с помощью элемента ИЛИ на n_2 - входов, тогда аппаратурные затраты данного элемента, выраженные через двухвходовые логические элементы, составят:

$$C_{ИЛИ2} = (l_2 - 1). \quad (2.16)$$

Подмножество групп $l_2 + l_3$ объединим с помощью $(l_2 + l_3)/2$ - входовых элементов ИЛИ, для подачи управляющих сигналов, корректирующих соответствующий информационный разряд.

В этом случае аппаратурные затраты данной группы элементов ИЛИ составят:

$$C_{ИЛИ3} = k[(l_1 + l_3)/2 - 1]. \quad (2.17)$$

Аппаратурные затраты, обеспечивающие формирование сигнала «Отказ», для группы синдромов, принадлежащих подмножеству n_1 , включают: r -входовый элемент ИЛИ, $(k+1)$ -входовый элемент ИЛИ, элемент НЕ и двухвходовый элемент И.

При реализации на двухвходовых элементах данные аппаратурные затраты составят:

$$C_{ОТКАЗ} = k + 3 + k + 2 = 2k + 5. \quad (2.18)$$

Аппаратурные затраты элемента И, разрешающего прохождение управляющих сигналов на корректор составят:

$$C_{И2} = k. \quad (2.19)$$

Аппаратурные затраты корректора составят: $C_{КОР} = 4л.$

Таким образом, для реализации предлагаемого метода коррекции аппаратурные затраты составят:

$$C_{ОБЩ} = C_{ЧЕТ} + C_{КУ} + C_{СИН} + C_{РЕГ} + C_{И_1} + C_{ИЛИ_1} + C_{ИСИНДР} + \\ + C_{ДЕШ} + C_{ИЛИ_2} + C_{ИЛИ_3} + C_{ОТКАЗ} + C_{И2} + C_{КОР}$$

или

$$C_{ОБЩ} = 6 * M * (k + 4) + k * [(l_1 + l_3) / 2 - 1] + 16 * (k + 3) * (l_1 + l_2 + l_3) + l_2 + 24k + 14 \quad (2.20)$$

Аппаратурные затраты декодирующего устройства включают затраты на реализацию схемы синдрома ошибок, дешифратора, $(k+1)$ - логических элементов ИЛИ, объединяющих группы выходов дешифратора, логических элементов для формирования сигнала отказ, корректора и логических элементов И, разрешающих прохождение управляющих сигналов на корректор:

$$C_{ДЕК} = C_{СИН} + C_{ДЕШ} + C_{ИЛИ_2} + C_{ИЛИ_3} + C_{ОТК} + C_{И2},$$

или

$$C_{ДЕК} = 16(l_1 + l_2 + l_3)(k + 3) + k[(l_1 + l_3) / 2 - 1] + 7k + 8 + l_2. \quad (2.21)$$

На рис. 2.5-6 представлены соответственно зависимости аппаратурных затрат резервного оборудования от числа информационных разрядов накопителя ЗУ.

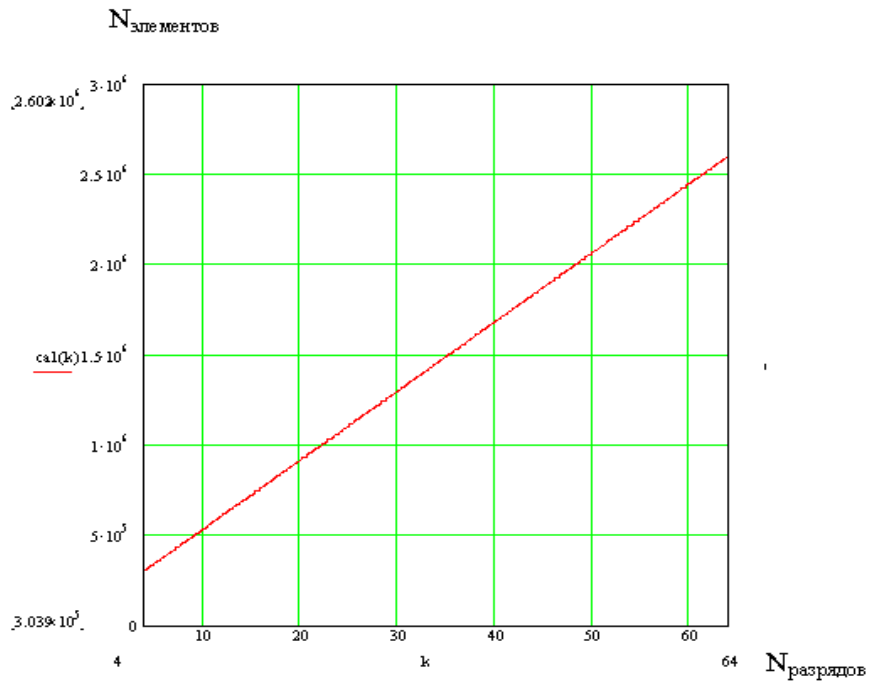


Рис. 2.5. Аппаратурные затраты на резервирование

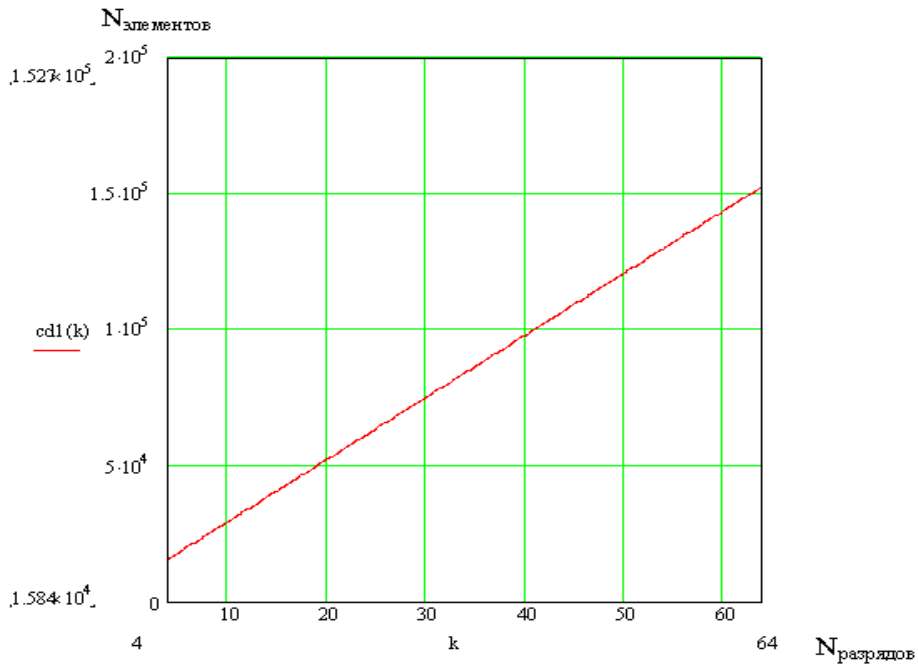


Рис. 2.6. Аппаратурные затраты декодирующего устройства

Из представленных графиков видно, что аппаратные затраты на резервирование и аппаратные затраты декодирующего устройства имеют линейную зависимость от числа информационных разрядов.

2.3.2. Достоверность функционирования отказоустойчивого запоминающего устройства

Оценку достоверности функционирования отказоустойчивых ЗУ рассмотрим на примере для четырех информационных разрядов с использованием первого варианта кодирования. В этом случае: $r=k+4=8$; $n=k+r=12$.

Предположим, что емкость накопителя M составляет 10000 4-х разрядных ячеек памяти, а интенсивность отказа одного логического элемента равна

$$\lambda_i = 1 * 10^{-9} \text{ 1/ч}, (p(t) = e^{-10^{-9} * t}). \quad (2.22)$$

Вероятность безотказной работы накопителя по одному выходу равна:

$$p1(t) = p(t)^{6M}. \quad (2.23)$$

Аппаратурные затраты на построение декодирующего устройства составят 30000 двухвходовых логических элемента.

Достоверность функционирования отказоустойчивого ЗУ оценим используя выражение:

$$D(t) = p_{ДЕК}(t) \sum_{i=0}^{k-1} C_n^i p1(t)^{(n-i)} [1 - p1(t)]^i + p_{ДЕК}(t) \sum_{i=1}^n C_n^i p1(t)^{(n-i)} [1 - p1(t)]^i - \quad (2.24)$$

$$P_{ДЕК}(t)^2 \sum_{i=0}^{k-1} C_n^i p1(t)^{(n-i)} [1 - p1(t)]^i * \sum_{i=1}^n C_n^i p1(t)^{(n-i)} [1 - p1(t)].$$

2.3.3. Обоснование выбора метода обнаружения и коррекции ошибок в устройствах хранения и передачи информации

Проведем оценку влияния кратности исправляемой ошибки аппаратные затраты и достоверность функционирования устройств памяти при реализации предлагаемых подходов кодирования информации.

Сравнительную оценку достоверности функционирования электронных устройств в зависимости от кратности исправляемой ошибки проведем на примере использования четвертого подхода.

Исходные данные:

- количество информационных разрядов $k=4$;

- количество контрольных разрядов $R=8$;
- вероятность безотказной работы одного простейшего логического элемента $P(t) = e^{-10^{-11} \cdot t}$;
- емкость накопителя $M = 6000$;
- вероятность безотказной работы одного выхода $P1(t) = P(t)^{6M}$.

В табл. 2.2-2.5 приведена числовая характеристика рассматриваемого метода. По этим данным определяем множества $n_1^i, n_2^i, n_3^i, n_4^i, n_5^i$, а также значения l_1^i, l_2^i, l_3^i , где i – кратность ошибки (в таблице указаны в скобках).

Таблица 2.2 - Изменение кратности ошибки от 0 до 3 (корректируется 88%)

Контролируемый параметр	Всего	Корректируемых	0 ошибок	1 ошибка	2 ошибки	3 ошибки
Общее количество ошибок	11152	9864	16	256	1760	9120
Некорректируемые	16	-	-	-	-	-
Имеющие совпадения	1272	-	-	0	0	1272
Без совпадений	9864		-	256(16)	1760(110)	7848(491)
Только в информационных	224	224(14)	-	64	96	64
Только в контрольных	4768	4416(276)	-	192	960	3264
И в контрольных и в информационных	6144	5224(328)	-	0	704	4520

Таблица 2.3 - Изменение кратности ошибки от 0 до 4 (корректируется 59%)

Контролируемый параметр	Всего	Корректируемых	0 ош.	1 ош.	2 ош.	3 ош.	4 ош.
Общее количество ошибок	40272	23648	16	256	1920	8960	29120
Некорректируемые	16	-	16	-	-	-	
Имеющие совпадения	16608	-	-	80	848	3104	12576
Без совпадений	23648		-	176(11)	1072(67)	5856(366)	16544(1034)
Только в информационных	240	144(9)	-	48	48	48	0
Только в контрольных	12688	9232(577)	-	128	704	2496	5904
И в контрольных и в информационных	27328	14272(892)	-	0	320	3312	10640

Таблица 2.4 - Изменение кратности ошибки от 0 до 5 (корректируется 17%)

Контролируемый параметр	Всего	Корректируемых	0 ош.	1 ош.	2 ош.	3 ош.	4 ош.	5 ош.
Общее количество ошибок	110160	18944	16	256	1920	8960	29120	69888
Некорректируемые	32	-	16	-	-	-		16
Имеющие совпадения	91216	-	-	256	1520	7680	23200	58560
Без совпадений	18944		-	0	400(25)	1280(80)	5920(370)	11312(707)
Только в информационных	240	16(1)	-	0	16	0	0	0
Только в контрольных	25360	7552(472)	-	0	256	768	2176	4352
И в контрольных и в информационных	84528	11376(711)	-	0	128	512	3744	6960

Таблица 2.5-Числовые значения множеств групп n , l для ошибок кратности 0-5

Множество групп	Кратность ошибки		
	0-3	0-4	0-5
n_1	16	11	0
n_2	110	67	25
n_3	491	366	80
n_4	-	1034	370
	0-3	0-4	0-5
n_5	-	-	707
l_1	14	9	1
l_2	276	577	472
l_3	328	892	711

Используя выражение (2.24) проведем сравнительную оценку достоверности функционирования от использования четвертого подхода кодирования от кратности исправляемой ошибки. В результате получим графические зависимости (рис. 2.7), отображающий зависимость достоверности функционирования запоминающего устройства от времени.

D4(t)

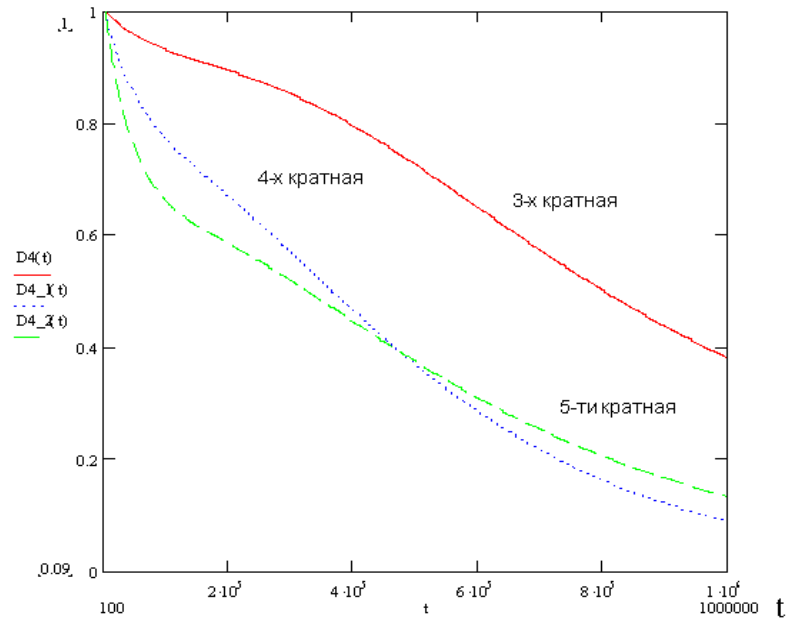


Рис. 2.7. Сравнительная оценка достоверности функционирования предлагаемого метода V-4 от кратности исправляемой ошибки

Используя данные из табл. 2.4 рассчитаем аппаратные затраты для построения запоминающего устройства $ca4$, сложность декодирующего устройства $cd4$, достоверность функционирования $D4$.

$$ca4 := 6M \cdot (k + 8) + k \cdot \left(-1 + \frac{114 + 134}{2} \right) + 16 \cdot (k + 7) \cdot (124 + 134 + 114) + 124 + 27 \cdot k + 48$$

$$ca4_1 := 6M \cdot (k + 8) + k \cdot \left(-1 + \frac{114_1 + 134_1}{2} \right) + 16 \cdot (k + 7) \cdot (124_1 + 134_1 + 114_1) + 124_1 + 27 \cdot k + 48$$

$$ca4_2 := 6M \cdot (k + 8) + k \cdot \left(-1 + \frac{114_2 + 134_2}{2} \right) + 16 \cdot (k + 7) \cdot (124_2 + 134_2 + 114_2) + 124_2 + 27 \cdot k + 48$$

$$cd4 := 16 \cdot (114 + 124 + 134) \cdot (k + 7) + k \cdot \left(-1 + \frac{114 + 134}{2} \right) + 11 \cdot k + 124 + 40$$

$$cd4_1 := 16 \cdot (114_1 + 124_1 + 134_1) \cdot (k + 7) + k \cdot \left(-1 + \frac{114_1 + 134_1}{2} \right) + 11 \cdot k + 124_1 + 40$$

$$cd4_2 := 16 \cdot (114_2 + 124_2 + 134_2) \cdot (k + 7) + k \cdot \left(-1 + \frac{114_2 + 134_2}{2} \right) + 11 \cdot k + 124_2 + 40$$

$$P14(t) := p(t)^{cd4} \cdot \left[\begin{array}{l} p1(t)^{16} + p1(t)^{15} \cdot (1 - p1(t)) \cdot err_14 + p1(t)^{14} \cdot (1 - p1(t))^2 \cdot err_24 + \\ + p1(t)^{13} \cdot (1 - p1(t))^3 \cdot err_34 \end{array} \right]$$

$$P14_1(t) := p(t)^{cd4_1} \cdot \left[\begin{array}{l} p1(t)^{16} + p1(t)^{15} \cdot (1-p1(t)) \cdot err_14_1 + p1(t)^{14} \cdot (1-p1(t))^2 \cdot err_24_1 + \\ + p1(t)^{13} \cdot (1-p1(t))^3 \cdot err_34_1 + p1(t)^{12} \cdot (1-p1(t))^4 \cdot err_44_1 \end{array} \right]$$

$$P14_2(t) := p(t)^{cd4_2} \cdot \left[\begin{array}{l} p1(t)^{16} + p1(t)^{15} \cdot (1-p1(t)) \cdot err_14_2 + p1(t)^{14} \cdot (1-p1(t))^2 \cdot err_24_2 + \\ + p1(t)^{13} \cdot (1-p1(t))^3 \cdot err_34_2 + p1(t)^{12} \cdot (1-p1(t))^4 \cdot err_44_2 + p1(t)^{11} \cdot \\ \cdot (1-p1(t))^5 \cdot err_54_2 \end{array} \right]$$

$$P24(t) := p(t)^{cd4} \cdot \sum_{i=1}^{n4} \left[\left(\frac{n4!}{i! \cdot (n4-i)!} \right) \cdot p1(t)^{n4-i} \cdot (1-p1(t))^i \right]$$

$$P24_1(t) := p(t)^{cd4_1} \cdot \sum_{i=1}^{n4} \left[\left(\frac{n4!}{i! \cdot (n4-i)!} \right) \cdot p1(t)^{n4-i} \cdot (1-p1(t))^i \right]$$

$$P24_2(t) := p(t)^{cd4_2} \cdot \sum_{i=1}^{n4} \left[\left(\frac{n4!}{i! \cdot (n4-i)!} \right) \cdot p1(t)^{n4-i} \cdot (1-p1(t))^i \right]$$

$$D4(t) := P14(t) + P24(t) - P14(t) \cdot P24(t)$$

$$D4_1(t) := P14_1(t) + P24_1(t) - P14_1(t) \cdot P24_1(t)$$

$$D4_2(t) := P14_2(t) + P24_2(t) - P14_2(t) \cdot P24_2(t)$$

Из графика видно, что лучшим из рассматриваемых вариантов является метод с кратностью ошибок от 0 до 3-х.

2.3.4. Сравнительная оценка аппаратных затрат при реализации предлагаемых методов кодирования информации

Используя выражения для оценки аппаратных затрат и данные таблиц построим графики (рис. 2.8-9) зависимости аппаратных затрат для построения запоминающего устройства ca_i и сложности его декодирующего устройства cd_i (i – номер предлагаемого варианта V1-V6) от количества информационных разрядов (k).

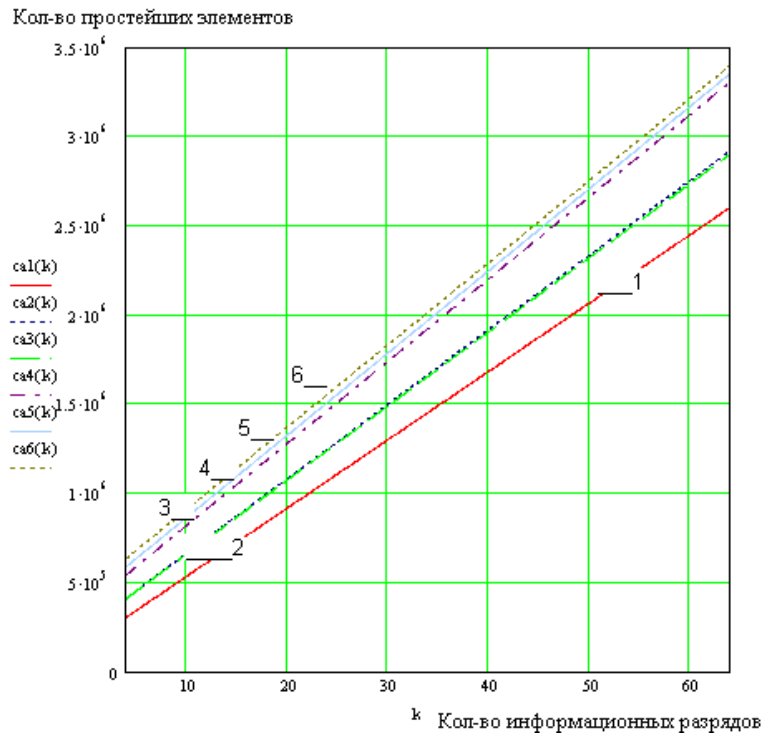


Рис. 2.8. Аппаратурные затраты запоминающего устройства

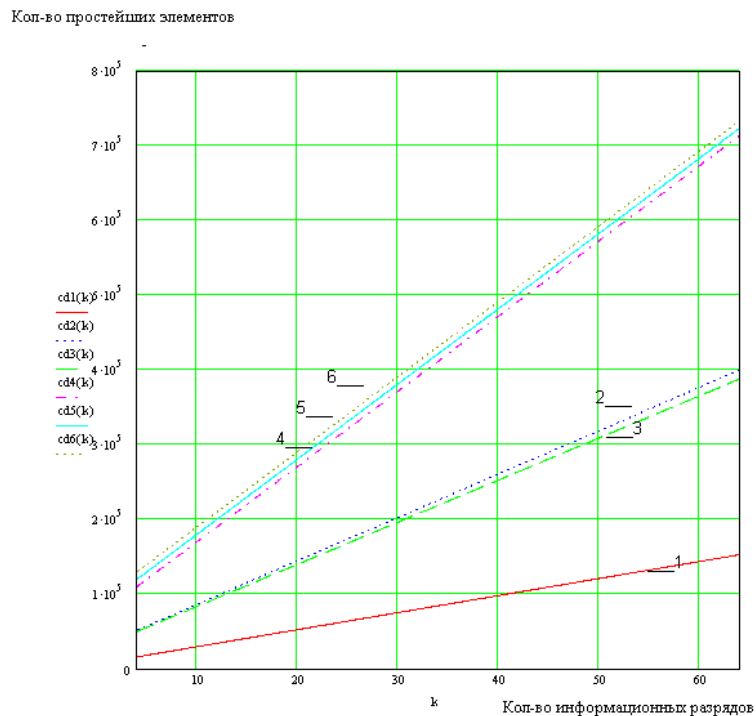


Рис. 2.9. Аппаратурные затраты декодирующего устройства

Наименьшие общие аппаратурные затраты запоминающего устройства соответствуют первому из предлагаемых подходов.

2.3.5. Сравнительная оценка достоверности функционирования при реализации предлагаемых методов кодирования информации

Используя выражения 2.15-2.19 построим график (рис. 2.10) и проведем оценку достоверности функционирования запоминающего устройства, работающего на основе предлагаемых подходов (варианты V1-V6).

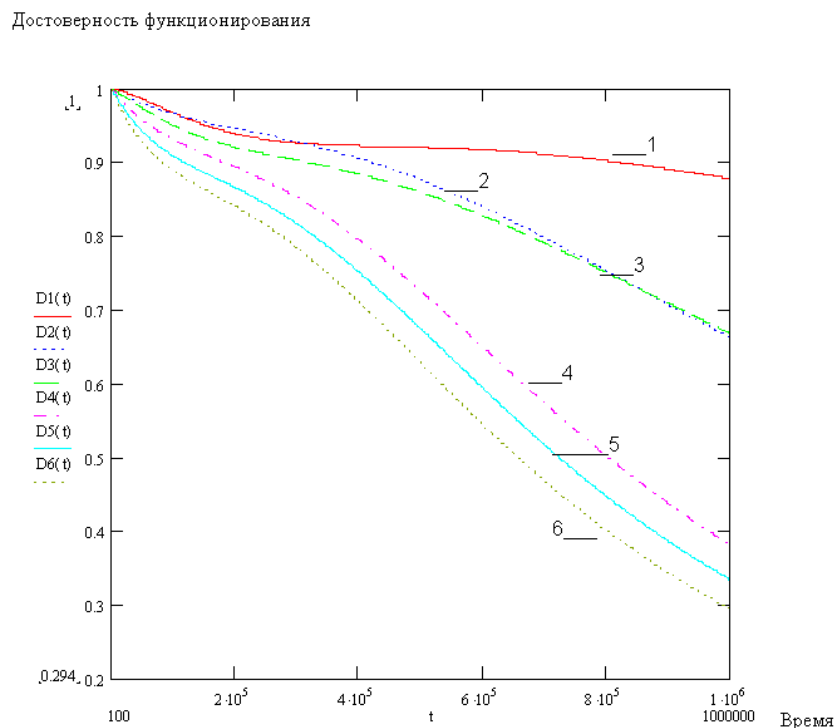


Рис. 2.10. Сравнительная оценка достоверности функционирования ЗУ при использовании предлагаемых подходов кодирования

Лучшие характеристики с точки зрения достоверности функционирования имеет устройство, построенное на основе первого метода.

Это объясняется использованием минимального количества простейших логических элементов для его построения.

Однако при этом процент обнаруживаемых и исправляемых ошибок у первого варианта – наименьший, а шестого – самый большой.

2.3.6. Обоснование методик кодирования информации при увеличении числа информационных разрядов

Сравнительная оценка достоверности функционирования аппаратуры, изложена для случая использования четырех разрядных информационных слов. Однако при увеличении разрядности информационных слов картина может измениться.

В случае использования восьми разрядной аппаратуры существует два подхода к выбору лучшего метода:

- 1) применение одного из предлагаемых методов для целого восьми разрядного слова;
- 2) применение одного из предлагаемых методов для каждой половины восьмиразрядного слова (два по четыре).

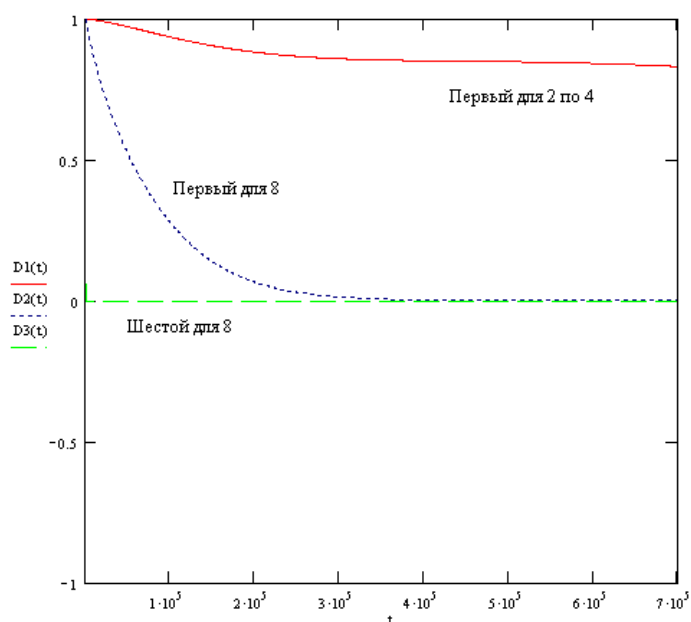


Рис. 2.11. Сравнительная оценка достоверности функционирования предлагаемых методик кодирования для восьми разрядного слова

Из рис.2.11, видно, что лучшим с точки зрения достоверности функционирования является первый метод, примененный для каждой половины восьми разрядного кода.

2.3.7. Сравнительная оценка предлагаемого подхода с существующими методами

Проведенный выше анализ показывает, что лучшую достоверность функционирования дает первый из предлагаемых методов. Поэтому он и был выбран для сравнения с существующими (применяемыми на практике) в настоящее время методами.

Учитывая, что при использовании предлагаемого метода корректируются 12 одиночных ошибок, 31 двойных, 96 тройных и обнаруживаются $2^n - 16$ ошибок получим сравнительную оценку зависимостей достоверности функционирования ЗУ от использования предлагаемого метода кодирования, мажоритарного метода резервирования, дублирования, метода проверки на четность, которые представлены на рис. 2.12.

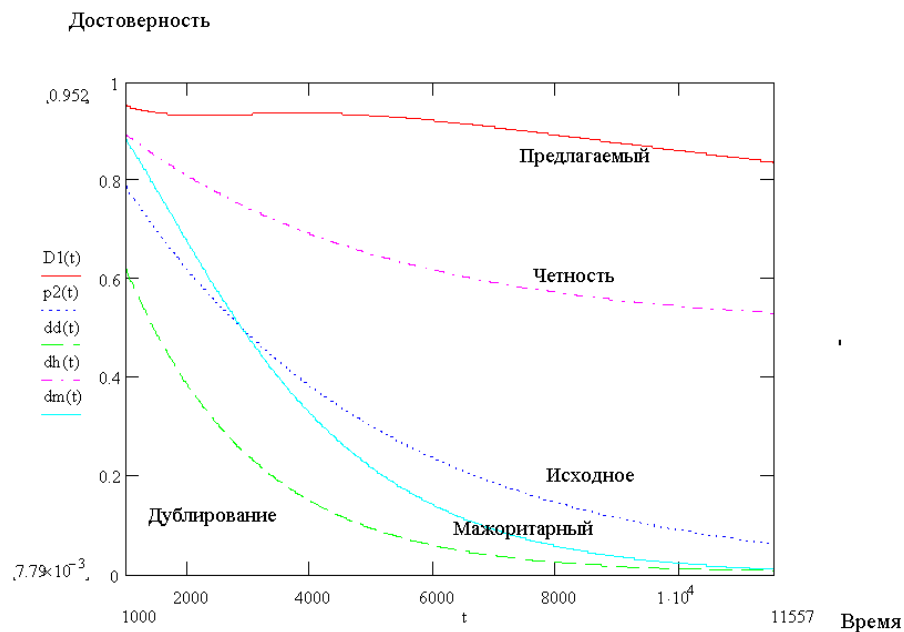


Рис. 2.12. Сравнительная оценка достоверности функционирования предлагаемой методики резервирования с существующими методами

Таким образом, для изготовления отказоустойчивого устройства памяти (макетного образца) предлагается использовать первую методику примененный для каждой половины восьми разрядного кода т.е. при изготовлении микросборок с восьми разрядным информационным кодом потребуются аппаратные затраты вдвое большие, чем с четырех разрядным

кодом, а при изготовлении микросборок с шестнадцатью информационными разрядами – аппаратные затраты увеличатся в четыре раза.

2.4. Рекомендации для технической реализации предлагаемых методик кодирования

Причиной постоянных неисправностей в ИС ЗУ являются отказы ИС, а случайных — изменение содержимого ЗУ из-за флуктуации питающего напряжения, кратковременных помех, воздействия α -частиц. Влияние α -частиц стало проявляться в связи с тем, что с ростом интеграции динамических ЗУ емкость элемента ЗУ уменьшается, что сопровождается снижением критического заряда, способного изменить состояния конденсатора ячейки ЗУ.

Неисправности ИС ЗУ обычно проявляются как неисправности одного бита, линии выборки разряда, линии выборки слова, обеих линий, всей ИС.

Для повышения надежности ЗУ используются корректирующие коды (КК). Наиболее распространен класс кодов с коррекцией одиночной и обнаружением двойной ошибки (КО—ОД). Самым известным среди них является код Хэмминга. В последние годы были разработаны и другие коды, являющиеся модификацией кода Хэмминга. Примером таких кодов являются коды с нечетными весами столбцов проверочной матрицы [87].

Число ошибок, возникающее в результате неисправности ИС ОЗУ, зависит от типа неисправности. Например, неисправность одной ячейки вызывает одиночную ошибку, а всей ИС ОЗУ — кратную. Обычно биты и адреса ОЗУ распределяют по ИС и ТЭЗ таким образом, чтобы ошибки можно было исправлять с помощью выбранного КК. При использовании кода КО—ОД наилучшим является размещение «один бит в одной ИС». В этом случае даже отказ всей ИС не нарушает работу ЗУ. Благодаря такому размещению код КО—ОД повышает время наработки на отказ накопительной части ЗУ до

значения, сравнимого с аналогичным показателем остальных компонентов ЗУ (дешифраторов, усилителей).

Как правило, ИС ЗУ образует часть кодового слова, как показано. При таком размещении неисправность логики обрамления накопительной части ЗУ может привести к групповой ошибке в соседних битах (кратность ошибки равна числу разрядов кодового слова, размещенного в одной ТЭЗ) [62].

Коды КО—ОД не в состоянии обнаружить подобные ошибки. Для этой цели применяются коды КО—ОД с дополнительной способностью обнаружения кратной ошибки в одной группе битов (КО—ОД—ООГ).

Если не использовано размещение ЗУ по принципу «один бит в одной ИС», то неисправность ИС ЗУ может вызвать кратную (по числу бит в ИС) ошибку. Например, если 32-битное слово ЗУ размещается в 8-битных ИС, неисправность последней может вызвать групповую ошибку в восьми соседних битах. При такой организации ЗУ используются коды с коррекцией ошибок в одной группе и обнаружением в двух группах (КОГ—ОД Г).

С ростом емкости ЗУ его надежность уменьшается и в ряде случаев оказывается целесообразным применение кода с коррекцией тройных и обнаружением одиночных ошибок (КТ—ОО).

2.5. Разработка методики алгоритмизации декодирования

На основе полученных правил кодирования формируется стратегия декодирования, решающая задачу различения ошибок в информационных и контрольных разрядах и определяющая правила коррекции возникающих ошибок, которая включает следующие пункты:

1) выявляются одинаковые дополнительные проверки (имеющие повторения), по которым из множества N исключаются синдромы ошибок, принадлежащие подмножеству n_1 (выявляются некорректируемые ошибки, для которых формируется сигнал «Отказ устройства»);

2) определяются группы одинаковых синдромов (указывающих на

ошибку в соответствующих информационных разрядах) для подмножества n_2 ;

3) определяются группы синдромов ошибок, принадлежащих подмножеству n_3 , для которых не требуется коррекция информационных разрядов;

4) выявляются группы одинаковых значений синдромов ошибок, принадлежащих подмножеству n_4 и позволяющих исправлять ошибки в соответствующих информационных разрядах.

В табл. 2.6 представлены часть значений синдромов ошибок для подмножеств n_2 , n_3 , n_4 . (исключены синдромы ошибок подмножества n_1 , имеющие одинаковые значения дополнительных проверок).

Таблица 2.6. - Значения синдромов ошибок для подмножеств n_2 , n_3 , n_4

Ошибка	Инф.разр.	Принятые КР	Сформирован. КР	Синдром
	у ₁ у ₂ у ₃ у ₄	г ₁ г ₂ г ₃ г ₄ г ₅ г ₆ г ₇ г ₈	г ₁ г ₂ г ₃ г ₄ г ₅ г ₆ г ₇ г ₈	е ₁ е ₂ е ₃ е ₄ е ₅ е ₆ е ₇ е ₈
Только в контрольных	0101	x 01011101	01010101	00001000
	0101	x 00010101	01010101	01000000
	0101	xx 00010001	01010101	01000100
	0101	xx 01111101	01010101	00101000
	0101	xxx 00100101	01010101	01110000
	0101	xxx 00011100	01010101	01001001
	Только в информационных	x 0101	01100011	01010101
x 0101		11110000	01010101	10100101
xx 0101		11000110	01010101	10010011
xx 0101		01101100	01010101	00111001
xxx 0101		10100101	01010101	11110000
xxx 0101		10011100	01010101	11001001

Ошибка	Инф.разр.	Принятые КР	Сформирован. КР	Синдром
	у ₁ у ₂ у ₃ у ₄	г ₁ г ₂ г ₃ г ₄ г ₅ г ₆ г ₇ г ₈	г ₁ г ₂ г ₃ г ₄ г ₅ г ₆ г ₇ г ₈	е ₁ е ₂ е ₃ е ₄ е ₅ е ₆ е ₇ е ₈
И в контрольных, и в информационных	x 0101	x 10110000	01010101	11100101
	x 0101	x 00111110	01010101	01101011
	x 0101	x 10001111	01010101	11011010
	xx 0101	x 10111010	01010101	11101111
	x 0101	xx 11101011	01010101	10111110
	x 0101	xx 01111011	01010101	00101110

Для рассматриваемого примера, реализующего предлагаемый метод кодирования имеем:

общее количество ошибок-4768

- число одинаковых синдромов ошибок, имеющих одинаковые дополнительные проверки (подмножество n_1)-2544 (число обнаруживаемых ошибок);
- 2224- число корректируемых ошибок (46%);
- число ошибок только в информационных разрядах- 224 ($l_1= 14$ -групп, каждая из которых включает по 16 одинаковых синдромов);
- число ошибок только в контрольных разрядах – 592 ($l_2= 37$ -групп, каждая из которых включает по 16 одинаковых синдромов);
- число ошибок, имеющих искажения одновременно в информационных и контрольных разрядах- 1408 ($l_3= 88$ -групп, каждая из которых включает по 16 одинаковых синдромов).

Вывод. Применение итеративного линейного кода, использующего проверки на четность, правые и левые диагональные проверки позволяет:

обнаруживать максимальное количество ошибок (за исключением ошибочных кодовых наборов трансформируемых в разрешенные);

корректировать ошибки трехкратные ошибки (в настоящее время неизвестны эффективные методы построения линейных кодов исправляющих больше двухкратной ошибки);

в зависимости от правила проведения дополнительных проверок, предлагаемый метод позволяет корректировать от 50% до 94% обнаруживаемых ошибок;

исправлять ошибки различной конфигурации (имеет свойства нелинейного кода) при условии обнаружения некорректируемых ошибок;

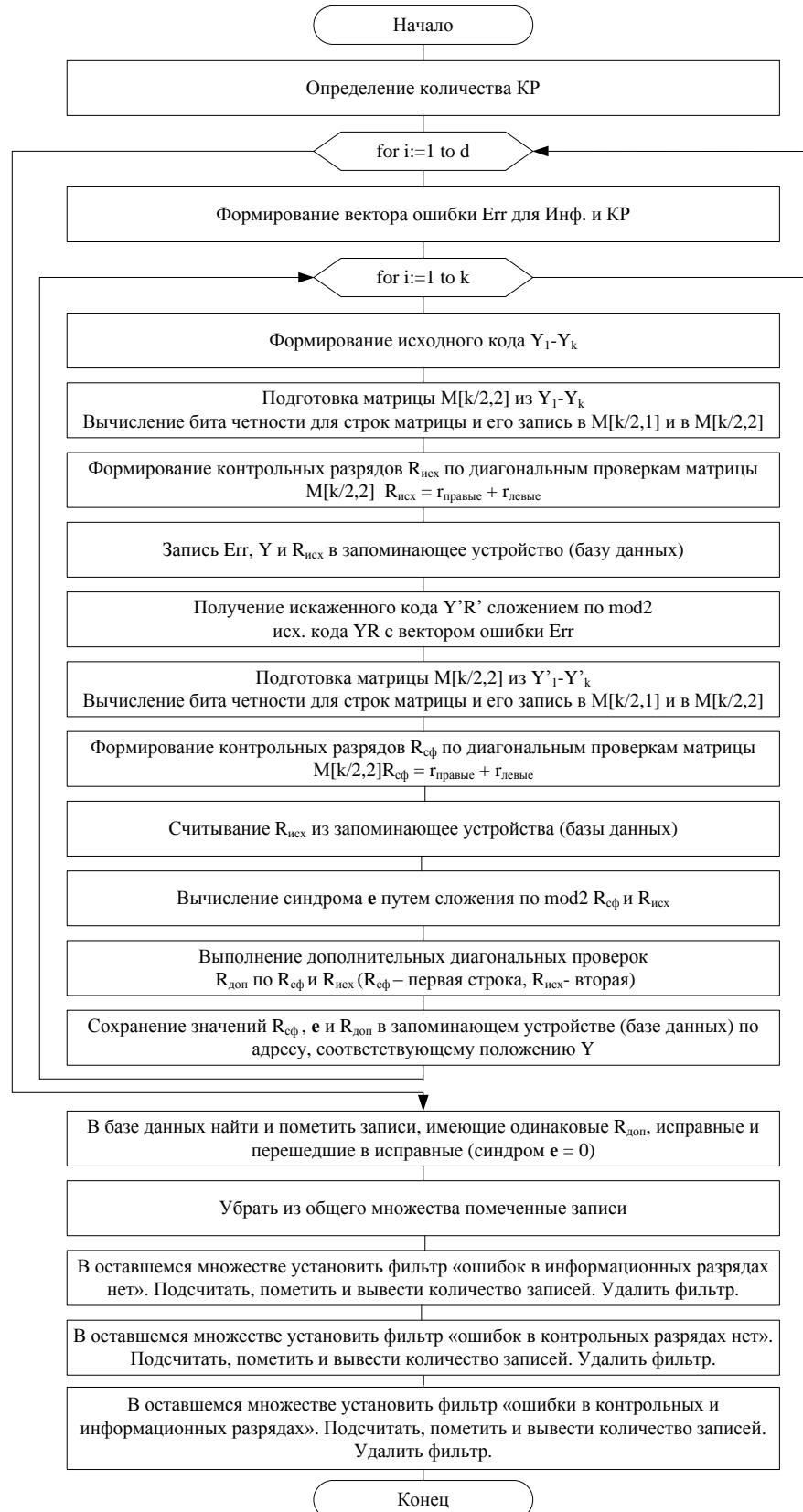
осуществлять коррекцию модульных ошибок при малом числе информационных разрядов.

2.6. Модель функционально-кодовой защиты устройства памяти

На основе приведенных алгоритмов кодирования и декодирования информации разработана программная модель защиты ПЗУ от ошибок.

Алгоритм программной модели представлен на рис.2.13. См. приложение 2.

Вариант №1



2.7. Анализ угроз информационной безопасности и основные мероприятия по их предотвращению

Чтобы обеспечить необходимую защиту информационной системы (ИС) нужно, во-первых, выявить основные угрозы безопасности, во-вторых, определить меры по предотвращению угроз, в-третьих, разработать и внедрить мероприятия защиты.

Вообще угроза представляет собой потенциально возможное событие, действие (воздействие), процесс или явление, которое может привести к нанесению ущерба чьим-либо интересам.

Чаще всего угрозы являются следствием наличия уязвимых мест в системе. Промежуток времени, пока имеется возможность использовать уязвимое место в системе до момента его устранения называется *окном опасности* [15]. Знание возможных угроз, а также уязвимых мест защиты, которые эти угрозы обычно эксплуатируют, необходимо для того, чтобы выбирать наиболее экономичные средства обеспечения безопасности.

Разнообразие потенциальных угроз информации в ИС столь велико, что не позволяет предусмотреть каждую из них, поэтому анализируемые характеристики угроз следует выбирать с позиций здравого смысла, одновременно выявляя не только сами угрозы, вероятность их осуществления, размер потенциального ущерба, но и их источники.

Все угрозы информационной безопасности (ИБ) можно классифицировать по нескольким критериям [15]:

- по аспекту ИБ (доступность, целостность, конфиденциальность);
- по компонентам ИС, на которые угрозы нацелены (программное обеспечение, аппаратное обеспечение, инфраструктура, каналы связи, пользователи);

- по способу осуществления (случайные/преднамеренные действия природного/техногенного характера);
- по расположению источника угроз (внутри/вне рассматриваемой ИС).

Рассмотрим инфраструктуру КС (рис. 2.14) и классификацию угроз по компонентам ИС, а также обозначим способы защиты от возможных угроз (таблица 2.7.) [20,82].

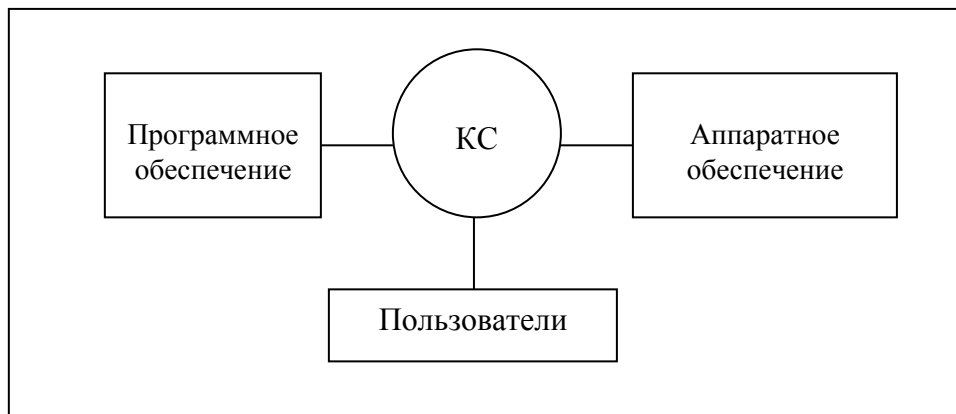


Рис. 2.14. Инфраструктура КС

Таблица 2.7. - Классификация угрозы по компонентам ИС, и способы защиты

Вид угрозы	Способы защиты
<p>1. Угрозы со стороны пользователей</p> <ul style="list-style-type: none"> – <i>непреднамеренная</i> (до 65% всех потерь) – неумение или отказ работы с системой – <i>преднамеренная</i> («обиженные сотрудники») – <i>ошибки администрирования</i> 	<ol style="list-style-type: none"> 1. обучение сотрудников, 2. введение пропускного режима и его соблюдение, 3. удаление учетных данных после увольнения сотрудника, 4. регулярная смена паролей 5. разграничение полномочий между сотрудниками, 6. соблюдение правил работы с конфиденциальной информацией

Вид угрозы	Способы защиты
<p>2. Угрозы со стороны программного обеспечения</p> <ul style="list-style-type: none"> – логические ошибки ПО – уязвимость ПО – вредоносное ПО 	<ol style="list-style-type: none"> 1. покупка и использование лицензионного программного обеспечения со встроенной защитой, 2. использование антивирусных программ, 3. использование сетевых экранов, 4. отслеживание уязвимостей и своевременное их устранение
<p>3. Угрозы со стороны аппаратного обеспечения</p> <ul style="list-style-type: none"> – разрушение или отказ АО – перебои напряжения 	<ol style="list-style-type: none"> 1. использование АО со встроенными средствами защиты, 2. бесперебойное питание, защита от скачков напряжения, 3. ремонт только в сертифицированных сервисных центрах, 4. антивандальные корпуса
<p>4. Угрозы со стороны инфраструктуры</p> <ul style="list-style-type: none"> – пожары, наводнения, стихийные бедствия, – выход из строя систем кондиционирования, охлаждения, отопления – нарушение физических границ инфраструктуры 	<ol style="list-style-type: none"> 1. защита помещений (заборы, охрана и т.п.), 2. противопожарные системы, 3. системы охлаждения и вентиляции, 4. работа с сертифицированными охранными предприятиями
<p>5. Угрозы со стороны КС</p> <ul style="list-style-type: none"> – разрывы КС – искажение при передаче в КС – использование общедоступных КС 	<ol style="list-style-type: none"> 1. шифрование при передаче в «общедоступных» КС, 2. передача с избыточным копированием (дублирование), 3. использование проверки контрольных сумм, 4. физическая защита КС

Все вышеперечисленные угрозы являются общими для всех типов ИС, но нужно отметить, что понятие «угроза» в разных ситуациях может трактоваться по-разному [15]. Например, в открытой организации (или ИС)

угрозы конфиденциальности (хищения конфиденциальной информации) может просто не существовать, однако несанкционированный доступ к этой информации или ее изменение – является существенной угрозой. Как пример можно привести, сайт Правительства РФ, где публикуются законы. Информация на этом сайте общедоступна, и не является конфиденциальной, но несанкционированный доступ к этой информации, повлекший ее изменение или удаление, является серьезной угрозой.

В любом случае, для того чтобы разработать и внедрить систему защиты ИС нужно:

1. Оценить, что является угрозами для данной системы, а что – нет.
2. Оценить экономическую эффективность защиты от определенных угроз.

3. Чтобы средства безопасности были максимально эффективными, механизмы защиты должны быть заложены на этапе проектирования ИС и аппаратных средств, а не после внедрения [34]. Внутреннюю защиту обойти сложнее, чем внешнюю.

4. В ходе планирования и реализации мер по защите информации необходимо опираться на Российские и международные стандарты «ISO/IEC 17799:2005», «URGENT ACTION Standard 1200 – Cyber security» [16].

5. Защищать информационную систему имеет смысл только комплексно, т.е. одновременно от всех угроз, как программное обеспечение, аппаратные средства, так и инфраструктуру.

Библиографический список

Выводы к главе 2

В результате проведенных исследований разработан модифицированный итеративный код, позволяющий:

7. Корректировать ошибки трехкратные ошибки в полубайте информации (в настоящее время *неизвестны эффективные методы построения линейных кодов исправляющих больше двух - кратной ошибки*), при условии обнаружения ошибок в остальных разрядах кодового набора, за исключением ошибок трансформируемых в разрешенные кодовые наборы (*новое свойство линейного кода - коррекция ошибок заданной кратности при условии обнаружения максимального количества некорректируемых ошибок*), при этом обеспечивается возможность:

8. Исправлять ошибки различной конфигурации (имеет свойства *нелинейного кода*) при условии обнаружения некорректируемых ошибок;

9. Осуществлять коррекцию модульных ошибок при малом числе информационных разрядов т.е. исключить основной недостаток кода Рида-Соломона (при исправлении ошибки в восьми разрядном модуле информации код Рида-Соломона требует 2040 информационных разрядов - поэтому исключается возможность его использования для обеспечения отказоустойчивости мало разрядных специализированных ЭВМ военного назначения);

10. Иметь минимальные временные затраты на декодирование (в отличие от кодов Рида-Соломона реализующих процедуру циклического декодирования);

11. Исключить влияние неисправного резервного оборудования на работу устройств ЭВМ при наличии ошибок в контрольных разрядах и отсутствии ошибок в информационных;

12. Сигнализировать о неисправности устройства памяти при возникновении некорректируемой ошибки;

13. Защищать информационную систему имеет смысл только комплексно.

ГЛАВА 3. МЕТОДИКИ ЗАЩИТЫ ПРОЦЕССОРА И ЭЛЕМЕНТОВ КИТС

3.1. Выбор комплекса защиты информации для корпоративных информационно-телекоммуникационных сетей

Защита информации в корпоративных информационно-телекоммуникационных сетях (КИТС) - недопущение:

- изменения (модификации) информации как частичное или значительное изменение состава и содержания сведений;
- уничтожения (разрушения) информации, как акт вандализма с целью прямого нанесения материального ущерба;
- хищения (ознакомления) конфиденциальной информации различными путями и способами без нарушения ее целостности;
- блокирования – изменения условий хранения информации, которые делают ее недоступной для пользователя.

Чтобы реализовать данные цели, подбирают специальные средства защиты (СЗ). СЗ образуют барьер одной или нескольким угрозам.

Попытка реализации угрозы (или ряда угроз) злоумышленником (тот, кто предпринимает такую попытку) называется атакой [82,83].

Цели злоумышленника – изменить, и/или уничтожить, и/или похитить, и/или заблокировать конфиденциальную информацию.

Задача противодействия данным целям формулируется следующим образом:

Построить такую систему защиты (путем подбора оптимального количества наиболее эффективных СЗ, которая смогла бы обеспечить защиту целостности, конфиденциальности, полноты и доступности информации заданным качеством.

Формализуем данную задачу.

Исходные данные

Обозначим:

$УГ = \{УГ_1, УГ_2, \dots, УГ_M\}$ – множество всех возможных угроз информации в КИТС;

$Ц = \{Ц_1, Ц_2, \dots, Ц_N\}$ – множество целей злоумышленника.

Каждая цель представляет собой совокупность угроз, например $Ц_2 = \{УГ_1, УГ_3, УГ_{10}, УГ_{11}, УГ_{12}, УГ_{13}, УГ_{14}, УГ_{16}, УГ_{17}\}$;

$A = \{A_1, A_2, \dots, A_P\}$ – множество атак. Для достижения одной и той же цели злоумышленника могут быть предприняты разные атаки, состоящие из одинаковых угроз.

Подчеркнем тот факт, что за время атаки может быть инициировано несколько однотипных угроз;

$СЗ = \{СЗ_1, СЗ_2, \dots, СЗ_R\}$ – множество средств защиты, которые способны (в различных подмножествах) обнаружить и нейтрализовать соответствующие угрозы с вероятностью P_{ij} (i – номер $СЗ$, j – номер угрозы).

Требуется: найти такое подмножество $СЗ^* \in СЗ$, чтобы атака A_P была нейтрализована.

В качестве *критерия* оптимального множества $СЗ^*$ можно взять:

- 1) минимум вероятности достижения нарушителем какой-либо или всех целей;
- 2) минимум среднего уровня потерь системы от реализации нарушителем всех целей;
- 3) максимум вероятности успешного противодействия системой с множеством $СЗ^*$ реализации всех целей нарушителем.

По первому и второму критерию задачу целесообразно решать в случае, когда основной целью системы защиты - максимальное снижение уровня несанкционированного доступа к информации (преобладание программных и аппаратных средств защиты). По третьему критерию – когда основная задача защиты состоит в максимальном возможном уровне успешного распознавания факта проникновения или действий

злоумышленника в КИТС с целью принятия дальнейших мер по противодействию ему (целесообразность использования данного критерия рекомендуется при преобладании средств физической защиты).

Кроме того, данная задача в ряде случаев может решаться по критерию минимума интегрального показателя “стоимость - риск”, выражающего совокупные затраты на организацию защиты информации и соответствующие им потери от действий злоумышленника.

По этим критериям оптимизации защиты информации в корпоративной сети задачи нахождения оптимального состава комплекса средств защиты (нахождения оптимального $CЗ^*$) могут быть сформулированы следующим образом [20,70].

Задача 1. Критерий минимум вероятности достижения злоумышленником своих целей.

Определить такие значения $x_i (i = \overline{1, R}, x_i - \text{номер } CЗ)$, что

$$P(CЗ^*) = \min_x \prod_{i=1}^N P_i^{x_i},$$

при ограничениях:

$$C(CЗ^*) \leq C_{доп};$$

$$P_i^{x_i} \leq P_i \text{ доп}$$

где $P_i^{x_i}$ - вероятность достижения злоумышленником i -й цели;

$C(CЗ^*)$ - суммарная стоимость выбранных $CЗ$;

$C_{доп}$ – максимально допустимое значение стоимости систем защиты;

$P_i \text{ доп}$ - допустимое значение вероятности реализации злоумышленником i -ой цели;

N – количество целей злоумышленника.

Задача 2. Критерий - максимум вероятности успешного противодействия системы защиты (отобранных в комплекс средств защиты) целям злоумышленника.

Определить такие значения $x_i (i = \overline{1, R}, x_i - \text{номер СЗ})$, что

$$P(CЗ^*) = \max_x \prod_{i=1}^N (1 - P_i^{3л}),$$

при ограничениях:

$$C(CЗ^*) \leq C_{дон};$$

$$P_i^{3л} \leq P_i \text{ дон}$$

Задача 3. Критерий – минимум среднего уровня потерь от реализации целей нарушителя.

Определить такие значения x_i , что

$$P(CЗ^*) = \min_x \sum_{i=1}^N P_i^{3л} * C_i$$

При ограничениях

$$C(CЗ^*) \leq C_{дон};$$

$$P_i^{3л} \leq P_i \text{ дон}$$

Здесь C_i - средняя величина потерь от реализации нарушителем i -ой цели. Складывается из потерь от нарушения конфиденциальности, от невыполнения каких-либо обязательных работ, из стоимости восстановления системы защиты при реализации злоумышленником какой-либо цели и т.п.

Сравнивая задачи 1 и 2, отметим, что они, в совокупности, близки. Задача 3 трудновыполнима в силу того, что значение C_i очень сложно определить. Поэтому остановимся на задаче 1 и упростим ее.

Пусть есть только одна цель C_i . Существует окно опасности $t_{опас}$ – промежуток времени, когда злоумышленник «атакует». Тогда задача 1 формулируется следующим образом:

Определить такие $x_i (i = \overline{1, R})$ что $P_i^{3Л} \rightarrow \min$ при ограничениях:

$$C(C3^*) \leq C_{дон}, P_i^{3Л} \leq P_{идоп}(t_{окна})$$

Решение.

А. Определение $P_i^{3Л}$.

Рассмотрим пример.

Пусть злоумышленник попытается реализовать цель C_1 , состоящую из трех угроз $УГ_1, УГ_2, УГ_3$, инициирую атаки в течение $t_{опас}$. За это время, возможно будет инициирован выделенный набор из трех угроз ($УГ_1, УГ_2, УГ_3$) по несколько раз.

Построим граф переходов (рис.3.1.1)

Дуги обозначаются $P_{a_i}/УГ_j$ и имеют смысл вероятности перехода из состояния a_i в другое под действием j -ой угрозы. «Ждущая» вершина имеет смысл изменения вероятности нахождения в данном состоянии от последовательности однотипных угроз.

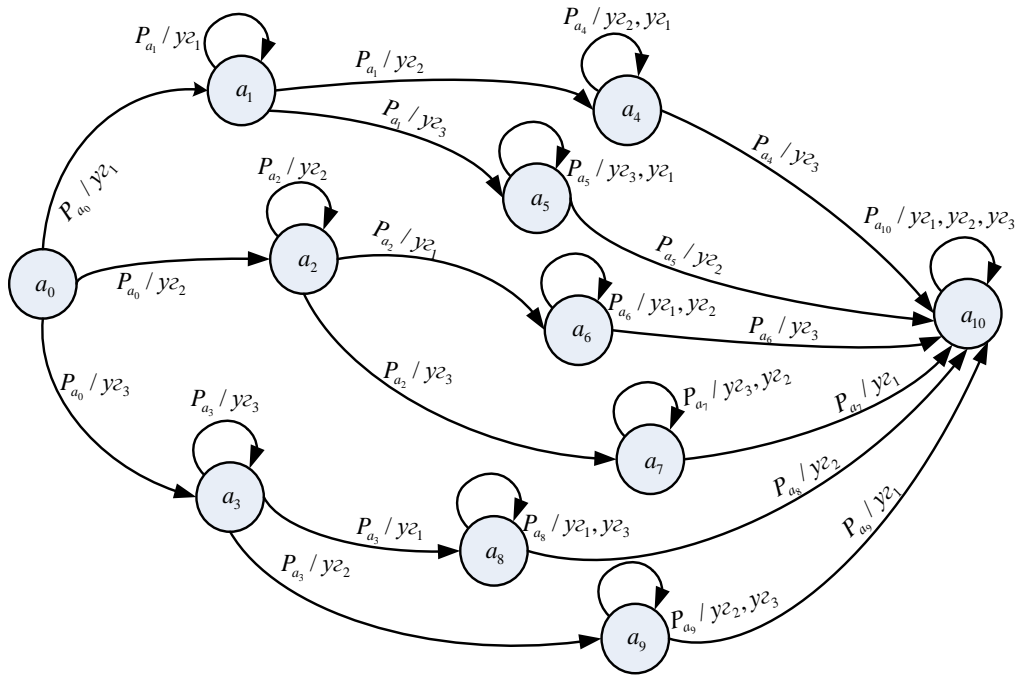


Рис.3.1.1. Определение $P_i^{3Л}$. Граф переходов

Для определения P_{ai} рассмотрим типовую конструкцию (элемент) графа (рис. 3.1.2).

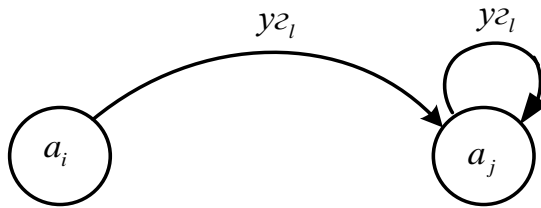


Рис.3.1.2. Типовой элемент графа переходов

Если положить, что каждая угроза реализует различное количество раз, то аналитическая зависимость для определения $P_i^{3Л}$ становится очень громоздкой. Требуется упрощения для практического применения данного подхода.

Отметим факт, что атака может быть, например, такой: $(УГ_1, УГ_1, УГ_2, УГ_2, УГ_3, УГ_3)$ или такой: $(УГ_1, УГ_2, УГ_3, УГ_1, УГ_2, УГ_3)$, т.е. разной по времени

возникновения угроз, роли не играет, достаточно выполнить перерасчет на большее количество угроз в соответствующем состоянии (рис. 3.1.3).

Для дальнейшего рассмотрения положим, что угрозы в атаке реализуются пакетами. Для нашего примера атака выглядит следующим образом.

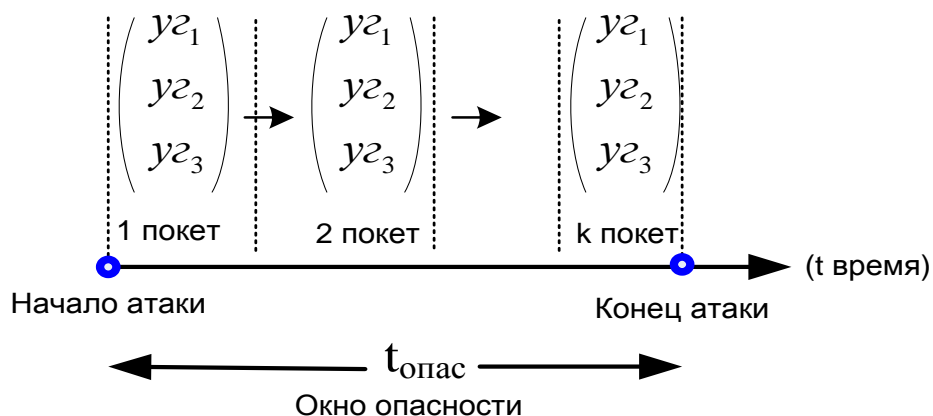


Рис.3.1.3. Атака пакетами угроз

С таким упрощением граф переходов будет выглядеть следующим образом (рис.3.1.4).

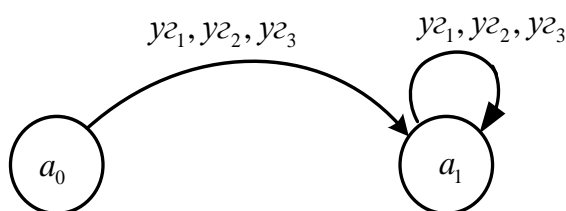


Рис. 3.1.4. Упрощенный граф переходов

В данной модели $P_{a_0} = 0$

Обобщенный алгоритм поиска оптимального состава СЗ, противодействующего атаке злоумышленника при реализации его конкретной цели в КИТС приведен на рис.3.1.5.



Рис. 3.1.5. Схема алгоритма определения состава комплекса средств защиты информации в КИТС

В качестве экспериментальной проверки рассмотрим подбор состава комплекса средств защиты.

В качестве защищаемой автоматизированной биллинговой системы (АБС) рассматривалась система низшего уровня, состоящая из рабочей станции, сервера и аппаратуры передачи данных. Цели злоумышленника, перечень угроз информации соответствуют рассмотренным выше. Наибольшую опасность с точки зрения наносимого ущерба несут угрозы, связанные с нарушением процесса функционирования системы, что требует акцентирования большего внимания на их нейтрализацию. В качестве характеристик средств защиты, на основании которых осуществлялся их выбор, использовались стоимость средства и вероятность успешного функционирования по нейтрализации соответствующей угрозы. Моделирование проводилось для различных значений ограничения на стоимость системы защиты под управлением ОС MS Windows XP, на ПК.

Рассматривался подбор комплекса защитных средств, нейтрализующих все цели злоумышленника. Предположительно атака осуществляется четырехкратным повторением комплекса угроз ($k=4$). Время решения программы составило 40 с. Результаты моделирования представлены в табл. 3.1.1 и на рис. 3.1.6,7.

Таблица 3.1 - Результаты моделирования

Подбор комплекса защитных средств, нейтрализующих цель №3				Подбор комплекса защитных средств, нейтрализующих все цели			
№ набора СЗ	Номера средств защиты набора	Общая стоимость комплекса средств защиты	Вероятность достижения цели	№ варианта	Номера средств защиты набора	Общая стоимость комплекса средств защиты	Вероятность реализации злоумышленником всех целей
1	3, 4, 6	2285,00	4,41E-01	1	1, 3, 5, 7, 8	6939,00	9,41E-04
2	3, 6, 7	3224,00	1,12E-02	2	1, 3, 7, 8, 9	5529,00	2,75E-03
3	3, 6, 5	3285,00	4,47E-02	3	1, 3, 4, 7,	5939,00	3,93E-04

					8		
4	3, 6	1785,00	8,10E-01	4	1, 6, 10	4065,00	5,89E-03
5	3, 6, 8	2785,00	8,29E-02	5	1, 3, 5, 6	5785,00	4,18E-04
6	3, 6, 9	1875,00	5,65E-01	6	2, 3, 5, 7, 8	4439,00	2,02E-03
7	3, 7, 8	2689,00	1,90E-01	7	2, 3, 7, 8, 9	3029,00	2,24E-02
8	3, 4, 7, 8	3189,00	1,12E-02	8	2, 3, 4, 7, 8	3439,00	1,34E-02
9	3, 5, 7, 8	4189,00	2,70E-02	9	2, 6, 10	1815,00	4,80E-02
10	3, 7, 8, 9	2779,00	1,51E-01	10	2, 3, 5, 6	3535,00	9,92E-03
11	3, 4, 7, 8, 9	3279,00	7,84E-03	11	3, 5, 7, 8	4189,00	4,90E-03
				12	3, 5, 7, 8, 9	4279,00	9,43E-03
				13	3, 4, 5, 7, 8	4689,00	1,34E-03
				14	3, 5, 6	3285,00	6,98E-03



Рис. 3.1.6. Зависимость вероятности достижения цели №3 злоумышленником от стоимости комплекса средств защиты



Рис. 3.1.7. Зависимость вероятности достижения всех целей злоумышленником от стоимости комплекса средств защиты

Из результатов экспериментов видно, что с увеличением объема ассигнований на средства защиты в целом вероятность реализации злоумышленником всех целей значительно снижается. Причем, данная зависимость носит явно выраженный экспоненциальный с отрицательным коэффициентом характер. Это общая тенденция. Тем не менее, в отдельных случаях стоимость средств не показатель снижения вероятности реализации злоумышленником всех целей, например, включение в состав комплекса дорогого средства защиты №7, нейтрализующего многие из угроз, нежелательно из-за низкой эффективности блокирования этих угроз. «Выигрывает», как правило, комплекс, состоящий из многих недорогих средств защиты, специализирующихся на угрозах определенного вида.

3.2. Обеспечение отказоустойчивости сумматора на основе корректирующих линейных кодов

Преимущественное распространение в современных вычислительных машинах имеют сумматоры параллельного действия, в которых передача чисел и образование суммы происходит одновременно для всех разрядов.

Исследуем возможность использования линейных кодов для обеспечения отказоустойчивости сумматора на примере операции сложения [9,33].

Допустим, требуется сложить два четырех разрядных числа: $A=0111$ и $B=0011$, которые хранятся в отказоустойчивом ОЗУ и кодируются на основе первого метода функционально-кодовой защиты информации.

В этом случае, информационная матрица для числа A имеет вид:

$$A_M \Rightarrow \begin{array}{r} 011 \\ 110, \end{array}$$

правый столбец которой, содержит значения разрядов проверки на четность строк информационной матрицы.

Соответственно правые и левые диагональные проверки информационной матрицы дадут результат:

$$0000\ 1111.$$

Таким образом, из ОЗУ в операционное устройство процессора число A поступает в виде: $0111\ 0000\ 1111$.

Информационная матрица числа B имеет вид:

$$B_M \Rightarrow \begin{array}{r} 000 \\ 110, \end{array}$$

при этом число B поступает на вход сумматора в виде:

$$0011\ 01101100.$$

Арифметическое суммирование информационных разрядов даст результат:

$$S = \begin{array}{r} 0111 \\ \underline{0011} \\ 1010 \end{array}$$

Информационная матрица S_M имеет вид:

$$S_M \Rightarrow \begin{array}{r} 101 \\ 101, \end{array}$$

соответственно правые и левые диагональные проверки информационной матрицы дадут значения сигналов в контрольных разрядах: 11111111.

Таким образом, в результате выполнения операции сложения на выходе арифметического устройства должен быть сформирован кодовый набор: 1010 11111111. Однако, арифметическое сложение контрольных разрядов слагаемых даст результат:

$$\begin{array}{r} 00001111 \\ + 01101100 \\ \hline S_{k+} = 01111011 \end{array}$$

который, отличается от правильного значения: 11111111.

Аналогичным образом, сложение контрольных разрядов слагаемых по mod 2 даст результат:

$$\begin{array}{r} 00001111 \\ + 01101100 \\ \hline S_{k \bmod 2} = 01100011, \end{array}$$

который тоже отличается от правильного значения.

Таким образом при тривиальном подходе выполнения операций в АУ процессора, контрольные разряды кодовых наборов (операндов) не могут быть использованы для исправления возникающих ошибок.

В связи с этим возникает необходимость адаптации значений контрольных разрядов операндов к выполнению различных операций процессором.

3.3. Разработка подхода обнаружения и коррекции ошибок арифметических операций функционального ядра КСОН

Для формирования “правильных” значений контрольных разрядов возникает необходимость определения правил формирования поправки к значению контрольных разрядов, полученных в результате выполнения арифметической операции S_{k+} .

Правила формирования поправки, может быть получено на основе построения информационных матриц, учитывающих перенос единицы в старший разряд, при наличии единиц в одноименных разрядах.

Свойство 3.1. При одновременном наличии единиц в младших информационных разрядах слагаемых и отсутствии цепи переноса в остальных разрядах, значение поправки равно: $C_1=01011010$, которое получается в результате арифметического сложения чисел $A_1 =0001$, $B_1 =0001$, формирования информационной матрицы из полученного результата 0010[33]:

$$A_{M1} \Rightarrow \begin{array}{r} 00\ 0 \\ 10\ 1 \end{array}$$

и кодирования данной матрицы путем проведения правых и левых диагональных проверок: 01011010.

Свойство 3.2. При одновременном наличии единиц во вторых разрядах слагаемых и отсутствии цепи переноса (наличии единиц в одноименных разрядах слагаемых или наличии единицы в рассматриваемом разряде одного из слагаемых и поступлении переноса из младшего разряда в данный разряд) в остальных разрядах, поправка формируется на основе информационной матрицы:

$$A_{M2} \Rightarrow \begin{array}{r} 01\ 1 \\ 00\ 0 \end{array}$$

и равна: $C_2=01100011$.

Свойство 3.3. При одновременном наличии единиц в третьих разрядах и отсутствии цепи переноса в остальных разрядах, поправка формируется на основе информационной матрицы:

$$A_{M3} \Rightarrow \begin{array}{r} 10\ 1 \\ 00\ 0 \end{array}$$

и равна: $C_3=10100101$.

Для рассматриваемого примера, при одновременном наличии единиц в четвертых разрядах слагаемых значение поправки не приводится, так как наличие единиц в четвертых разрядах слагаемых приводит к переполнению разрядной сетки.

Свойство 3.4. При наличии цепи переноса в первых и вторых разрядах слагаемых значение поправки равно:

$$C_4 = C_1 \oplus C_2 = \oplus \begin{array}{r} 01011010 \\ 01100011 \\ \hline 00111001 \end{array},$$

или получается на основе информационной матрицы A_{M4} , сформированной на основе сложения по mod2 одноименных разрядов информационных матриц A_{M1} и A_{M2} :

$$A_{M4} \Rightarrow \begin{array}{r} 01 \ 1 \\ 10 \ 1 \end{array}$$

Свойство 3.5. При наличии цепи переноса в первых и третьих разрядах слагаемых значение поправки равно:

$$C_5 = C_1 \oplus C_3 = \oplus \begin{array}{r} 01011010 \\ 10100101 \\ \hline 11111111 \end{array},$$

или получается на основе информационной матрицы A_{M5} , сформированной на основе сложения по mod2 одноименных разрядов информационных матриц A_{M1} и A_{M3} :

$$A_{M5} \Rightarrow \begin{array}{r} 10 \ 1 \\ 10 \ 1 \end{array}$$

Свойство 3. 6. При наличии цепи переноса во вторых и третьих разрядах слагаемых значение поправки равно:

$$C_6 = C_2 \oplus C_3 = \oplus \begin{array}{r} 01100011 \\ 10100101 \\ \hline \end{array}$$

11000110 ,

или на основе информационных матриц A_{M2} и A_{M3} :

Свойство 3.7. При наличии цепи переноса в первых, вторых и третьих разрядах слагаемых значение поправки равно:

$$\begin{array}{r} 01011010 \\ \oplus 01100011 \\ C_7=C_1 \oplus C_2 \oplus C_3 \ 10100101 \\ \hline 10011100, \end{array}$$

или на основе информационных матриц A_{M1} , A_{M2} и A_{M3} .

Свойство 3.8. При отсутствии цепи переноса в информационных разрядах (отсутствии единиц в одноименных разрядах слагаемых) значение поправки равно: 0000 0000.

Для рассматриваемого примера (сложения чисел $A=0111$ и $B=0011$) требуется поправка $C_7=1001\ 1100$.

Тогда сложение по mod2 значений контрольных разрядов данных слагаемых и поправки даст результат:

$$\begin{array}{r} 01100011 \\ \oplus 10011100 \\ S_{k \bmod 2} \oplus C_7 = 11111111 \end{array}$$

который, соответствует правильному значению контрольных разрядов для результата суммы $S=1010$, полученного в результате арифметического сложения чисел $A=0111$ и $B=0011$.

Предлагаемый способ позволяет контролировать работу сумматора АЛУ и при этом позволяет:

- обнаруживать максимальное количество ошибок (за исключением ошибок, трансформируемых в разрешенные кодовые наборы);
- корректировать ошибочную работу сумматора (информационных разрядов) по сформированным значениям контрольных разрядов (кратность исправляемой ошибки до $k-1$) по правилам исправления ошибок ЗУ;

- обнаруживать и исправлять ошибки произвольной конфигурации в контрольных разрядах (передаются контрольные разряды, полученные в результате кодирования результата суммы);
- сигнализировать о возникновении некорректируемых ошибок.

3.4. Разработка функционально-кодовой защиты процессора при выполнении логических операций

3.4.1. Разработка способа коррекции ошибок при выполнении операции сложения по mod 2

Рассмотрим основные положения способа контроля операции сложения по mod2 на примере.

Допустим, требуется сложить два четырех разрядных числа:

$A=0101$ и $B=0011$, которые хранятся в отказоустойчивом ОЗУ, т.е. кодируется рассмотренным способом.

В этом случае из ОЗУ в операционное устройство процессора число A поступает в виде кодового набора: 0101 0101 0101, а число B поступает в виде: 0011 0110 1100.

При выполнении операции сложения по mod2 информационных и контрольных разрядов рассматриваемых слагаемых получим кодовый набор 0110 0011 1001.

Информационная матрица полученного результата имеет вид:

01 1

10 1

правые и левые диагональные проверки данной матрицы дадут результаты: 00111001, что позволяет сформулировать свойство 10.

Свойство 3.9. Результат сложения по mod 2 контрольных разрядов слагаемых соответствует результату сложения по mod 2 информационных разрядов рассматриваемых слагаемых.

Данное свойство позволяет контролировать операцию сложения по mod2 и при этом обнаруживать и исправлять, возникающие ошибки по правилам предлагаемого метода кодирования.

3.4.2. Разработка способа коррекции ошибок при выполнении операции сдвига

Рассмотрим основные положения способа контроля операции сдвига на примере кодового набора: 0111 0000 1111.

Пусть требуется провести операцию сдвига вправо на один разряд, в результате получим кодовый набор: 0011.

Информационная матрица полученного результата имеет вид:

$$\begin{array}{c} 00\ 0 \\ 11\ 0 \end{array}$$

Правые и левые диагональные проверки данной матрицы дадут результата: 01101100, который отличается от исходного набора контрольных разрядов 00001111.

В связи с этим возникает необходимость формирования поправки к исходному кодовому набору, позволяющей получить набор контрольных разрядов соответствующий значению информационных разрядов, полученных при сдвиге вправо. С этой целью построим матрицу поправок при сдвиге вправо, разряды которой формируются следующим образом: $r_4=0\oplus u_4$; (0 если в старший разряд не переносится единица из другого регистра в противном случае $r_4=y_1^i \oplus u_4$, где y_1^i значение сигнала переноса из другого регистра, например в старший разряд регистра дополнительного из младшего разряда регистра сумматора при выполнении операции умножения) $r_3=y_4\oplus u_3$; $r_2=y_3\oplus u_2$; $r_1=y_2\oplus u_1$.

Для рассматриваемого примера получим кодовый набор: 0100.

В этом случае матрица поправок при сдвиге вправо имеет вид:

$$\begin{array}{c} 01\ 1 \\ 00\ 0. \end{array}$$

При кодировании данной матрицы предлагаемым методом получим значение поправки: 01100011.

Сложение по mod2 исходного значения контрольных разрядов с значением поправки даст правильное значение контрольных разрядов при сдвиге информационных разрядов вправо:

$$\begin{array}{r} 00001111 \\ \oplus 01100011 \\ \hline 01101100. \end{array}$$

При сдвиге информационных разрядов влево разряды матрицы поправок формируются следующим образом: $r_4 = y_4 \oplus u_3$; $r_3 = y_3 \oplus u_2$; $r_2 = y_2 \oplus u_1$; $r_1 = 0 \oplus u_1$

Для рассматриваемого примера получим значения разрядов матрицы поправок при сдвиге влево: 1001, соответственно матрица поправок при сдвиге вправо имеет вид:

$$\begin{array}{r} 101 \\ 011. \end{array}$$

Правые и левые диагональные проверки данной матрицы дадут результат: 10010011.

При сложении по mod2 исходного набора контрольных разрядов и значения поправки получим результат:

$$\begin{array}{r} 00001111 \\ \oplus 10010011 \\ \hline 10011100, \end{array}$$

который соответствует значению информационных разрядов сдвинутых влево.

Действительно при сдвиге информационных разрядов рассматриваемого примера имеем: 1110.

Соответственно матрица информационных разрядов имеет вид:

110

11 0

10 1,

для которой имеем набор значений контрольных разрядов: 10011100.

Свойство 3.10. Сложение по mod2 исходного значения контрольных разрядов с значением поправки при сдвиге вправо (влево) даст правильное значение контрольных разрядов при сдвиге информационных разрядов.

Данное свойство позволяет контролировать операцию сдвига и при этом обнаруживать и исправлять, возникающие ошибки по правилам предлагаемого метода кодирования.

3.4.3 Разработка способа коррекции ошибок при выполнении логической операции ИЛИ

Рассмотрим основные положения способа контроля операции ИЛИ на примере.

Допустим, требуется выполнить логическую операцию ИЛИ относительно двух четырех разрядных числа:

$A=0101$ и $B=0011$, которые хранятся в отказоустойчивом ОЗУ, т.е. кодируется рассмотренным способом.

В этом случае из ОЗУ в операционное устройство процессора число A поступает в виде кодового набора: 0101 01010101, а число B поступает в виде: 0011 01101100.

При выполнении операции ИЛИ для информационных разрядов и сложения по mod2 контрольных разрядов рассматриваемых слагаемых получим кодовый набор 0111 00111001, у которого значения контрольных разрядов отличаются от правильного набора контрольных разрядов 00001111 для полученного значения информационных разрядов 0111.

Для формирования поправки построим матрицу поправок, используя операцию логическую И относительно информационных разрядов

рассматриваемых операндов, в результате получим кодовый набор: 0001(операция И выполняется для младших разрядов операндов).

В этом случае матрица поправок операции ИЛИ имеет вид:

$$00\ 0$$

$$01\ 1$$

правые и левые диагональные проверки данной матрицы дадут результат поправки: 00110110.

Свойство 3. 11. Операция сложения по mod2 полученных значений контрольных разрядов и значения поправки, сформированной на основе матрицы поправок для логической операции ИЛИ, даст правильное значение контрольных разрядов.

Действительно, для рассматриваемого примера имеем результат:

$$\begin{array}{r} 00111001 \\ \oplus 00110110 \\ \hline 00001111, \end{array}$$

который является правильным для кодового набора 0111, полученным в результате логической операции ИЛИ, что позволяет контролировать операцию ИЛИ и при этом обнаруживать и исправлять, возникающие ошибки по правилам предлагаемого метода кодирования.

3.4.4 Разработка способа коррекции ошибок при выполнении логической операции И

Рассмотрим основные положения способа контроля операции И на примере.

Допустим, требуется выполнить логическую операцию И для двух четырех разрядных числа: $A=0101$ и $B=0011$, которые хранятся в отказоустойчивом ОЗУ, т.е. кодируется рассмотренным способом.

В этом случае из ОЗУ в операционное устройство процессора число A поступает в виде кодового набора: 0101 01010101, а число B поступает в виде: 0011 01101100.

При выполнении операции И для информационных разрядов и сложения по mod2 контрольных разрядов рассматриваемых операндов получим кодовый набор 0001 00111001, у которого значения контрольных разрядов отличаются от правильного набора контрольных разрядов 00110110 для полученного значения информационных разрядов 0001.

Для формирования поправки построим матрицу поправок, используя операцию логическую ИЛИ относительно информационных разрядов рассматриваемых операндов, в результате получим кодовый набор: 0111.

В этом случае матрица поправок операции И имеет вид:

$$\begin{array}{r} 01\ 1 \\ 11\ 0 \end{array}$$

правые и левые диагональные проверки данной матрицы дадут результат поправки: 00001111.

Свойство 3.12. Операция сложения по mod2 полученных значений контрольных разрядов и значения поправки, сформированной на основе матрицы поправок для логической операции И, даст правильное значение контрольных разрядов.

Действительно, для рассматриваемого примера имеем результат:

$$\begin{array}{r} 00111001 \\ \oplus 00001111 \\ \hline 00110110, \end{array}$$

который является правильным для кодового набора 0001, полученным в результате логической операции И, что позволяет контролировать операцию И при этом обнаруживать и исправлять, возникающие ошибки по правилам предлагаемого метода кодирования.

3.4.5. Разработка методики коррекции ошибок при выполнении логической операции НЕ

Рассмотрим основные положения способа контроля операции инверсии на примере.

Допустим, требуется выполнить логическую операцию НЕ для четырех разрядного числа: $A=0101$ которое, хранятся в отказоустойчивом ОЗУ и, кодируется рассмотренным методом.

В этом случае из ОЗУ в операционное устройство процессора число A поступает в виде кодового набора: 0101 01010101.

Информационная матрица исходного числа A имеет вид:

01 1

01 1

При выполнении операции НЕ для информационных разрядов получим кодовый набор 1010 01010101, у которого значения контрольных разрядов отличаются от правильного набора контрольных разрядов 11111111 для полученного значения информационных разрядов 1010.

Соответственно информационная матрица, построенная относительно инверсного значения числа \bar{A} имеет вид:

10 1

10 1

Для формирования поправки используем матрицу поправок для логической операции НЕ, которая получена путем сложения по mod2 одноименных разрядов исходной информационной матрицы и информационной матрицы построенной относительно инверсного значения исходного числа. В результате имеем:

11 0

11 0.

Примечание: Крайний правый столбец матрицы поправок для логической операции НЕ представляет собой результат сложения по mod2 разрядов проверки на четность исходной и инверсной информационных матриц. Для матриц, содержащих четыре информационных разряда (в данном случае) данный столбец всегда имеет нулевые значения, что не характерно для матриц большей размерности. В то же время остальные столбцы рассматриваемой матрицы в своих разрядах всегда имеют только единичные значения.

Правые и левые диагональные проверки данной матрицы дадут результат поправки: 10101010.

Свойство 3.13. Операция сложения по mod2 полученных значений контрольных разрядов и значения поправки, сформированной на основе матрицы поправок для логической операции НЕ, даст правильное значение контрольных разрядов.

Действительно, для рассматриваемого примера имеем результат:

$$\begin{array}{r} 01010101 \\ \oplus 10101010 \\ \hline 11111111, \end{array}$$

который является правильным для кодового набора 1010, полученным в результате логической операции НЕ, что позволяет контролировать операцию НЕ и при этом обнаруживать и исправлять, возникающие ошибки по правилам предлагаемого метода кодирования.

3.5. Разработка функциональной схемы отказоустойчивого процессора повышенной достоверности функционирования

На Рис.3.5.1 представлена функциональная схема отказоустойчивого процессора; на Рис.3.5.2- функциональная схема блока коррекции; на Рис.3.5.3- функциональная схема блока контроля; на рис.3.5.4- функциональная схема формирования поправки при выполнении арифметических операций; на Рис.3.5.5-функциональная схема формирования поправки при выполнении операции сдвига; на Рис.3.5.6- функциональная схема формирования поправки при выполнении операции ИЛИ; на Рис.3.5.7- функциональная схема формирования поправки при выполнении операции И; на Рис.3.5.8- функциональная схема формирования поправки при выполнении операции НЕ.

Процессор (Рис.3.5.1) содержит управляющий узел 1, операционный узел 2, дешифратор 3 кода операции, генератор 4 тактовых импульсов, блок 5

управления, первый коммутатор 6, второй коммутатор 7, третий коммутатор 8, счетчик 9 команд, счетчик 10 сдвигов, регистр 11 адреса, регистр 12 числа, регистр 13 сумматора, регистр 14 дополнительный, регистр 15 дополнительного кода, сумматор 16, блок 17 коррекции, блок 18 контроля, управляющая память 19, первые входы 20 блока 17 коррекции, вторые входы 21 блока 17 коррекции, первые входы 21 блока 17 коррекции, первые выходы 22 блока 17 коррекции, вторые выходы 23 блока 17 коррекции, первые входы 24^I блока 18 контроля, вторые входы 24 блока 18 контроля, третьи входы 25 блока 18 контроля, четвертые входы 26 блока 18 контроля, первые выходы 27 блока 18 контроля, вторые выходы 28 блока 18 контроля, третий выход 29 блока 18 контроля, (выход 28 -ошибка некорректируемая ошибка, выход 29-корректируемая ошибка), выходы 30 устройства обмена, выходы 31 запоминающего устройства, вход 30 с устройства обмена, выход 31 запоминающего устройства, выходы 32 на устройство обмена, выходы 33 для формирования адреса запоминающего устройства, выходы 34 для считывания операндов на запоминающее устройство, выходы 35 синхроимпульсов, выходы 36 для сигналов управления, выходы 37 для команды считывание, выходы 38 для сигналов записи, выходы 39 для сигналов установки в нулевое состояние.

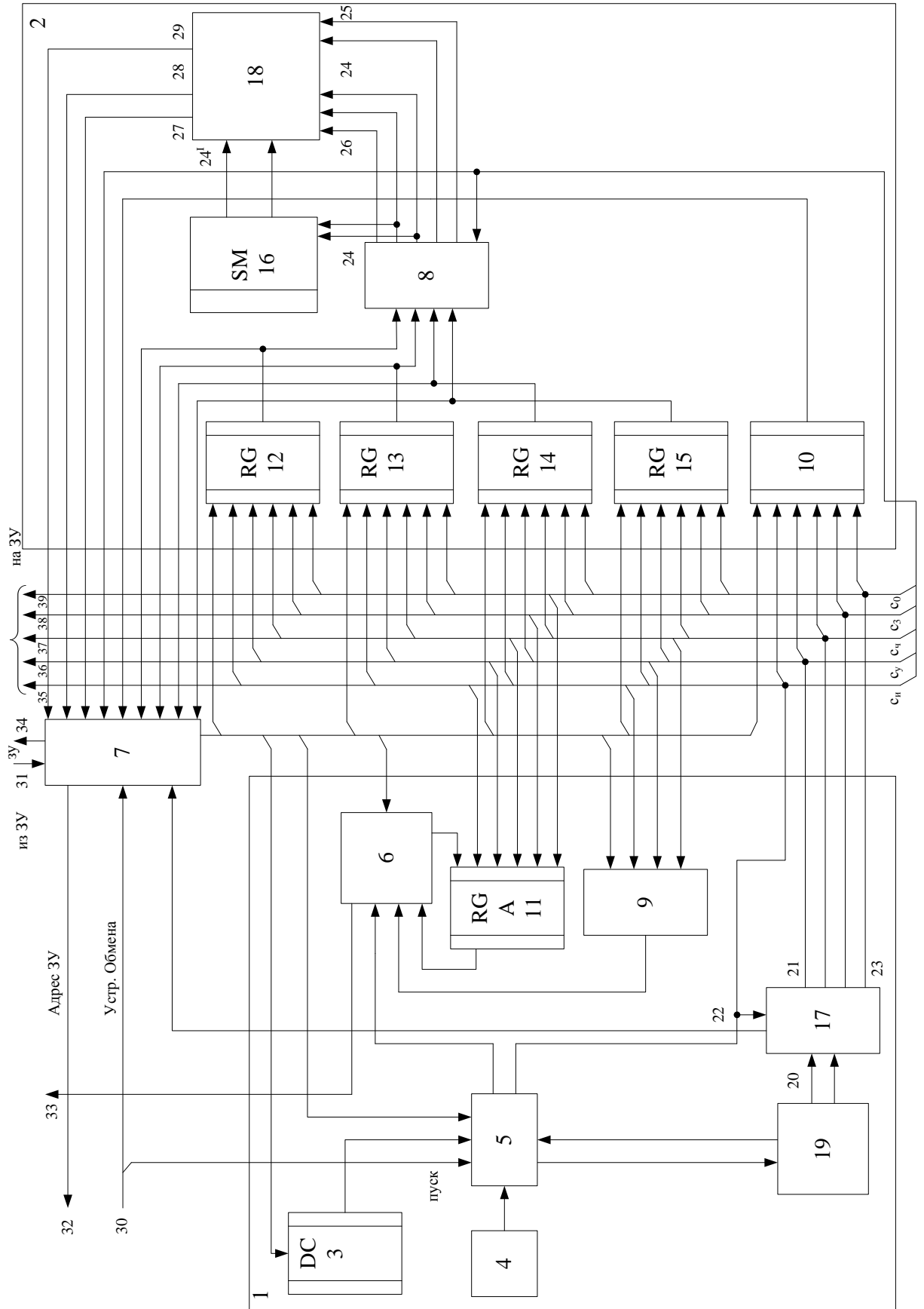


Рис. 3.5.1 - Функциональная схема отказоустойчивого процессора

Блок 17 коррекции (Рис.3.5.2) содержит кодирующую схему 40, схему 41 сравнения, первую группу 42 элементов ИЛИ, вторую группу 43 элементов ИЛИ, элемент 44 НЕ, элемент 45 И, формирователь 46 вектора ошибки, корректор 47, дешифратор 48, первую группу 49 элементов И, выходы 20 управляющей памяти 19, вход 21 разрешающий считывание выходной информации, выходы 22 “Отказ процессора”, выходы 23 для сигналов управления функциональными узлами процессора.

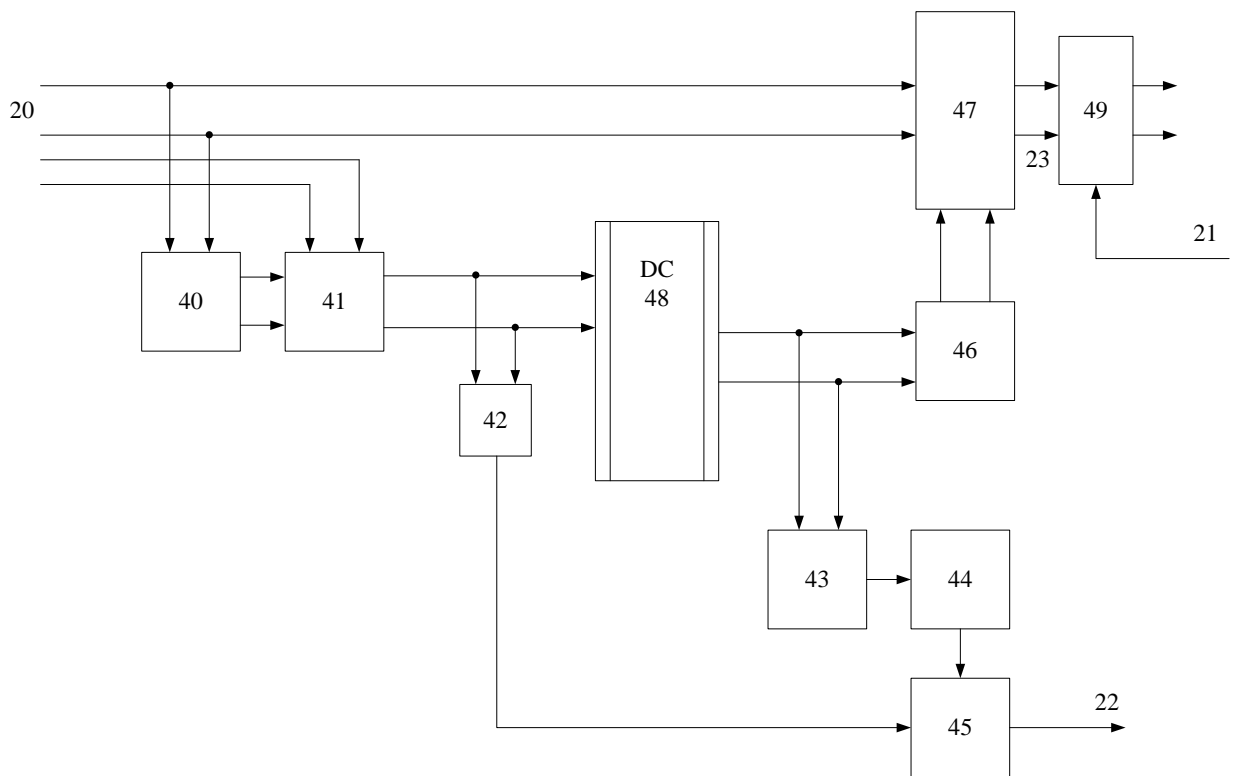


Рис. 3.5.2 - Функциональная схема блока коррекции

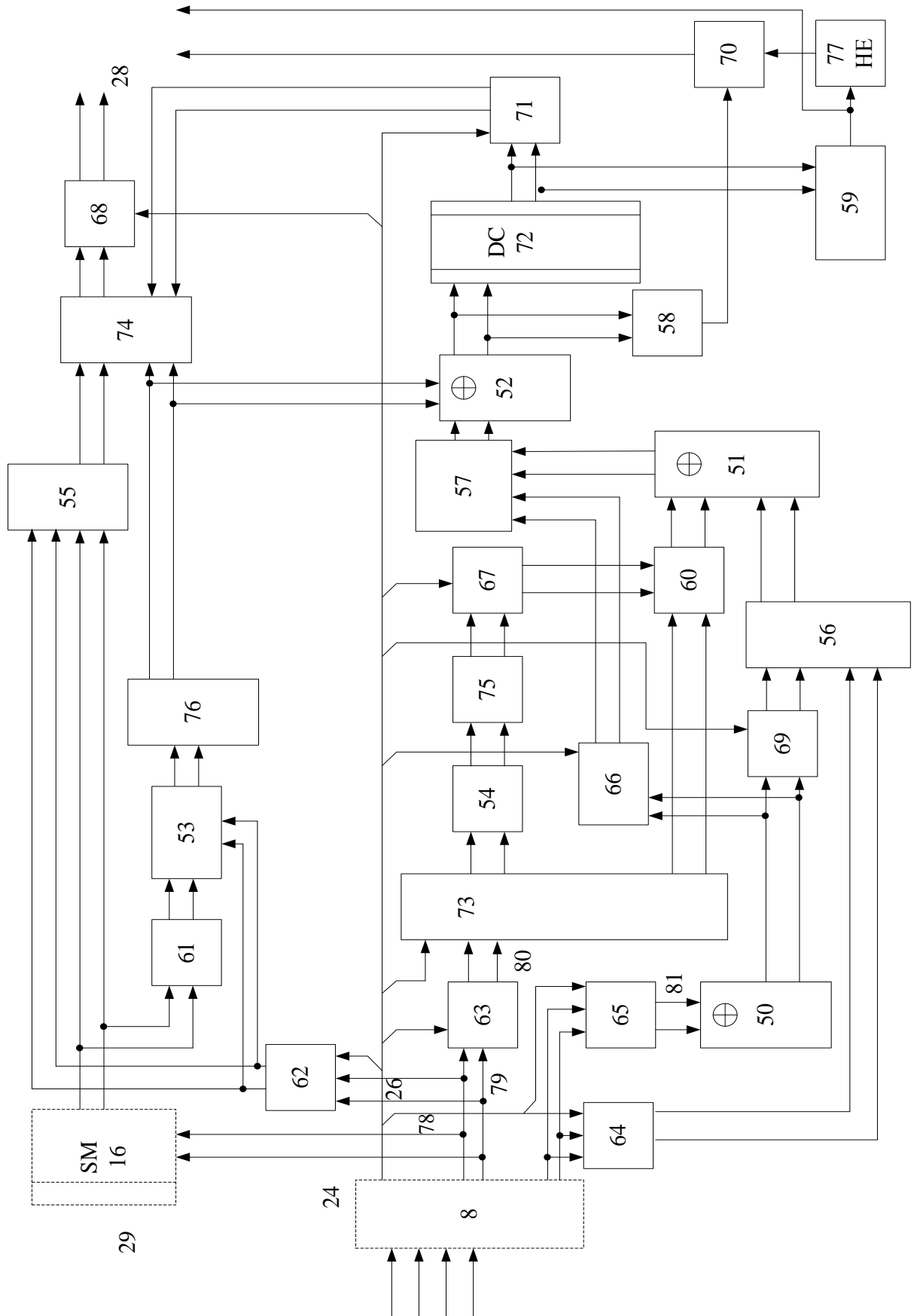


Рис. 3.5.3 - Функциональная схема блока контроля

Блок 18 контроля (Рис.3.5. 3) содержит первую группу 50 элементов неравнозначности, вторую группу 51 элементов неравнозначности, третью группу 52 элементов неравнозначности, первую группу 53 элементов ИЛИ, вторую группу 54 элементов ИЛИ, третью группу 55 элементов ИЛИ, четвертую группу 56 элементов ИЛИ, пятую группу 57 элементов ИЛИ, шестую группу 58 элементов ИЛИ, седьмую группу 59 элементов ИЛИ, восьмую группу 60 элементов ИЛИ, первую группу 61 элементов И, вторую группу 62 элементов И, третью группу 63 элементов И, четвертую группу 64 элементов И, пятую группу 65 элементов И, шестую группу 66 элементов И, седьмую группу 67 элементов И, восьмую группу 68 элементов И, девятую группу 69 элементов И, элемент 70 И, формирователь 71 вектора ошибки, дешифратор 72, формирователь 73 поправки, который содержит функциональную схему формирования поправки при выполнении арифметических операций, функциональную схему формирования поправки при выполнении операции сдвига, функциональную схему формирования поправки при выполнении операции ИЛИ, функциональную схему формирования поправки при выполнении операции И, функциональную схему формирования поправки при выполнении операции НЕ, корректор 74, первую кодирующую схему 75, вторую кодирующую схему 76, элемент 77 НЕ, первые входы 78 формирователя поправки 72, вторые входы 79 формирователя 73 поправки, выходы 80 формирователя 73 поправки (значения поправки при выполнении операций сдвига, И, ИЛИ, НЕ), выходы 81 формирователя поправки 73 (значения поправки при выполнении арифметических операций).

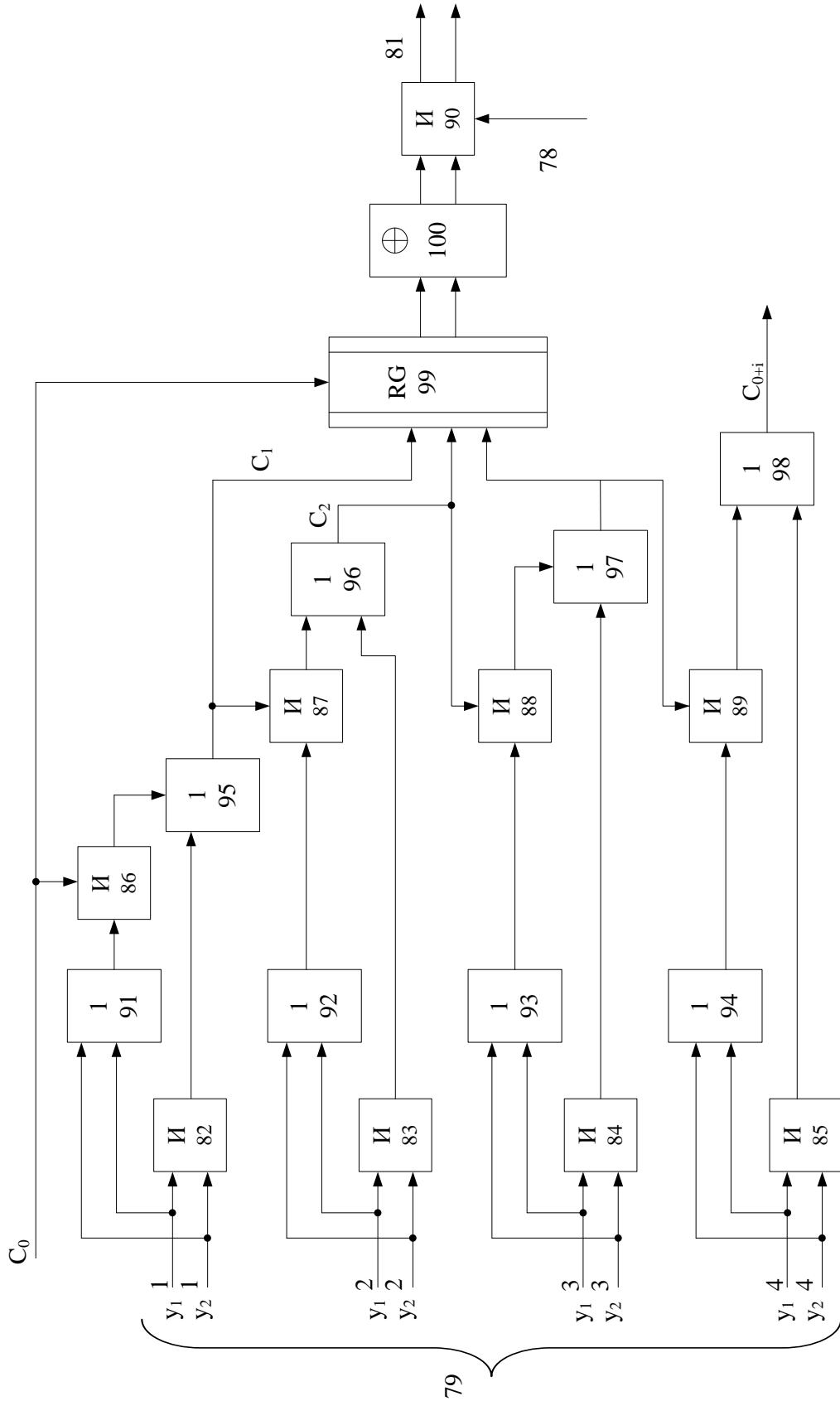


Рис. 3.5.4 - Функциональная схема формирования поправки при выполнении арифметических операций

Функциональная схема формирования поправки при выполнении арифметических операций (Рис.3.5.4) содержит первый элемент 82 И, второй элемент 83 И, третий элемент 84 И, четвертый элемент 85 И, пятый элемент 86 И, шестой элемент 87 И, седьмой элемент 88 И, восьмой элемент 89 И, блок элементов 90 И, первый элемент 91 ИЛИ, второй элемент 92 ИЛИ, третий элемент 93 ИЛИ, четвертый элемент 94 ИЛИ, пятый элемент 95 ИЛИ, шестой элемент 96 ИЛИ, седьмой элемент 97 ИЛИ, восьмой элемент 98 ИЛИ, блок регистров 99, группу элементов 100 неравнозначности, информационные входы 79, вход 78 разрешающий считывание выходной информации, информационные выходы 81.

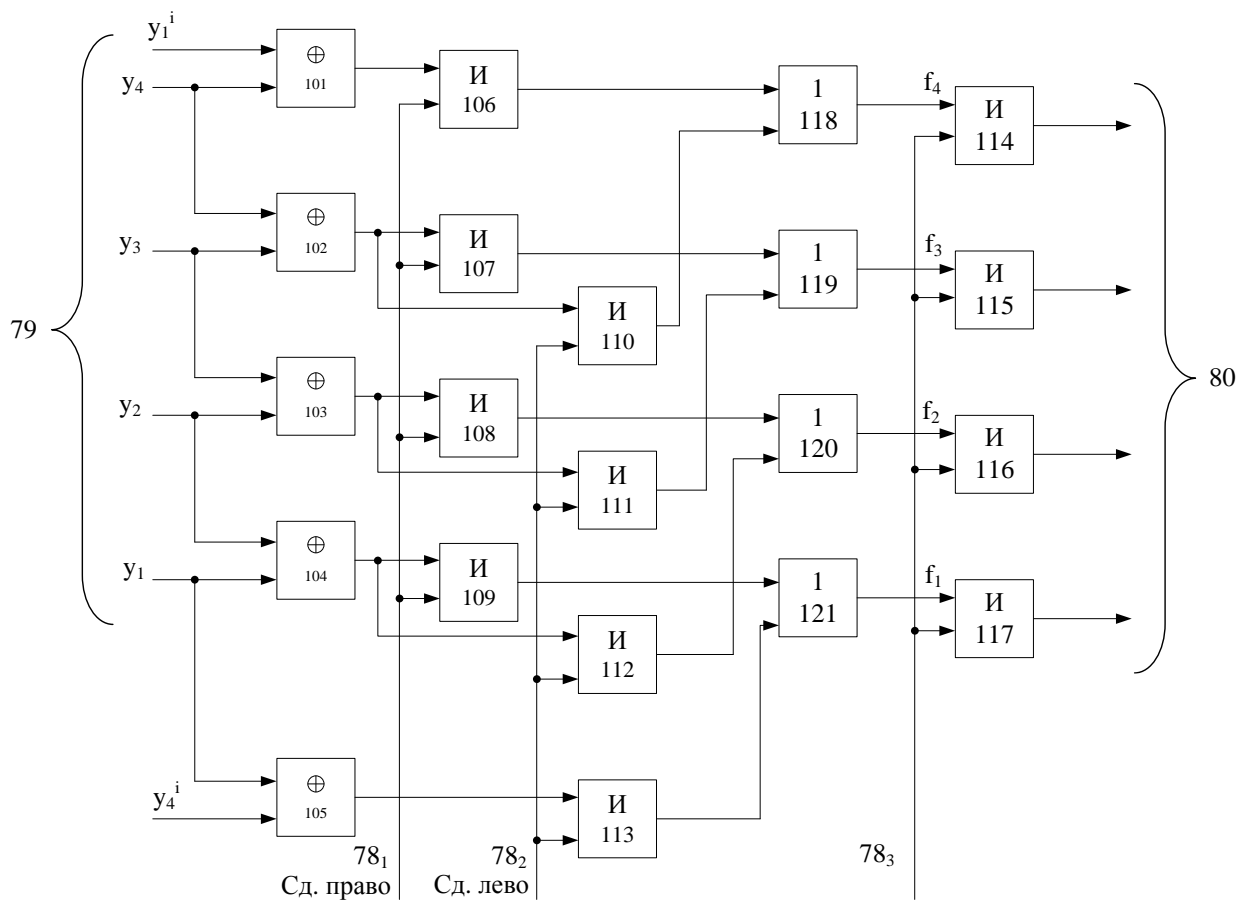


Рис. 3.5.5 - Функциональная схема формирования поправки при выполнении операции сдвига

Функциональная схема формирования поправки при выполнении операции сдвига (Рис.3.5.5) содержит первый элемент 101 неравнозначности, второй элемент 102 неравнозначности, третий элемент 103 неравнозначности, четвертый элемент 104 неравнозначности, пятый элемент 105 неравнозначности, первый элемент 106 И, второй элемент 107 И, третий элемент 108 И, четвертый элемент 109 И, пятый элемент 110 И, шестой элемент 111 И, седьмой элемент 112 И, восьмой элемент 113 И, девятый элемент 114 И, десятый элемент 115 И, одиннадцатый элемент 116 И, двенадцатый элемент 117 И, первый элемент 118 ИЛИ, второй элемент 119 ИЛИ, третий элемент 120 ИЛИ, четвертый элемент 121 ИЛИ, информационные входы 79, вход 78_1 управляющего сигнала сдвиг вправо, вход 78_2 управляющего сигнала сдвиг влево, вход 78_3 разрешающий считывание выходной информации, выходы 80 схемы формирования поправки при выполнении операции сдвига.

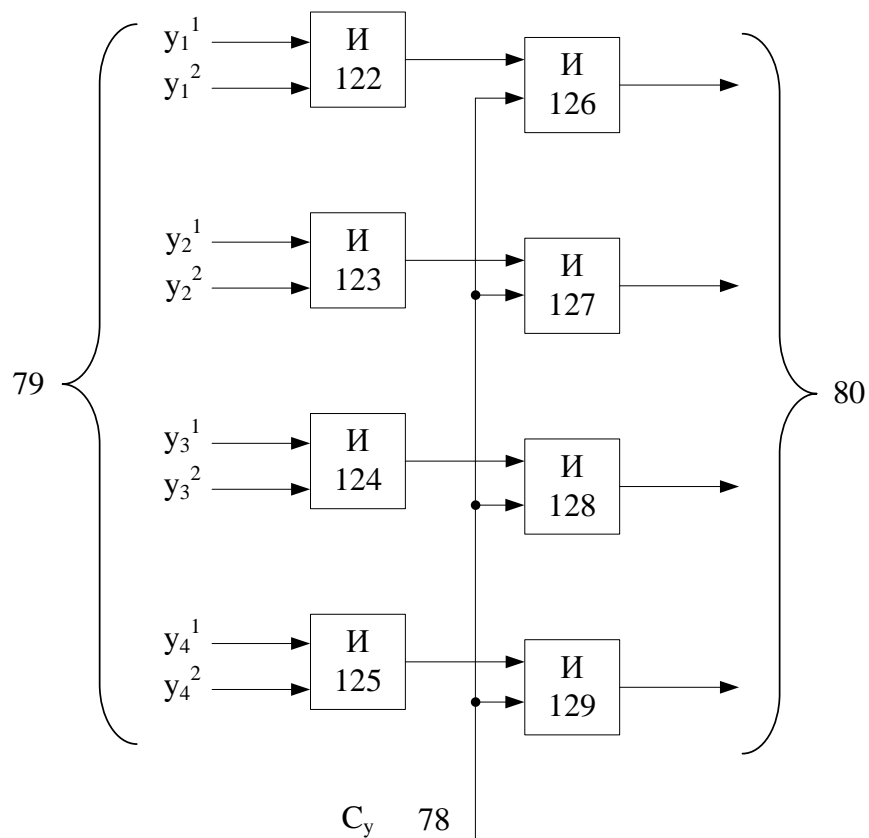


Рис. 3.5.6 - Функциональная схема формирования поправки при выполнении операции ИЛИ

Функциональная схема формирования поправки при выполнении логической операции ИЛИ (Рис.3.5.6) содержит первый элемент 122 И, второй элемент 123 И, третий элемент 124 И, четвертый элемент 125 И, пятый элемент 126 И, шестой элемент 127 И, седьмой элемент 128 И, восьмой элемент 129 И, информационные входы 79, вход 78 разрешающий считывание выходной информации, информационные выходы 80 функциональной схемы формирования поправки при выполнении логической операции ИЛИ.

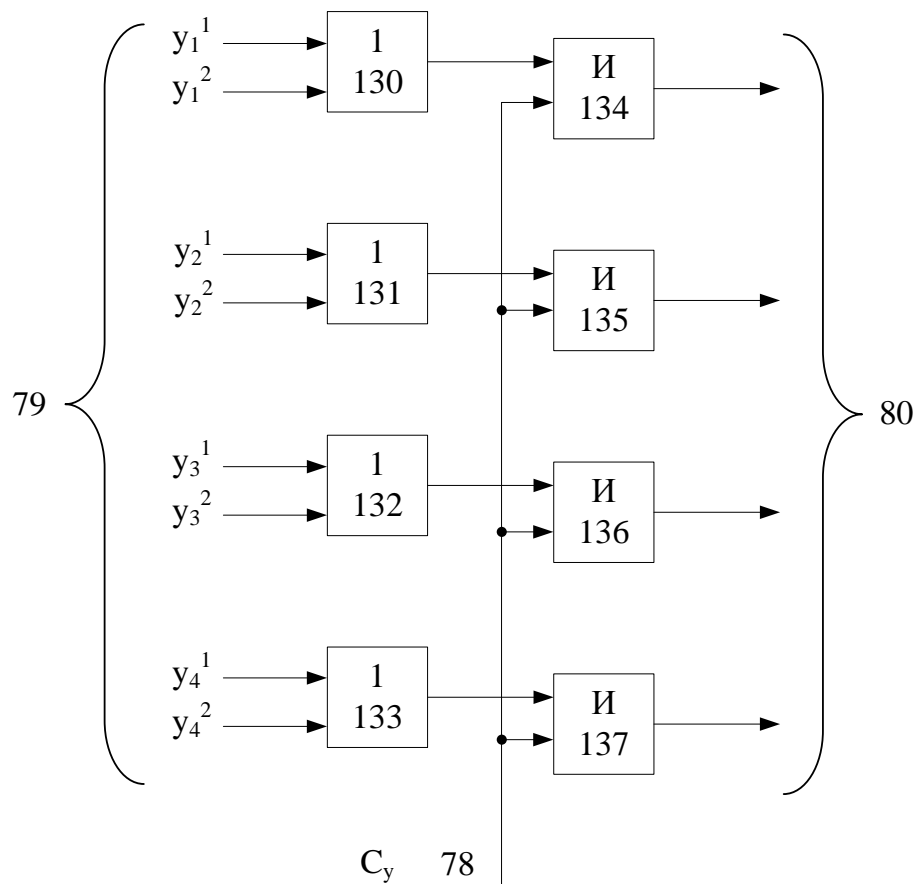


Рис. 3.5.7 - Функциональная схема формирования поправки при выполнении операции И

Функциональная схема формирования поправки при выполнении логической операции И (Рис.3.5.7) содержит первый элемент 130 ИЛИ, второй элемент 131 ИЛИ, третий элемент 132 ИЛИ, четвертый элемент 133 ИЛИ, первый элемент 134 И, второй элемент 135 И, третий элемент 136 И, четвертый элемент 137 И, информационные входы 79, вход 78 разрешающий считывание выходной информации, информационные выходы 80 функциональной схемы формирования поправки при выполнении логической операции И.

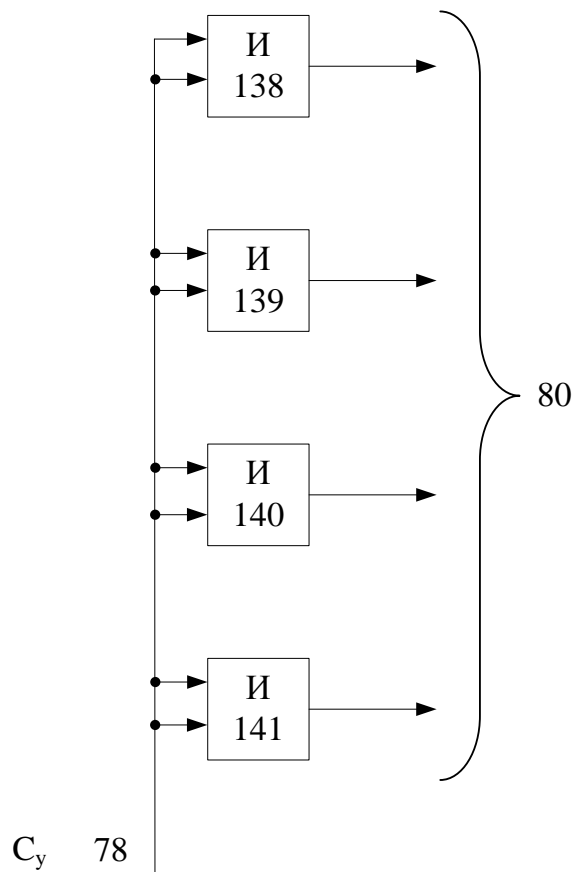


Рис. 3.5.8 - Функциональная схема формирования поправки при выполнении операции НЕ

Функциональная схема формирования поправки при выполнении логической операции НЕ (Рис.3.5.8) содержит первый элемент 138 И, второй элемент 139 И, третий элемент 140 И, четвертый элемент 141 И, вход 78

разрешающий считывание выходной информации, информационные выходы 80 функциональной схемы формирования поправки при выполнении логической операции НЕ.

Процессор включает в себя два основных устройства: управляющий узел 1 и операционный узел 2.

Управляющий узел 1 координирует действия узлов операционного узла 2 между собой и с другими устройствами ЭВМ, а также выполняет набор операций, включающих команды обращения к памяти. Он вырабатывает в определенной временной последовательности управляющие сигналы, под действием которых в узлах операционного узла 2 выполняются требуемые действия.

Каждое такое элементарное действие, выполняемое в операционном узле 2 в течение одного тактового периода, называется микрооперацией.

В определенные тактовые периоды одновременно могут выполняться несколько микроопераций. Такая совокупность одновременно выполняемых микроопераций называется микрокомандой, а весь набор микрокоманд, предназначенных для решения определенной задачи называется микропрограммой.

Общий временной интервал, в течение которого происходит выборка, хранение и преобразование одной команды в набор управляющих сигналов, называется циклом работы управляющего узла 1.

Таким образом, управляющий узел 1 осуществляет преобразование команды в соответствующий набор управляющих сигналов и обеспечивает:

чтение команды, находящейся в очередной ячейке памяти;

расшифровку кода операции (команды);

отыскание операндов (чисел) по указанному адресу, содержащемуся в команде;

обеспечить выдачу управляющих сигналов в операционный узел для выполнения над ними действий, указанных в коде операции команды.

В данном случае используется микропрограммный управляющий узел 1, в котором микрокоманды хранятся в управляющей памяти 19.

В этом случае, слова, отображающие команды хранятся в памяти в последовательно пронумерованных ячейках, что позволяет формировать адрес очередной команды, добавлением единицы к адресу предыдущей команды, при этом слово состоит из нескольких частей: например, кода операции, указывающего вид операции и адресов чисел, над которыми должна быть произведена соответствующая операция.

Дешифратор 3 кода операций по выбранной из оперативной памяти команде определяет номер требуемой микропрограммы в управляющей памяти 19.

Генератор 4 тактовых импульсов предназначен для формирования тактовых и синхронизирующих импульсов.

Счетчик 9 команд предназначен для формирования адреса ячейки памяти очередной команды, путем естественной выборки, т.е. прибавлением к его содержимому единицы.

Регистр 11 адреса предназначен для формирования адреса ячейки памяти при командах условного или безусловного перехода с возвратом.

Блок 5 управления предназначен для определения адреса очередной микрокоманды в управляющей памяти 20, формирования адреса очередной команды (управления работой первого коммутатора б), координации работы (выдачи синхроимпульсов) устройств процессора .

Управляющая память 19 представляет собой постоянное запоминающее устройство и предназначена для выдачи (в зависимости от кода операции) выдачи управляющих сигналов (сигналов управления, сигналов считывания, сигналов записи, сигналов установки в нулевое состояние) на функциональные узлы процессора. При этом слово памяти

содержит информационные разряды (для управляющих сигналов) и контрольные разряды, сформированные на основе предлагаемого метода кодирования.

В этом случае, для формирования значений контрольных разрядов управляющей памяти 19, информационные разряды представляются в виде двухстрочной информационной матрицы:.....

$$\begin{array}{ccccccc} y_1 & y_2 & \dots & y_{k/2} & r_{c1} & & \\ y_{(k/2)+1} & y_{(k/2)+2} & \dots & y_k & r_{c2} & & \end{array},$$

при этом:

1) для каждой строки информационной матрицы организуется проверка на четность;

2) проводятся правые и левые диагональные проверки. Число диагональных проверок (число контрольных разрядов диагональных проверок) определяется по формуле (разряды на четность не передаются):

$$R_{\text{д}} = k + 4$$

Блок 17 коррекции (Рис.3.5.2) предназначен для обнаружения и исправления ошибок, возникающих при считывании информации с управляющей памяти 19. В этом случае, при считывании микрокоманды, кодирующей схемой 40 проводится формирование (аналогичным образом) вектора контрольных разрядов R^{II} принятого кодового набора.

Таким образом, в период считывания информации, на входах схемы 41 сравнения имеем соответственно векторы контрольных разрядов

$$\begin{array}{l} R = r_1 r_2 \dots r_{k+4}, \\ R^{\text{II}} = r_1^{\text{II}} r^{\text{II}} \dots r_{k+4}^{\text{II}}. \end{array}$$

Схема 41 сравнения представляют собой схему поразрядного сравнения и предназначена для формирования значений синдрома ошибки на основе передаваемой и полученной информации.

Результат сложения по mod 2 значений сигналов переданных и сформированных контрольных разрядов даст синдром ошибки.

В зависимости от значения синдрома ошибки (значения сигнала на выходе дешифратор 48) формирователь вектора ошибки 46 формирует вектор ошибки: $E = e_1e_2e_3\dots\dots\dots e_{k+4}$, имеющий единичные значения сигналов в разрядах кодового набора, которые имеют ошибку.

Если кодовый набор не содержит ошибки, то синдром равен нулю.

Если кодовый набор содержит некорректируемую ошибку, то на выходе элемента 42 ИЛИ появится единичное значение сигнала, а на выходе элемента 43 ИЛИ будет присутствовать нулевое значение. В этом случае на выходе 22 появится сигнал «отказ процессора».

Если кодовый набор содержит корректируемую ошибку, то на выходе элементов 42 и 43 ИЛИ одновременно появятся единичные значения сигналов. В этом случае сигнал на выходе 22 имеет нулевое значение, формирователь 46 вектора ошибки формирует вектор ошибки для исправления ошибки информационных разрядов кодового набора.

Корректор 47 включает k -элементов неравнозначности и предназначен для исправления ошибок, возникающих в информационных разрядах управляющей памяти 19. При исправлении ошибок реализуется функция по mod2 относительно информационных разрядов управляющей памяти 19.

Операционный узел 2 предназначен для выполнения арифметических и логических операций и включает в свой состав (Рис.3.5.1) счетчик 10 сдвигов, регистр 12 числа, регистр 13 сумматора, регистр 14 дополнительный, регистр 15 дополнительного кода, сумматор 16, блок 18 контроля.

Счетчик 10 сдвигов предназначен для подсчета количества сдвигов при выполнении операций умножения и деления, количества сдвигов промежуточных результатов и нормализации.

Регистр 11 адреса представляет собой регистр памяти и предназначен для хранения адресе очередной команды.

Регистр 12 числа представляет собой регистр памяти и предназначен для хранения операндов при выполнении арифметических и логических операций (хранения множимого, при выполнении операции умножения и делителя при выполнении операции деления).

Регистр 13 сумматора (аккумулятора) представляет собой регистр сдвига (вправо - при выполнении операции умножения и влево - при выполнении операции деления), и предназначен для хранения делимого старших разрядов результата умножения.

Заметим, что при выполнении операции деления, блоком 5 управления анализируется значение знакового разряда регистра 12 числа и регистра 13 сумматора дополнительного.

Регистр 14 дополнительный представляет собой регистр сдвига (вправо - при выполнении операции умножения и влево - при выполнении операции деления), и предназначен для хранения множителя и младших разрядов результата умножения при выполнении операции умножения и результата деления при выполнении операции деления).

Заметим, что при выполнении операции умножения, блоком 5 управления анализируется значение младшего разряда регистра 14 дополнительного.

Регистр 15 дополнительного кода представляет собой регистр памяти и предназначен для хранения отрицательного числа в дополнительном коде (при выполнении операции вычитания и операции деления).

Сумматор 16 представляет собой параллельный n - разрядный сумматор и предназначен для выполнения операции сложения чисел.

Блок 18 контроля (Рис.3.5.2) предназначен для обнаружения и коррекции ошибок, возникающих при выполнении арифметических и логических операций.

Так, при выполнении операции арифметических операций, результат суммы с выхода 24^1 сумматора 16, через открытую первую группу 61

элементов И, первую группу 53 элементов ИЛИ поступает на вход второй кодирующей схемы 76 в котором формируются значения контрольных разрядов относительно принятой информации.

Одновременно значения контрольных разрядов слагаемых с выходов 25 третьего коммутатора 8 через открытую пятую группу 65 элементов И поступают на входы первой группы 50 элементов неравнозначности, где производится суммирование одноименных контрольных разрядов по mod2. В то же время значения информационных разрядов слагаемых поступают на вход 79 формирователя 73 поправки.

В данном случае значения информационных разрядов поступают на вход схемы формирования поправки при выполнении арифметических операций (Рис. 3.5.4). При помощи элементов 82-89 И, элементов 91-98 ИЛИ формируются значения сигналов переноса C_i в соответствии с положениями рассмотренными в приложении.

Так например, при сложении числа $A=0010$ **01011010** и числа $B=0110$ **00111001** (значения контрольных разрядов выделены жирным шрифтом) на выходе первого элемента 50 неравнозначности получим значение **01100011**. Открывается элемент 83 И схемы формирования поправки при выполнении арифметических операций, что приводит к формированию сигнала переноса C_2 , а наличие единичного сигнала в третьем разряде одного из слагаемых и сигнала C_2 приводит к открытию элемента 88 И, что в свою очередь приводит к появлению сигнала C_3 . В блоке регистров 99 памяти записаны значения поправок.

Сигналу переноса C_2 соответствует значение поправки 01100011, а сигналу переноса C_3 значение поправки равно 10100101.

В этом случае сложение по mod2 поправок, считываемых с блока регистров 99 на блоке элементов неравнозначности 100 даст значение поправки: 11000110. В результате сложения на второй группе 51 элементов неравнозначности значений контрольных разрядов **01100011**, поступающих с

выхода первой группы 50 элементов неравнозначности с значением поправки 11000110, поступающей с выхода 81 формирователя 73 поправки через восьмую группу элементов ИЛИ, получим окончательное значение контрольных разрядов для рассматриваемой суммы информационных разрядов, т.е. при сложении чисел $A=0010$ **01011010** и $B=0110$ **00111001** получим: 1000 **10100101** (жирным шрифтом представлены сформированные значения контрольных разрядов для полученного результата суммы информационных разрядов).

Если значения контрольных разрядов, поступающие с выходов второй кодирующей схемы 76 на вход третьей группы 52 элементов неравнозначности, совпадают с значениями контрольных разрядов, поступающих с выходов второй группы 51 элементов неравнозначности, то на выходе третьей группы 52 элементов неравнозначности имеем нулевые значения, т.е. операция суммирования проведена правильно, если не совпадают - произошла ошибка. Таким образом на выходе третьей группы 52 элементов неравнозначности имеем значение синдрома ошибки.

Значения синдрома ошибки поступает на входы дешифратора 72.

В зависимости от значения синдрома ошибки (значения сигнала на выходе дешифратора 72) формирователь вектора ошибки 71 формирует вектор ошибки: $E = e_1 e_2 e_3 \dots e_{k+4}$, имеющий единичные значения сигналов в разрядах кодового набора, которые имеют ошибку.

Если кодовый набор не содержит ошибки, то синдром равен нулю.

Если кодовый набор содержит некорректируемую ошибку, то на выходе шестого элемента 58 ИЛИ появится единичное значение сигнала, а на выходе седьмого элемента 59 ИЛИ будет присутствовать нулевое значение, В этом случае на выходе 28 элемента 70 И появится сигнал «отказ процессора».

Если кодовый набор содержит корректируемую ошибку, то на выходе групп элементов 58 и 59 ИЛИ одновременно появятся единичные значения сигналов. В этом случае сигнал на выходе 28 имеет нулевое значение,

формирователь 71 вектора ошибки формирует вектор ошибки для исправления ошибки разрядов кодового набора. На выходе 29 седьмого элемента 59 появится сигнал «корректируемая ошибка».

Если информация содержит большее количество информационных разрядов, то учитывается перенос из младшего полубайта информации $-C_0$, и C_{0+i} – перенос в старший полубайт информации.

Корректор 74 включает элементы неравнозначности и предназначен для исправления ошибок, возникающих в разрядах кодового набора.

При исправлении ошибок реализуется функция по mod2 относительно разрядов кодового набора и значений сигналов вектора ошибки.

При контроле операций сдвига, формирователь 73 поправки реализует логику работы функциональной схемы формирования поправки при выполнении операции сдвига.

Допустим необходимо выполнить операцию сдвига вправо содержимое 1101 (результаты частных произведений) регистра 13 сумматора, для которого имеем следующие значения контрольных разрядов **11110000**.

После выполнения операции сдвига вправо получим кодовый набор, содержащий значения информационных разрядов 0110, которые с выходов 24 третьего коммутатора 8, через вторую группу элементов 62 И, третью группу 55 элементов ИЛИ поступают на вход корректора 74, а через первую группу 53 элементов ИЛИ на вход второй кодирующей схемы 76 на выходе которой получим значение контрольных разрядов: 00111001.

Одновременно схемой формирования поправки при выполнении операции сдвига (Рис.3.5.5) по исходным значениям информационных разрядов формируются разряды матрицы поправок в соответствии с выражениями представленными в приложении:

$r_4 = y_1^i \oplus y_4$; (где y_1^i – значение младшего разряда старшего полубайта информации. В данном случае рассмотрим вариант без переноса, т.е. берется

0, так как в старший разряд регистра сумматора нет переноса) $r_3=y_4\oplus y_3$;
 $r_2=y_3\oplus y_2$; $r_1=y_2\oplus y_1$.

При этом на выходах элементов 101-104 получим соответственно значения сигналов: 1011, которые по сигналу на входе 78₁ «сдвиг вправо», через открытые элементы 106-109 И, элементы 118-121 ИЛИ и поступлении разрешающего сигнала «считывание» поступают на выходы формирователя 73 поправки и далее на входы первой кодирующей схемы 75.

По значению разрядов 1011 матрицы поправки первой кодирующей схемой 75 определяются разряды четности и организуются диагональные проверки. В результате получим значение поправки 11001001, которое поступает на первые входы второй группы 51 элементов неравнозначности.

В это же время исходные значения контрольных разрядов 11110000, через четвертый элемент 64 И, четвертый элемент 56 ИЛИ поступают на вторые входы второй группы 51 элементов неравнозначности. Сложение по mod2 поступающей на входы второй группы 51 элементов неравнозначности даст результат: 00111001, который при отсутствии ошибки равен значению информации, поступающей с выходов второй кодирующей схемы 76.

Таким образом, если значения контрольных разрядов, сформированные второй кодирующей схемой 76 и полученных на выходе второй группы 51 элементов неравнозначности совпадают, то операция сдвига выполнена правильно, если нет, то произошла ошибка. В этом случае обнаружение и коррекция возникающих ошибок осуществляется точно также как и при выполнении арифметических операций.

Если операция сдвига вправо проводится для регистра 14 дополнительного, то учитывается перенос значения младшего разряда регистра 13 сумматора в старший разряд регистра 14 дополнительного, т.е. функциональная схема формирования поправки при выполнении операции сдвига реализует функцию:

$$r_4=y_1^i\oplus y_4; \quad r_3=y_4\oplus y_3; \quad r_2=y_3\oplus y_2; \quad r_1=y_2\oplus y_1.$$

При выполнении операции сдвига влево (при выполнении операции деления) функциональная схема формирования поправки при выполнении операции сдвига реализует функцию: $r_4=y_4\oplus y_3$; $r_3=y_3\oplus y_2$; $r_2=y_2\oplus y_1$; $r_1=y_4^i\oplus y_1$.

В этом случае подаются управляющие сигналы на входы 78₂, 78₃, далее схема работает аналогичным образом.

При выполнении логической операции ИЛИ формирователь поправки работает следующим образом. Значения контрольных разрядов операндов суммируются по mod2 в первой группе 50 элементов неравнозначности, после чего результат поступает на первые входы второй группы 51 элементов неравнозначности, на вторые входы которого поступает значение поправки. При этом функциональная схема формирования поправки при выполнении логической операции ИЛИ (Рис. 3.5.6), для формирования информационной матрицы поправки, реализует логическую функцию И относительно одноименных разрядов операндов.

Так, например для чисел $A=0101\ 01010101$ и $B=0011\ 01101100$ после выполнения операции ИЛИ для информационных разрядов и сложения по mod2 контрольных разрядов рассматриваемых слагаемых, получим кодовый набор 0111 00111001, у которого значения контрольных разрядов отличаются от правильного набора контрольных разрядов 00001111 для полученного значения информационных разрядов 0111.

В этом случае, логические операции И, выполненные относительно информационных разрядов рассматриваемых операндов на элементах 122,123,124,125 И дадут результат: 0001(операция И выполняется для младших разрядов операндов).

Тогда матрица поправок операции ИЛИ, формируемая первой кодирующей схемой 75 имеет вид:

00 0

01 1

правые и левые диагональные проверки данной матрицы дадут результат поправки: 00110110.

Сложение результата 00111001, полученного на выходе первой группы 50 элементов неравнозначности с значением поправки 00110110 дадут правильное значение контрольных разрядов для полученного значения информационных разрядов 0111:

$$\begin{array}{r} 00111001 \\ \oplus 00110110 \\ \hline 00001111. \end{array}$$

Если произошла ошибка, то ее обнаружение и коррекция осуществляется как и для операций описанных выше.

При выполнении логической операции И формирователь поправки работает следующим образом. Значения контрольных разрядов операндов суммируются по модулю в первой группе 50 элементов неравнозначности, после чего результат поступает на первые входы второй группы 51 элементов неравнозначности, на вторые входы которого поступает значение поправки. При этом функциональная схема формирования поправки при выполнении логической операции И (Рис.3.5.7), для формирования информационной матрицы поправки, реализует логическую функцию ИЛИ относительно одноименных разрядов операндов на группе логических элементов 130, 131, 132, 133 ИЛИ. Далее формирование поправки, обнаружение и коррекция возникающих ошибок проводится аналогичным способом.

При контроле логической операции НЕ формирование поправки осуществляется функциональной схемой формирования поправки при выполнении логической операции НЕ (Рис.3.5.8), которая формирует единичные значения информационных разрядов матрицы поправок на основе использования элементов 138, 139, 140, 141 И.

Далее формирование контрольных разрядов проводится аналогично формированию контрольных разрядов при выполнении операций И и ИЛИ.

При контроле выполнения операции сложения по mod2 не требуется формирования поправки к результату суммирования по mod2 значений контрольных разрядов операндов. В этом случае результат суммирования с выходов первой группы элементов 50 неравнозначности через шестую группу 66 элементов И, пятую группу 57 элементов ИЛИ поступает на ходы третьей группы 52 элементов неравнозначности, где сравнивается с значениями контрольных разрядов сформированных на выходах второй кодирующей схемы 76. Далее устройство работает аналогичным образом.

Работа процессора начинается с приходом сигнала “Пуск” по входной группе 30 входов устройства обмена процессора с периферийными блоками. По этой команде блок 5 выдает команду на считывание с управляющей памяти 19 содержимого первой ячейки памяти.

В первой ячейке памяти расположена команда “Сброс системы”, которая устанавливает в исходное состояние регистры и блоки процессора,

В счетчик 9 команд записывается “1”, устройство управления 1 выдает микрокоманды в следующей последовательности:

1) На первом такте сигналы микрокоманды и значения контрольных разрядов поступают на выход блока 17 коррекции, где проводится обнаружение и коррекция возникающих ошибок в соответствии с функциональной схемой, представленной на Рис.3.5.7.

В этом случае, при считывании микрокоманды, кодирующей схемой 40 проводится формирование вектора контрольных разрядов R^{Π} принятого кодового набора.

Таким образом, в период считывания информации, на входах схемы 41 сравнения имеем соответственно векторы контрольных разрядов

$$R = r_1 r_2 \dots r_{k+4},$$

$$R^{\Pi} = r_1^{\Pi} r^{\Pi} \dots r_{k+4}^{\Pi}.$$

Схема 41 сравнения формирует значения синдрома ошибки на основе передаваемой и полученной информации.

Результат сложения по mod 2 значений сигналов переданных и сформированных контрольных разрядов даст синдром ошибки.

В случае возникновения ошибок, на одном из выходов дешифратора 48 формируется единичный сигнал по которому формирователь 46 вектора ошибки формирует вектор ошибки $E = e_1 e_2 e_3 \dots e_{k+4}$.

В этом случае корректор 47 осуществляет исправление ошибок, возникающих в информационных разрядах управляющей памяти 19. При исправлении ошибок реализуется функция по mod2 относительно сигналов вектора ошибки и информационных разрядов управляющей памяти 19.

Если ошибок нет, или возникла корректируемая ошибка, набор микрокоманд поступает на вход считывания счетчика 9 команд и на вход записи регистра 11 адреса, при этом содержимое счетчика 9 команд пересылается в регистр 11 адреса (или через первый коммутатор 6 на адресные входы запоминающего устройства при естественной выборке команд непосредственно со счетчика 9 команд);

2) На втором такте к содержимому счетчика 9 команд прибавляется единица - подготавливается адрес следующей команды;

3) На третьем такте сигналы микрокоманды поступают на вход считывания регистра 11 адреса и на вход считывания содержимого ячейки памяти запоминающего устройства по указанному адресу. При этом команда, хранящаяся в первой ячейке памяти, записывается в регистр 12 числа;

4) На четвертом такте сигналы микрокоманды подаются на вход считывания регистра 12 числа, вход второго коммутатора 7 и на вход дешифратора 3 кода операции, где раскодируются, после чего управляющий узел 1 переходит ко второму этапу работы.

Для примера рассмотрим порядок исполнения одной из команд, записанной в регистре 12 числа после выполнения первых четырех тактов.

Пусть в поле кода операции команды содержимого регистра 12 числа записана команда сложения содержимого регистра 13 сумматора с числом

расположенным запоминающем устройстве по адресу, указанному в поле адреса регистра 12 числа (при использовании одноадресной команды).

Управляющий узел 1 при этом выдает следующие микрокоманды:

5) на пятом такте сигналы микрокоманды подаются на вход считывания регистра 12 числа, на вход второго коммутатора 7, первого коммутатора 6 и на вход записи регистра 11 адреса (адрес, хранящийся в регистре 12 числа записывается в регистр 11 адреса, содержимое регистра 12 числа обнуляется);

6) На шестом такте сигналы микрокоманды подаются на вход считывания регистра 11 адреса, на вход первого коммутатора 6, на вход считывания запоминающего устройства и на вход записи регистра 12 числа (из запоминающего устройства в регистр 12 числа записывается второе слагаемое (считаем, что первое слагаемое уже находится в регистре 13 сумматора);

7) На седьмом такте сигналы микрокоманды подаются на вход считывания регистра 12 числа и регистра 13 сумматора, при этом арифметико-логическое устройство осуществляет операцию сложения и запись результата сложения в регистр 13 сумматора следующим образом.

Результат суммы с выхода сумматора 16, поступает на блок 18 контроля, в котором через открытую первую группу 61 элементов И, первую группу 53 элементов ИЛИ поступает на вход второй кодирующей схемы 76 в котором формируются значения контрольных разрядов относительно принятой информации.

Одновременно значения контрольных разрядов слагаемых с выходов 25 третьего коммутатора 8 поступают на входы открытой пятой группы 65 элементов И и, с ее выходов на входы первой группы 50 элементов неравнозначности, где производится суммирование одноименных контрольных разрядов по mod2. В то же время значения информационных разрядов слагаемых через открытую третью группу 63 элементов И поступают на вход 79 формирователя 73 поправки.

В данном случае значения информационных разрядов поступают на вход схемы формирования поправки при выполнении арифметических операций. При помощи групп 82-89 элементов И, групп 91-98 элементов ИЛИ, блока 99 регистров, элементов 100 неравнозначности формируются значения сигналов переноса C_1 в соответствии с положениями рассмотренными в приложении. Так например, при сложении числа $A=0010$ и числа $B=0110$ открывается элемент 83 И что приводит к формированию сигнала переноса C_2 , а наличие единичного сигнала в третьем разряде одного из слагаемых и сигнала C_2 приводит к открытию элемента 84 И, что в свою очередь приводит к появлению сигнала C_3 . В блоке 99 регистров памяти записаны значения разрядов информационных матриц поправок. Сложение по mod2 одноименных разрядов поправок на элементах 100 неравнозначности даст требуемую поправку.

При поступлении разрешающего сигнала на вход 78, открывается элемент 90 И и полученная информация поступает на выходы 81 формирователя 73 поправок.

В результате на выходе 81 получим значение поправки: 1100 0110, которая поступает на первые входы второй группы 51 элементов неравнозначности.

Для рассматриваемого примера суммирование по mod2 значений сигналов контрольных разрядов первой группой 50 элементов неравнозначности даст результат:

$$\begin{array}{r} 01011010 \\ \oplus 00111001 \\ \hline 01100011, \end{array}$$

который поступает на вторые входы второй группы 51 элементов неравнозначности. В результате на выходе второй группы 51 элементов неравнозначности получим скорректированное значение контрольных разрядов для полученного результата суммы:

140

01100011

\oplus 1100 0110

10100101,

которое при правильном суммировании чисел А и В:

0010

+0110

1000,

соответствует правильному значению контрольных разрядов, поступающему с выходов второй кодирующей схемы 76 (10100101) на первые входы третьей группы 52 элементов неравнозначности, на вторые входы которой, чрез пятый элемент 57 ИЛИ поступают скорректированные значения контрольных разрядов 10100101.

Если значения контрольных разрядов, совпадают то на выходе третьей группы 52 элементов неравнозначности имеем нулевые значения, т.е. операция суммирования проведена правильно, если не совпадают - произошла ошибка.

Значения сигналов с выходов третьей группы 52 элементов неравнозначности поступают на входы дешифратора 72.

Если произошла некорректируемая ошибка, то на выходах шестого элемента 58 ИЛИ появится единичное значение сигнала, а на выходе седьмого элемента 59 ИЛИ будет присутствовать нулевое значение сигнала. В этом случае на выходе элемента 70 И (выход 28) появится сигнал «отказ процессора».

Если произошла корректируемая ошибка то единичное значение сигнала появится на выходе седьмого элемента 59 ИЛИ (выход 29 «корректируемая ошибка») и на входе формирователя 71 вектора ошибки, который формирует соответствующее значение вектора ошибки.

Корректор 74 включает k -элементов неравнозначности и предназначен для исправления ошибок, возникающих в информационных разрядах.

При исправлении ошибок реализуется функция по $\text{mod}2$ относительно разрядов кодового набора и разрядов вектора ошибки.

Аналогичным образом процессор функционирует при выполнении логических операций. В этом случае при выполнении логических операций «сдвиг» и «НЕ» исходные значения контрольных разрядов с входа 25 через открытую четвертую группу 64 элементов И, четвертую группу 56 элементов ИЛИ поступают на первые входы второй группы 51 элементов неравнозначности. Значение поправки контрольных разрядов с выходов первой кодирующей схемы 75 через открытую седьмую группу 67 элементов И, восьмую группу 60 элементов ИЛИ поступает на вторые входы второй группы 51 элементов неравнозначности, где формируется значение контрольных разрядов для данной операции.

При выполнении логических операций «И», «ИЛИ» и « $\text{mod}2$ » значения контрольных разрядов со входа 25, через открытую пятую группу 65 элементов И поступают на входы первой группы 50 элементов неравнозначности. В этом случае, при выполнении логической операции « $\text{mod}2$ » результат суммы через открытую шестую группу 65 элементов И, пятую группу 57 элементов ИЛИ поступает на первые входы третьей группы 52 элементов неравнозначности, т.е. сформированное значение контрольных разрядов поступает без поправки.

При выполнении логических операций «И» и «ИЛИ» результат суммы с выходов первой группы 50 элементов неравнозначности через открытую девятую группу 69 элементов И, четвертую группу 56 элементов ИЛИ поступает на первые входы второй группы 51 элементов неравнозначности, на вторые входы которой поступает значение поправки. Далее полученное значение контрольных разрядов сравнивается с значением контрольных разрядов сформированных второй кодирующей схемой 75 на третьей группе 52 элементов неравнозначности.

8) На восьмом такте выдается микрокоманда “Конец операций” осуществляется переход на следующую операцию, блок 5 управления приводится в исходное состояние и выдает разрешение на начало выполнения следующей команды, адрес которой указан в счетчике 9 команд.

На основе разработанных функциональной схем могут быть построены макетные образцы отказоустойчивых ЭВМ. Они реализуются на программируемой логической интегральной схеме (ПЛИС) в виде 16-разрядной отказоустойчивой ЭВМ повышенной достоверности функционирования.

Макетный образец должен представлять собой программно-аппаратное изделие, состоящее из ПЛИС и персонального компьютера с установленным специальным программным обеспечением, позволяющим эмулировать и визуализировать работу отказоустойчивой ЭВМ повышенной достоверности функционирования.

В нем должны быть реализованы алгоритмы моделирования ошибок работы:

- искусственная генерация ошибок хранения данных;
- генерация ошибок при передаче данных (записи/чтении) в устройство памяти;
- возможны другие типы ошибок (по согласованию).

Макетный образец отказоустойчивой ЭВМ повышенной достоверности функционирования, реализующий метод функционально кодовой защиты обеспечивать:

- коррекцию трехкратных, при условии обнаружения некорректируемых ошибок;
- в зависимости от правила проведения дополнительных проверок, корректировать от 50% до 98% обнаруживаемых ошибок;
- исправлять ошибки различной конфигурации при условии обнаружения некорректируемых ошибок;
- иметь минимальные временные затраты на декодирование;

- исключать влияние неисправного резервного оборудования на работу устройств ЭВМ при наличии ошибок в контрольных разрядах и отсутствии ошибок в информационных;

- сигнализировать о неисправности при возникновении некорректируемой ошибки.

Выводы к главе 3

1. Выбран комплекс защиты информации для корпоративных информационно-телекоммуникационных сетей, с учетом исследованных и предложенных методик кодирования.
2. Предложена методика обеспечения отказоустойчивости сумматора на основе корректирующих линейных кодов.
3. Разработаны подходы к обнаружению и коррекции ошибок арифметических операций функционального ядра КСОН.
4. Разработаны методики и структуры функционально-кодовой защиты процессора при выполнении логических операций.

5. Разработаны функциональные схемы отказоустойчивого процессора повышенной достоверности функционирования с использованием ПЛИС, которые были внедрены и показали хорошие результаты (см. приложения).

ЗАКЛЮЧЕНИЕ

В настоящее время для обеспечения отказоустойчивости функциональных узлов ЭВМ используются корректирующие линейные коды, исправляющие одиночную ошибку, реализация которых требует минимальных аппаратурных затрат на кодирование и декодирование информации, составляющих 30-40 % относительно резервируемого устройства.

Однако в настоящее время неизвестны эффективные методы построения линейных кодов исправляющих больше двух кратной ошибки.

Для контроля большинства логических операций невозможно сформировать контрольные разряды, которые оказались бы совместимыми с данными операциями.

Таким образом, *проблема* использования корректирующих кодов заключается в следующем:

1. Для обеспечения отказоустойчивости ЗУ специализированных ЭВМ наиболее целесообразно использовать корректирующие коды.

2. Для защиты ТККС, работающих в реальном масштабе времени, могут быть использованы *только линейные коды*.

3. Минимальные аппаратурные затраты на построение декодирующего устройства достигаются при использовании низкоплотных линейных кодов;

4. В настоящее время *не известны* эффективные методы построения линейных кодов, корректирующих больше двукратных ошибок.

5. При использовании корректирующих линейных кодов аппаратурные затраты на коррекцию одиночной ошибки составляют 30% относительно исходного ЗУ, двукратной – 100%, при коррекции ошибки большей кратности $C_{дек} \gg C_{исх}$ (возникает проблема “сторожа над сторожем”).

6. Возникновение ошибок, кратность которых превышает корректирующие возможности кода, приводит к ошибочной коррекции.

7. Существующие корректирующие коды используются, как правило, только для защиты устройств хранения и передачи информации (ОЗУ, ПЗУ), а разработанные *адаптированы* для исправления ошибок преобразователей информации.

Основные научные результаты:

1. Разработан модифицированный итеративный код, позволяющий: корректировать ошибки трехкратные ошибки в полубайте информации, при этом обеспечивается возможность: исправлять ошибки различной конфигурации (имеет свойства нелинейного кода); иметь минимальные временные затраты на декодирование (в отличие от кодов Рида-Соломона реализующих процедуру циклического декодирования: исключить влияние неисправного резервного оборудования на работу устройств ЭВМ при наличии ошибок в контрольных разрядах и отсутствии ошибок в информационных; сигнализировать о неисправности устройства памяти при возникновении некорректируемой ошибки.
2. Выбран комплекс защиты информации для корпоративных информационно-телекоммуникационных сетей, с учетом исследованных и предложенных методик кодирования.
3. Предложена методика обеспечения отказоустойчивости сумматора на основе корректирующих линейных кодов.
4. Разработаны подходы к обнаружению и коррекции ошибок арифметических операций функционального ядра КСОН.

5. Разработаны методики и структуры функционально-кодовой защиты процессора при выполнении логических операций.
6. Разработаны функциональные схемы отказоустойчивого процессора повышенной достоверности функционирования с использованием ПЛИС, которые были внедрены и показали хорошие результаты.

СПИСОК ЛИТЕРАТУРЫ

1. Авакян А.А., Искандаров Р.Д. Супернадёжный высокопроизводительный вычислитель для бортовых автоматов// Информационные технологии в проектировании и производстве. 1996, вып.1- 2. С. 24-32.
2. Александрович А.Е. Разработка методов и средств обеспечения и анализа надёжности отказоустойчивых вычислительных систем./Диссертация на соискание ученой степени к.т.н. 61: 96- 5/ 807-х, М.: 1994, 163 с.
3. Амато В. Основы организации сетей Cisco, том 1.-М.; Издательский дом "Вильямс", 2002, 512 с.
4. Анохин А.В., Бояринов И.М., Давыдов А.А. и др. Исправление двойных и обнаружение тройных ошибок в полупроводниковой памяти ЭВМ.// Вопросы кибернетики. Проблемы создания высокопроизводительных ЭВМ. М.: ВИНТИ 1984, С. 50-66.
5. Афанасьев В.Б. Сложность декодирования кодов Рида-Соломона // Тр. 4-го Международного сим. по теории информации. М., 1976. Ч. 2. С. 13-18.
6. Афанасьев В.Б., Безроднов В.И., Давыдов А.А. Исследование корректирующих кодов для контроля дисплея // Помехоустойчивое кодирование и надёжность ЭВМ. М.: Наука. 1987, С. 151-186.
7. Барановская Т.П., Лойко В.И., Семенов М.И., Трубилин А.И. Архитектура компьютерных систем и сетей. - М.: Финансы и статистика, 2003, 256 с.
8. Берлекэмп Э. Алгебраическая теория кодирования. М.: Мир, 1971, 346 с.
9. Блейхут Р. Теория и практика кодов, контролирующих ошибки. М.: Мир, 1986, 576 с.
10. Блох Э.Л., Зяблов В.В. Линейные каскадные коды. М.: Наука, 1982, 350 с.
11. Блох Э.Л., Зяблов В.В. Обобщенные каскадные коды. М.: Связь, 1976, 410 с.
12. Боровиков С.М. Теоретические основы конструирования, технологии и надёжности. - Мн.: Дизайн ПРО, 1998. - 335 с.
13. Бройдо В. Вычислительные системы, сети и телекоммуникации./ Учебник для вузов. 3-е изд.- СПб.: Питер, 2008, -768 с.
14. Бройдо В.Л. Вычислительные системы, сети и телекоммуникации./ Учебник для вузов. 2-е изд. - СПб.: Питер, 2006, - 703 с.
15. Галатенко В.А. Основы информационной безопасности: курс лекций: учеб. Пособие / Издание третье / Под ред. академика РАН В.Б. Бетелина - М.: ИНТУИТ.РУ «Интернет-университет Информационных Технологий», 2006. - 208 с.
16. Галатенко В.А. Стандарты информационной безопасности / Под ред. Академика РАН В.Б.Бетелина - М.: ИНТУИТ.РУ «Интернет-университет Информационных Технологий», 2004. - 328 с.
17. Галкин А. П. Целесообразность информационной защиты предприятия./ Материалы 3-ей Международной НТК «Перспективные технологии в средствах передачи информации», г. Владимир, 1999, с.64-67.
18. Галкин А. П. Зависимость эффективности сети связи от срывов. / Материалы

- 4-ой Международной НТК «Перспективные технологии в средствах передачи информации», г. Владимир-Суздаль, 2001, с.72-77.
19. Галкин А. П. Оценка необходимости защиты информации предприятия. «Вестник ассоциации Русская оценка», 1999-1, с.55-58.
20. Галкин А.П. Защита каналов связи предприятий и учреждений от несанкционированного доступа к информации: Учебное пособие. – Владимир.: Владимирский государственный университет, 2003. – 128 с.
21. Галкин А.П. и др. Анализ угроз информационной безопасности в АСУТП и основные мероприятия по их предотвращению. //Научно-технический журнал «Известия института инженерной физики».2009. №3. С. 2-3.
22. Галкин А.П. Проектирование эффективной сети связи с учетом срывов. \ Проектирование и технология электронных средств.-2003-№1, с.9-11.
23. Галкин А.П., Аркадьева М.С. Анализ экономической безопасности предприятия и некоторые мероприятия по ее улучшению./ Вестник государственного университета управления, №8(18), 2008,-С. 54-57.
24. Гарбузов Н.И., Абашкин А.Л. Модификация корректирующих кодов для запоминающих устройств с параллельно-последовательной передачей информации. // Девятая всесоюз. Конф. По теории кодирования и передачи информации. Тез. докл. ч.1. Одесса, 1988. С. 242-245.
25. Гост 20.911-89, Техническая диагностика. Основные термины и определения. – М.: Стандарты, 1990.
26. Гост 27.002-89, Надежность в технике. Термины и определения. – М.: Стандарты, 1989.
27. Гришин В.Ю., Зубов Н.Н., Смаглий А.М. Принципы построения перспективных отказоустойчивых бортовых вычислительных систем управления космическими аппаратами с длительным сроком активного существования. // Сборник докладов международной НТК " Актуальные проблемы анализа и обеспечения надежности и качества приборов, устройств и систем", Пенза, 1998, С. 22-23.
28. Давыдов А.А., Дрожжина-Лабинская А.Ю. Дополнительные корректирующие возможности кодов БЧХ, исправляющих двойные и обнаруживающие тройные ошибки. // Вопросы кибернетики. Комплексное проектирование элементарно-конструкторской базы супер-ЭВМ. М.: ВИНТИ, 1988. С. 86-112.
29. Дементьев В.А. Комплексное проектирование систем управления и контроля ЛА. М.: Машиностроение, 1980. 256 с.
30. Дементьев В.А., Крылов Л.Н., Осипов В.П. и др. Теория и синтез дискретных автоматов. М.: МО СССР, 1979. 379 с.
31. Денисова А., Вихарев И., Белов А., Наумов Г. Интернет. 2-е изд. – СПб. Питер. 2004.- 368 с.
32. Дж. Уолренд Телекоммуникационные и компьютерные сети. М.: Постмаркет. 2001, 480 с.
33. Доманицкий С.М. Построение надежных логических устройств. М.:

Энергоатомиздат, 1986, 480 с.

34. Домарев В.В. Безопасность информационных технологий. Методология создания систем защиты. Изд.-ТИД Диа Софт. 2002. – 688 с.

35. Дружинин Г.В. Надежность автоматизированных производственных систем. М.: Энергоатомиздат, 1986, 480 с.

36. Емелин Н.М., Новиков Н.Н., Павлов А.А. и др. Принцип агрегатно-модульной адаптации интеллектуальных систем к контролю технического состояния сложных объектов. // Измерительная техника. 1999, № 5. С. 43-46.

37. Зайцев Г.В., Зиновьев В.А., Семаков Н.В. Коды с минимальной плотностью проверок для исправления байтовых ошибок, стираний, дефектов. // Проблемы передачи информации. 1983. Т. 19. № 3. С. 29-37.

38. Закон Российской Федерации "Об оперативно-розыскной деятельности в РФ", 1992.

39. Защита от несанкционированного доступа к информации. Термины и определения. Руководящий документ Гостехкомиссии России. М.: Военное издательство, 1992.

40. Зыль С.Н. Повышение отказоустойчивости сетевых приложений реального времени./ Сети и системы связи. 2005, №6. с 33 -37.

41. Иванов М.А., Кларин А.П. Сигнатурный анализ в задачах контроля и диагностики цифровых устройств. М.: Изд. МИФИ, 1986, 26 с.

42. Иуэду К.А. Надежность, контроль и диагностика вычислительных машин и систем. М.: Высшая школа, 1989, 215 с.

43. Каган Б.М., Мкртумян И.Б. Основы эксплуатации ЭВМ. М.: Энергоатомиздат, 1988, 430 с.

44. Казарин О.В., Лагутин В.С., Петраков А.В. Защита достоверных цифровых электрорадио сообщений. Учебное пособие. -М,: РИО МГУ СИ, 1997. – 68 с.

45. Калинин Ю.К. Криптозащита сообщений в системах связи. Учебное пособие.- М.: МТУСИ, 2000.- 236 с.

46. Кнопелько В.К., Лосев В.В. Надежное хранение информации в полупроводниковых запоминающих устройствах. М.: Радио и связь, 1987, 238 с.

47. Лонгботом Р. Надежность вычислительных систем. М.: Энергоатомиздат, 1985, 284 с.

48. Мак-Вильямс Ф.Дж., Слоэн Н. Дж. Теория кодов, исправляющих ошибки. М.: 1979. 156 с.

49. Максимов Н.В. Компьютерные сети. М.: изд. Форум, 2007, 448 с.

50. Морелос-Сарагоса Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. – М.: Техносфера, 2005. – 320 с.

51. Мур М, Притск Т., Риггс К., Сауфвик П. и др. Телекоммуникации. СПб.: БХВ – Петербург, 2005. – 624 с.

52. Мырова Л.О., Попов В.Д. Анализ стойкости систем связи к воздействию излучений. М.: Радио и связь, 1993. С.21-28.

53. Мырова Л.О., Чпиженко А.З. Обеспечение стойкости аппаратуры связи к ионизирующим и электромагнитным излучениям. М.: Радио и связь, 1988. 296 с.
54. Обухов В.Е., Павлов В.В. Синтез избыточных дискретных устройств с реконфигурацией структуры. Киев: Наукова думка, 1979, 154 с.
55. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 2-е изд. СПб. Питер, 2004– 864 с.
56. Павлов А.А. Повышение достоверности функционирования микропроцессорных средств измерительной техники. // Измерительная техник. 1999, №4. С. 28–33.
57. Павлов А.А. Повышение достоверности функционирования устройств памяти ЭВМ на основе использования корректирующих кодов с апостериорной коррекцией ошибок. // КомпьюЛог. 1998, № 5,6 (29,30) С. 6–9.
58. Павлов А.А., Гориш А.В., Милов Ю.Г. Метод защиты памяти ЭВМ на основе корректирующих кодов с апостериорной коррекцией ошибок // Экология, мониторинг и рациональное природопользование. Научн. тр. Вып. 302(11)-М.: МГУЛеса, 1999, С.259–264.
59. Павлов А.А., Кузнецов А.Н. Метод построения отказоустойчивых дискретных устройств на основе корректирующих кодов повышенной обнаруживающей способности. // КомпьюЛог. 1998, № 4(28) С. 49–51.
60. Павлов А.А., Павлов А.А. Концептуальные основы построения отказоустойчивых запоминающих устройств с апостериорной коррекцией ошибок. // КомпьюЛог. 1999, № 1(31) С. 34–37.
61. Пархоменко П.П. Основы технической диагностики. Кн. 1.-М.: Энергия, 1976. 464 с.
62. Петраков А.В. Основы практической защиты информации М.: Радио и связь, 1999.- 368 с.
63. Петров В.М. Рассмотрение основных показателей радиационной стойкости, позволяющих анализировать безотказность микропроцессорных и транспьютерных устройств при радиационном воздействии // Сборник докладов международной НТК "Актуальные проблемы анализа и обеспечения надежности и качества приборов, устройств и систем", Пенза, 1998, С. 325–327.
64. Питерсон У., Уэлдон Э. Коды, исправляющие ошибки. М.: Мир, 1976, 380 с.
65. Повалев Э.И., Щербаков Ю.И. Аппаратно-ориентированный метод решения системы уравнений двоичных кодов БЧХ для процедуры параллельной коррекции трехкратных ошибок. // Автоматика и вычислительная техника. 1987. № 3. С. 66–71.
66. Половко А.М. Основы теории надежности. М.: Наука, 1964, 356 с.
67. Пятибратов А.П., Гудыно Л.П., Кириченко А.А. Вычислительные системы сети и телекоммуникации./ Учебник для ВУЗов. М.; Финансы и статистика, 2003, 560 с.
68. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях,-М.:Радио и связь,1999.-328 с.

69. Руководство по поиску неисправностей в объединенных сетях Cisco Systems. М.: Издательский дом "Вильямс", 2003, 1040 с.
70. Рутковская Д., Пилинский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы. М. Горячая линия – Телеком, 2004.
71. Сагалович Ю. Л. Введение в алгебраические коды: Учебное пособие. – М.: МФТИ, 2007. – 262 с.
72. Сагалович Ю.Л. Кодовая защита оперативной памяти ЭВМ от ошибок.// Автоматика и телемеханика, 1991, № 5, С. 4-40.
73. Самофалов К.Г., Корнейчук В.И., Городний А.В. Структурно-логические методы повышения надежности запоминающих устройств. М.: Машиностроение, 1976, 350 с.
74. Сапожников В.В., Сапожников В.В, Методы синтеза надежных автоматов. Л.:Энергия, 1980, 94 с.
75. Согомонян Е.С., Слабоков Е.В. Самопроверяемые устройства и отказоустойчивые системы. М.: Радио и связь, 1989, 207 с.
76. Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации. Термины и определения. Руководящий документ Гостехкомиссии России. М.: Военное издательство, 1992.
77. Суворов А.В. Телекоммуникационные системы, компьютерные сети и Интернет./Учебное пособие для вузов. М.: Феникс, 2007, 384 с.
78. Тахаан Осама, А.П. Галкин, Аль-Муриш Мохаммед, Е.Г.Суслова. Финансовая и информационная безопасность и риски при проектировании./ Экономические проблемы ресурсного обеспечения инновационного развития региона. Материалы международной НК. Владимир, 2009. С. 112-118.
79. Тахаан Осама, Галкин А.П. Выбор комплекса защиты информации для корпоративных информационно-телекоммуникационных сетей./ Известия института инженерной физики. 2010-№2. С. 2-6.
80. Тахаан Осама, Галкин А.П. Информационная защита корпоративной сети системой обнаружением атак с нечеткой логикой./ Известия института инженерной физики. 2009-№4(14). С. 2-4.
81. Тахаан Осама, Галкин А.П. Повышение отказоустойчивости транспортного уровня вычислительных сетей путём реорганизации сквозной «точка-точка» множественной адресации./ Перспективные технологии в средствах передачи информации. Материалы 9-й Международной НТК. Владимир, 2011. С. 123-125.
82. Тахаан Осама, Галкин А.П. Угрозы информационной безопасности и защита от них для телекоммуникационных сетей радиосистем./ Проектирование и технология электронных средств. № 2. 2010. С. 28-30.
83. Тахаан Осама, Галкин А.П., Аль-Муриш Мохаммед. Обнаружения атак и нарушений в корпоративной сети./ Перспективные технологии в средствах передачи информации. Материалы 8-й Международной НТК. Владимир, 2009. С. 184-188.

84. Тахаан Осама, Галкин А.П., Аркадьева М.С. Кризис, безработица и информационная безопасность предприятия./ Экономические проблемы ресурсного обеспечения инновационного развития региона. Матер.международ. научн. конф. Владимир, 2009. С.119-122.
85. Тахаан Осама, Дерябин А.В., Дерябин В.М. Поэлементная или комплексная информационная защита./ Перспективные технологии в средствах передачи информации. Материалы VIII международная научно-техническая конференция, конф. Том 2. Владимир, 2009, С. 188-192.
86. Терминологический словарь «Бизнес-Безопасность-Телекоммуникации» - Учебное пособие / Составители А.А.Аржанов, Е.Г.Новикова, А.В.Петраков, С.В. Рабовский.- М.: РИО МТУСИ, 2000.- 304 с.
87. Толстяков В.С. Обнаружение и исправление ошибок в дискретных устройствах. М.: Советское радио,1972, 288 с.
88. Халяпин Д. Б., Ярочкин В. И. Основы защиты промышленной и коммерческой информации. Термины и определения. М.: ИПКИР, 1994, 231 с.
89. Хестер Н. Frontpage 2002 для Windows: Пер. С англ. - М.: ДМК Пресс, 2002. - 448с.
90. Хетагуров Я.А., Руднев Ю.Л. Повышение надежности цифровых устройств методами избыточного кодирования. М.: Энергия,1974, 370 с.
91. Хорев А.А. Защита информации от утечки по техническим каналам. Часть 1. Технические каналы утечки информации. Учебное пособие.М.:Гостехкомиссия РФ, 1998-320 с.
92. Хорев А.А. Способы и средства защиты информации.- М.: МО РФ,2001.-316с.
93. Шеннон К. Работы по теории информации и кибернетике. -М.: ИИЛ, 1963.- 829 с.
94. Щербаков Н.С. Достоверность работы цифровых устройств. М.: Машиностроение, 1989, 224 с.
95. Щербаков Н.С. Самокорректирующиеся дискретные устройства. М.: Машиностроение, 1975, 214 с.
96. Яблонский С. В. Введение в дискретную математику. М.: Наука,1979, 272 с.
97. Aichelmann F. J.Jr. Local paging memory buffer for minimizing Concurrence of hard and soft data errors. // IBM Tech. Disclosure Bull. 1980.№ 11. P. 4931-4932.
98. Blake J. B. and Mandel R. On-orbit observations of single event upset in Harris HM-6508 1K RAMs.// IEEE Trans. Nucl. Sci. 1989. NSo33, 1616-1619.
99. Blaum M. Systematic unidirectional burst detecting codes. //IBM RJ 5662 (57161), May 1987.
100. Bose B. On systematic SEC/MUED code. // Proc. FTCS. June 1981.V.11 P. 265-267.
101. Bose B. Systematic unidirectional error - detecting codes. //IEEE Trans. Computer. Nov. 1985. V.c-34 P. 1026-1032.

102. Chen C.L. Error correcting codes with Byte Error detection capability //IEEE Trans. Computer. July 1983. V.c-32 p. 615-621.
103. Libson M. R., Harvey H.E. A general-purpose memory reliability simulation. // IBM J. Res. Develop. 1984/ v.28. P. 196-206.
104. Meyer J.F., Wei L. Influence of workload on error recovery in random access memories. // IEEE Trans. Computer. 1988. V.37.№4. P, 500-507.
105. Nrener R., Fujiwara E., Agarwal V.K. Low Overhead, High Coverage Built-in Self-Test PLA Desidn/FTCS-15. 1985. P. 37-41.

Функционально–кодовая защита на основе итеративных кодов

Коррекция ошибок заданной кратности, при условии обнаружения ошибок в остальных разрядах информации, может быть обеспечена на основе итеративного кода.

Процедура построения двумерного итеративного кода состоит в следующем. Заданную совокупность информационных символов делят на группы (блоки, модули) информации, по b - разрядов в каждой группе (b – любое число позволяющее разделить исходную совокупность символов без остатка). Полученные модули информации представляют в виде информационной матрицы (1):

$$\begin{array}{cccc} y_{11} & y_{12} & \dots & y_{1b} \\ y_{21} & y_{22} & \dots & y_{2b} \\ \dots & \dots & \dots & \dots \\ y_{m1} & y_{m2} & \dots & y_{mb} \end{array} \quad (1)$$

Затем осуществляется кодирование информации по методу четности (путем сложения по $\text{mod } 2$ символов строк и столбцов полученной матрицы). В результате имеем двумерный итеративный код, позволяющий обнаруживать и исправлять любую одиночную ошибку:

$$\begin{array}{cccc} y_{11} & y_{12} & \dots & y_{1b} & h_1 \\ y_{21} & y_{22} & \dots & y_{2b} & h_2 \\ \dots & \dots & \dots & \dots & \dots \\ y_{m1} & y_{m2} & \dots & y_{mb} & h_m \\ \hline & z_1 & z_2 & \dots & z_b \end{array} \quad (2)$$

где $H = h_1, h_2, \dots, h_m$ - вектор четности строк; $Z = z_1, z_2, \dots, z_b$ - вектор четности столбцов. Вектора четности строк и столбцов образуют совокупность контрольных разрядов $R_1 = \{r_1, r_2, r_m, r_{m+1}, \dots, r_{m+b}\}$. При получении кодовой комбинации относительно информационных разрядов повторно формируется значения контрольных разрядов $R_1'' = \{r_1'', r_2'', r_m'', r_{m+1}'', \dots, r_b''\}$. В данном случае,

разница между переданными значениями контрольных разрядов и полученными после приема информации образует синдром ошибки E :

$$\begin{array}{r} R^{\Pi} = r_1^{\Pi} r_2^{\Pi} \dots r_m^{\Pi} r_{m+1}^{\Pi} \dots r_{m+b}^{\Pi} \\ \oplus R = r_1 r_2 \dots r_m r_{m+1} \dots r_{m+b} \\ \hline E = e_1 e_2 \dots e_m e_{m+1} \dots e_{m+b} \end{array} \quad (3)$$

При этом, разряды синдрома ошибки $e_1 e_2 \dots e_m$ (полученные относительно вектора четности строк) указывают модуль информации, имеющей ошибку, а разряды $e_m e_{m+1} \dots e_{m+b}$ (полученные относительно вектора четности столбцов) указывают ошибочный разряд в модуле информации.

Так как комбинации строк и столбцов имеют минимальное кодовое расстояние $d = 2$, то минимальное расстояние данного кода $d = 4$. Этот код позволяет исправлять любую одиночную ошибку и обнаруживать значительную долю кратных ошибок.

Структуры ошибок, не обнаруживаемых двумерным итеративным кодом показаны на рис. 1 и рис. 2.:



Рис. 1. Структуры ошибок, не обнаруживаемых двумерным итеративным кодом: а) - ошибки кратности 4; б) - ошибки кратности 6.

В общем случае можно строить итеративные коды более высокой размерности (трехмерные, четырехмерные и т.д.), где каждый информационный символ будет являться компонентой одновременно x различных кодовых слов. Параметры итеративных кодов размерности x таковы [5]:

$$n = \prod_{i=1}^x n_i, \quad k = \prod_{i=1}^x k_i, \quad d = \prod_{i=1}^x d_i, \quad (4)$$

Результаты правых диагональных проверок образуются при суммировании значений следующих информационных разрядов:

$$\begin{aligned} d_1'' &= y_{1,1} \\ d_2'' &= y_{1,2} \oplus y_{2,1} \\ d_3'' &= y_{3,1} \oplus y_{2,2} \oplus y_{1,3} \\ &\dots \\ d_l'' &= y_{m,b} \end{aligned} \quad (6)$$

В этом случае, общее число диагональных проверок равно $2l$, или :

$$R_{\text{ДИАГ}} = 2(b + m - 1). \quad (7)$$

Пример 1. Пусть рассматриваемое слово состоит из четырех информационных разрядов, которые имеют нулевые значения. Для данного кодового набора информационная матрица имеет вид:

$$\begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$$

В этом случае проверки на четность строк и столбцов информационной матрицы дадут нулевые значения и, кроме этого, будут иметь нулевые значения результаты всех правых и левых диагональных проверок. При возникновении ошибки во всех информационных разрядах имеем четную ошибку не обнаруживаемую двумерным итеративным кодом, т.к. проверки на четность строк и столбцов информационной матрицы имеют нулевые

значения:

$$\begin{matrix} 1^* & 1^* \\ 1^* & 1^* \end{matrix}$$

В то же время правые и левые диагональные проверки дадут результат 101.

Утверждение 1. Итеративный код, реализующий правые и левые диагональные проверки, обнаруживает все четные ошибки не обнаруживаемые двумерным итеративным кодом и выявляет нечетные ошибки воспринимаемые двумерным итеративным кодом как корректируемые.

В свою очередь существуют структуры ошибок не обнаруживаемые итеративным кодом, реализующим правые и левые диагональные проверки и проверками на четность строк и столбцов. Структуры рассматриваемых ошибок представлены на рис. 4.

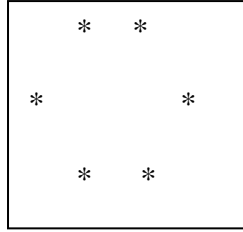


Рис. 4. Структуры ошибок, не обнаруживаемых диагональными проверками и проверками строк и столбцов

Так, например, относительно информационной матрицы, имеющей нулевые значения, диагональными проверками не будет обнаружена

следующая структура ошибки.
 010
 101 .
 010

Для того, чтобы исключить появление рассматриваемых ошибок, информационная матрица должна содержать не более двух строк.

Утверждение 2 Для информационной матрицы $[b \times 2]$ итеративный код, реализующий правые и левые диагональные проверки, обнаруживает максимальное количество возможных ошибок (за исключением множества 2^k-1 запрещенных кодовых наборов, трансформируемых в разрешенные кодовые наборы).

Таким образом, при использовании итеративного кода, реализующего правые и левые диагональные проверки, кодовый набор передается в виде:

$$Y = y_1 y_2 \dots y_k r_1 r_2 \dots r_{2l} . \tag{8}$$

Например для четырех информационных разрядов, представленных информационной матрицей: $\begin{matrix} y_1 & y_2 \\ y_3 & y_4 \end{matrix}$ правые диагональные проверки дадут

результат: $r_1 = y_1; r_2 = y_2 \oplus y_3; r_3 = y_4$, а левые диагональные проверки дадут результат: $r_4 = y_3; r_5 = y_1 \oplus y_4; r_6 = y_2$.

Результат сложения значений сигналов контрольных разрядов переданных и полученных даст синдром ошибки:

$$\oplus \begin{matrix} R & = & r_1 & r_2 & \dots & r_{2l} \\ R'' & = & r_1'' & r_2'' & \dots & r_{2l}'' \\ E & = & e_1 & e_2 & \dots & r_{2l} \end{matrix} , \tag{9}$$

где разряды вектора ошибки r_1, r_2, \dots, r_l - соответствуют правым диагональным проверкам, $r_l, r_{l+1}, \dots, r_{2l}$ - левым и сформированным относительно полученных информационных разрядов; $r_1'', r_2'', \dots, r_{2l}''$ - значения полученных контрольных разрядов.

Свойство 1. Существуют такие конфигурации ошибок в информационных и контрольных разрядах, для которых синдромы ошибок имеют **одинаковые значения**.

Для различения данных ошибок, при формировании значений синдромов ошибок, организуются **дополнительные правые и левые диагональные проверки** при формировании синдрома ошибки (относительно контрольных разрядов выражения).

$$\begin{array}{ll}
 v_1 = r_1 & v_{2l+2} = r_1'' \\
 v_2 = r_2 \oplus r_1'' & v_{2l+3} = r_1 \oplus r_2'' \\
 v_3 = r_3 \oplus r_2'' & v_{2l+4} = r_2 \oplus r_3'' \\
 \dots & \dots \\
 v_{2l} = r_{2l} \oplus r_{2l-1}'' & v_{4l+1} = r_{2l-1} \oplus r_{2l}'' \\
 v_{2l+1} = r_{2l}'' & v_{4l+2} = r_{2l}
 \end{array} \quad . \quad (10)$$

Таким образом, каждой ошибке из множества ошибок $M = (2^n)k$ можно поставить в соответствие значение синдрома ошибки и значение дополнительных диагональных проверок.

Свойство 2. Каждой совокупности значений синдрома ошибок и значений дополнительных проверок соответствует подмножество ошибок различной конфигурации.

Следствие 1. Для различения ошибок, принадлежащих данным подмножествам, следует **ограничить кратность исправляемых ошибок** и **увеличить число контрольных разрядов** (увеличить число проверок на четность строк и столбцов информационной матрицы).

**Программная модель функционально-кодовой защиты ПЗУ
функционального ядра КСОН**

```
//-----**-----
function get_dp_pr(S:String):string;
var i,l,pol_1:integer;
begin
  l:=length(S);
  pol_1:=l div 2;
  Result:=S[1];
  for i:=2 to pol_1 do
  begin
    if ((s[pol_1+i-1]='0') and (S[i]='0'))
      or ((s[pol_1+i-1]='1') and (S[i]='1')) then Result:=Result+'0'
    else Result:=Result+'1';
  end;
  Result:=Result+S[1];
end;

//*****

function get_Ch_d_lev(S:String):string;
var i,pol_1:integer;
begin
  pol_1:=length(S) div 2;
  Result:=S[pol_1+1];
  for i:=2 to pol_1 do
  begin
    if ((s[pol_1+i]='0') and (S[i-1]='0'))
      or ((s[pol_1+i]='1') and (S[i-1]='1')) then Result:=Result+'0'
    else Result:=Result+'1';
  end;
end;
```

```

    Result:=Result+S[pol_1];
end;
//*****
function get_Ch_str(Stroka:string):string;
var i,j,k:integer;
begin
    result:="";
    k:=0;
    for i:=1 to length(Stroka) do
    begin
        If Stroka[i]='1' then k:=k+1;
    end;
    k:= k mod 2;
    result:=IntToStr(k);
end;
//*****
procedure TForm1.FormShow(Sender: TObject);
begin
    DM.Tbl32_Query.Open;
    DM.Table1.Open;
    DM.DopQuery.Open;
    Label3.Caption:=IntToStr(DM.Tbl32_Query.RecNo);
end;
//*****
function BinStringToInt64(BinString : string) : longint;
var
    i : integer;
    Num : longint;
begin
    Num := 0;

```

```

for i := 1 to length(BinString) do
  if BinString[i] = '1' then
    Num := (Num shl 1) + 1 else
    Num := (Num shl 1);
  Result := Num;
end;

//*****

function IntToBin(Value: LongInt;Size: Integer): String;
var
  i: Integer;
begin
  Result:="";
  for i:=Size-1 downto 0 do
    begin
      if Value and (1 shl i)<>0 then
        begin
          Result:=Result+'1';
        end
      else
        begin
          Result:=Result+'0';
        end;
      end;
    end;
end;

//*****

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  DM.Tbl_32.Close;
  dm.Tbl32_Query.Close;
  DM.VektorQuery.Close;

```

```

DM.DopQuery.Close;
DM.Table1.Close;
DM.V_New_Tbl.Close;
DM.VektorTbl.Close;
end;
/**
function sum_mod_2(s_prd,s_prm:string):string ;
var i:integer;
begin
  Result:="";
  for i:=1 to Length(s_prd) do
    begin
      Result:=Result+IntToStr((StrToInt(s_prd[i])+StrToInt(s_prm[i])) mod 2);
    end;
  end;
end;
/**
procedure TForm1.Button3Click(Sender: TObject);
var i,j,k,kk,kkk,kkkk,kkkkk,kkkkkk:integer;
    Kod_err,R_p_err,Temp_str,
    Ispr_kod:string;
begin
  if DM.V_New_Tbl.Active then DM.V_New_Tbl.Close;
  DM.V_New_Tbl.EmptyTable;
  DM.V_New_Tbl.Open;
  Ispr_kod:='000000000000';
  Err_Create(Ispr_kod);
  for kk:=0 to 11 do
    begin
      Ispr_kod:=IntToBin(Round(Power(2,kk)),12);
      Err_Create(Ispr_kod);

```

```

end;
Ispr_kod:="";
for i:=1 to 11 do
for j:=i+1 to 12 do
begin
for k:=1 to 12 do
If (k<>i) and (k<>j) then Ispr_kod:=Ispr_kod+'0'
else Ispr_kod:=Ispr_kod+'1';
Err_Create(Ispr_kod);
Ispr_kod:="";
end;
Ispr_kod:="";
for i:=1 to 10 do
for j:=i+1 to 11 do
for kk:=j+1 to 12 do
begin
for k:=1 to 12 do
If (k<>i) and (k<>j) and (k<>kk) then Ispr_kod:=Ispr_kod+'0'
else Ispr_kod:=Ispr_kod+'1';
Err_Create(Ispr_kod);
Ispr_kod:="";
end;
end;
end;
//*****
procedure Err_Create(Ispr_kod:string);
var i,j,k,kk,t:integer;
V,e,chet_d_lev,chet_d_pr,R_sf,R_p,R_dop,Kod_err,R_p_err,
Pol_Str_1,Pol_Str_pr,R_p_Inv,Isch_Str,Str_tmp,Kontr_kod,R_kk,tmp_vse:string;
begin

```



```

Kod_err:=Copy(Ispr_kod,1,4);
R_p_err:=Copy(Ispr_kod,5,8);
for k:=0 to 15 do
begin
  Isch_Str:=IntToBin(k,4);
  Str:=Copy(Isch_Str,1,2);
  Pol_Str_1:=Str+get_Ch_str(1);
  Str:=Copy(Isch_Str,3,2);
  Pol_Str_pr:=Str+get_Ch_str(1);
  Str:=Pol_Str_1+ Pol_Str_pr;
  chet_d_pr:=get_Ch_d_pr(2);
  chet_d_lev:=get_Ch_d_lev(2);
  tmp_vse:="";
  for i:=length(chet_d_lev) downto 1 do
  begin
    tmp_vse:=tmp_vse+chet_d_lev[i];
  end;
  chet_d_lev:=tmp_vse;
  R_p:=chet_d_pr+chet_d_lev;
  R_p:=sum_mod_2(R_p,R_p_err);
  Isch_Str:=sum_mod_2(Isch_Str,Kod_err);
  Str:=Copy(Isch_Str,1,2);
  Pol_Str_1:=Str+get_Ch_str(1);
  Str:=Copy(Isch_Str,3,2);
  Pol_Str_pr:=Str+get_Ch_str(1);
  Str:=Pol_Str_1+ Pol_Str_pr;
  chet_d_pr:=get_Ch_d_pr(2);
  chet_d_lev:=get_Ch_d_lev(2);
  tmp_vse:="";
  for i:=length(chet_d_lev) downto 1 do

```

```

begin
  tmp_vse:=tmp_vse+chet_d_lev[i];
end;
chet_d_lev:=tmp_vse;
R_sf:=chet_d_pr+chet_d_lev;
e:=sum_mod_2(R_sf,R_p);
Str_tmp:=Str;
Str:=R_sf+R_p;
chet_d_pr:=get_Ch_d_pr(2);
chet_d_lev:=get_Ch_d_lev(2);
tmp_vse:="";
for j:=length(chet_d_lev) downto 1 do
begin
  tmp_vse:=tmp_vse+chet_d_lev[j];
end;
chet_d_lev:=tmp_vse;
R_dop:=chet_d_pr+chet_d_lev;
Str:=Str_tmp;
DM.V_New_Tbl.Edit;
DM.V_New_Tbl.Append;
DM.V_New_TblN_bait.AsInteger:=4;
DM.V_New_TblIspr_kod.AsString:=Ispr_kod;
DM.V_New_TblKod.AsString:=Isch_Str;
DM.V_New_TblR_sf.AsString:=R_sf;
DM.V_New_TblR_p.AsString:=R_p;
DM.V_New_TblE.AsString:=e;
DM.V_New_TblR_dop.AsString:=R_dop;
DM.V_New_Tbl.Post;
end;
end;

```

Акты внедрения



«Утверждаю»

Генеральный директор
ООО «Электронприбор» (г. Москва)

Н. И. Захарова


4 марта 2010 г.

Акт внедрения

Результаты, полученные **Осамой Тахаан** (гражданина Сирии) при выполнении диссертационной работы, внедрены на нашем предприятии в 2010 г. в виде расчетных методик, в частности, с учетом экономической целесообразности защиты информации. Высокий математический уровень подтверждается применением аппарата нечеткой логики и другими разработками.

Проведена проверка на наличие возможных путей проникновения в системы связи нашего предприятия.

Использованы рекомендации по защите компьютерных и телекоммуникационных сетей от несанкционированного доступа к информации.

Начальник информационного отдела -  **Иванов А.П.**Ведущий инженер-  **Андреев А.И.**

«Утверждаю»

Генеральный директор
НПО «РИК»,

А. В. Поляков

21.06.11

Акт внедрения

Результаты, полученные **Тахаан Осамой** (гражданина Сирии) при выполнении диссертационной работы, в частности:

- 1) Методики применения различных способов защиты информации от несанкционированного доступа (в том числе с разработанными им кодами и компьютерными программами);
- 2) Рекомендации по защите телекоммуникационных и компьютерных сетей;

внедрены на нашем предприятии в 2010-2011гг. Они нашли практическое применение при обмене информацией с нашими филиалами в гг. Иваново, Санкт-Петербург, Омске, в локальных сетях и т.п.

Указанные методики хороши тем, что при сравнительно небольших затратах обеспечивают высокую эффективность и не требуют специальной подготовки нашего персонала.

Начальник отдела-

Сирко С. Э.

Начальник лаборатории -

Смушко О.Л.