МЕЖГОСУДАРСТВЕННЫЙ СОВЕТ ПО СТАНДАРТИЗАЦИИ, МЕТРОЛОГИИ И СЕРТИФИКАЦИИ
(МГС)
INTERSTATE COUNCIL FOR STANDARDIZATION, METROLOGY AND CERTIFICATION
(ISC)

МЕЖГОСУДАРСТВЕННЫЙ СТАНДАРТ

ГОСТ 34332.5— 2021

БЕЗОПАСНОСТЬ ФУНКЦИОНАЛЬНАЯ СИСТЕМ, СВЯЗАННЫХ С БЕЗОПАСНОСТЬЮ ЗДАНИЙ И СООРУЖЕНИЙ

Часть 5

Меры по снижению риска, методы оценки

(IEC 61508-7:2010, NEQ)
(IEC 61508-4:2010, NEQ)

Издание официальное

Москва
Стандартинформ
2021

gost.e-ecolog.ru

Предисловие

Цели, основные принципы и общие правила проведения работ по межгосударственной стандартизации установлены ГОСТ 1.0 «Межгосударственная система стандартизации. Основные положения» и ГОСТ 1.2 «Межгосударственная система стандартизации. Стандарты межгосударственные, правила и рекомендации по межгосударственной стандартизации. Правила разработки, принятия, обновления и отмены»

Сведения о стандарте

1 ПОДГОТОВЛЕН Федеральным государственным унитарным предприятием «Российский научно-технический центр информации по стандартизации, метрологии и оценке соответствия» (ФГУП «СТАНДАРТИНФОРМ») совместно с Международной ассоциацией «Системсервис» (МА «Системсервис»)

2 ВНЕСЕН Федеральным агентством по техническому регулированию и метрологии

3 ПРИНЯТ Межгосударственным советом по стандартизации, метрологии и сертификации (протокол от 30 апреля 2021 г. № 139-П)

За принятие проголосовали:

Краткое наименование страны по МК (ИСО 3166) 004—97	Код страны по МК (ИСО 3166) 004—97	Сокращенное наименование национального органа по стандартизации
Армения	AM	ЗАО «Национальный орган по стандартизации и метрологии» Республики Армения
Беларусь	BY	Госстандарт Республики Беларусь
Киргизия	KG	Кыргызстандарт
Россия	RU	Росстандарт
Таджикистан	TJ	Таджикстандарт
Узбекистан	UZ	Узстандарт

4 Приказом Федерального агентства по техническому регулированию и метрологии от 28 мая 2021 г. № 478-ст межгосударственный стандарт ГОСТ 34332.5—2021 введен в действие в качестве национального стандарта Российской Федерации с 1 января 2022 г.

5 В настоящем стандарте учтены основные нормативные положения следующих международных стандартов:

IEC 61508-7:2010 «Функциональная безопасность электрических/электронных/программируемых электронных систем, связанных с безопасностью. Часть 7. Обзор методов и средств» («Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 7: Overview of techniques and measures», NEQ);

IEC 61508-4:2010 «Функциональная безопасность электрических/электронных/программируемых электронных систем, связанных с безопасностью. Часть 4. Термины и сокращения» («Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 4: Definitions and abbreviations», NEQ)

6 ВВЕДЕН ВПЕРВЫЕ

7 Настоящий стандарт подготовлен на основе применения ГОСТ Р 53195.5—2015[1)]

[1)] Приказом Федерального агентства по техническому регулированию и метрологии от 28 мая 2021 г. № 478-ст ГОСТ Р 53195.5—2010 отменен с 1 января 2022 г.

Информация о введении в действие (прекращении действия) настоящего стандарта и изменений к нему на территории указанных выше государств публикуется в указателях национальных стандартов, издаваемых в этих государствах, а также в сети Интернет на сайтах соответствующих национальных органов по стандартизации.

В случае пересмотра, изменения или отмены настоящего стандарта соответствующая информация будет опубликована на официальном интернет-сайте Межгосударственного совета по стандартизации, метрологии и сертификации в каталоге «Межгосударственные стандарты»

© Стандартинформ, оформление, 2021

В Российской Федерации настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

Содержание

1 Область применения 1
2 Нормативные ссылки 2
3 Термины и определения 2
4 Сокращения и обозначения 3
5 Меры по снижению риска 4
6 Методы оценки 4
Приложение А (справочное) Методы и средства для контроля случайных отказов аппаратных средств Э/Э/ПЭ СБЗС систем 5
Приложение Б (справочное) Методы и средства по предотвращению систематических отказов Э/Э/ПЭ СБЗС систем 18
Приложение В (справочное) Методы и средства достижения полноты безопасности программного обеспечения 34
Приложение Г (справочное) Методы и средства для проектирования специализированных интегральных схем 67
Приложение Д (справочное) Методы и средства для разработки связанного с безопасностью объектно-ориентированного программного обеспечения 78
Приложение Е (справочное) Вероятностный подход к определению полноты безопасности предварительно разработанного программного обеспечения 80
Приложение Ж (справочное) Определение свойств стадий жизненного цикла программного обеспечения 84
Библиография 89

Введение

Современные здания и сооружения (объекты капитального строительства) представляют собой сложные системы, в состав которых входит система строительных конструкций и ряд инженерных систем в разных сочетаниях, в том числе для жизнеобеспечения, реализации технологических процессов, энерго- и ресурсосбережения, обеспечения безопасности и другие системы. Эти системы взаимодействуют друг с другом, с внешней и внутренней средами, образуя единое целое при выполнении своих функций назначения.

Объекты капитального строительства жестко привязаны к местности. Рабочие характеристики зданий, сооружений и входящих в них систем могут быть реализованы, проверены и использованы только в том месте, в котором объекты построены и системы установлены.

Безопасность зданий и сооружений обеспечивается применением совокупности мер, мероприятий и средств снижения риска причинения вреда до приемлемого риска, и поддержания данного уровня в течение периода эксплуатации или использования этих объектов. К средствам снижения риска относятся системы, связанные с безопасностью зданий и сооружений (СБЗС системы). К таким системам относятся системы, неполный перечень которых представлен в ГОСТ 34332.1—2017 (приложение А, раздел А.3). Среди СБЗС систем наиболее распространенными являются системы, содержащие электрические и/или электронные, и/или программируемые электронные (Э/Э/ПЭ) компоненты. Такие системы, именуемые Э/Э/ПЭ СБЗС системами, в течение многих лет используют для выполнения функций безопасности. Наряду с ними используют системы, основанные на неэлектрических (гидравлических, пневматических, механических) технологиях, а также прочие средства уменьшения риска. Для решения задач безопасности зданий и сооружений во всех больших объемах используют программируемые электронные СБЗС системы.

Следующими по важности характеристиками систем, после характеристик назначения, являются характеристики безопасности. Важнейшей характеристикой безопасности систем признана их функциональная безопасность.

В настоящем стандарте установлен общий подход к вопросам обеспечения безопасности Э/Э/ПЭ СБЗС систем. Этот унифицированный подход был принят для разработки рациональной и последовательной практики для всех электрических систем обеспечения безопасности на основе стандартов серии IEC 61508.

В настоящем стандарте приведены краткие описания мер (методов и средств) по снижению риска и оценке соответствия на стадиях жизненного цикла Э/Э/ПЭ СБЗС систем, на которые даны ссылки в ГОСТ 34332.3—2021 и ГОСТ 34332.4—2021, а также приведены литературные источники с полным описанием таких мер.

Настоящий стандарт (совместно с ГОСТ 34332.1—ГОСТ 34332.4) ориентирован на обеспечение соблюдения требований безопасности зданий и сооружений, в том числе объектов транспортных инфраструктур, установленных техническими регламентами Таможенного союза [1]—[3], и в развитие базовых требований данных технических регламентов.

Настоящий стандарт входит в комплекс стандартов с общим наименованием «Безопасность функциональная систем, связанных с безопасностью зданий и сооружений» и является пятым стандартом этого комплекса «Часть 5. Меры по снижению риска, методы оценки». Другие стандарты, входящие в данный комплекс:

- часть 1. Основные положения;
- часть 2. Общие требования;
- часть 3. Требования к системам;
- часть 4. Требования к программному обеспечению;
- часть 6. Прочие средства уменьшения риска, системы мониторинга;
- часть 7. Порядок применения ГОСТ 34332, примеры расчетов.

Структура комплекса ГОСТ 34332 приведена на рисунке 1.

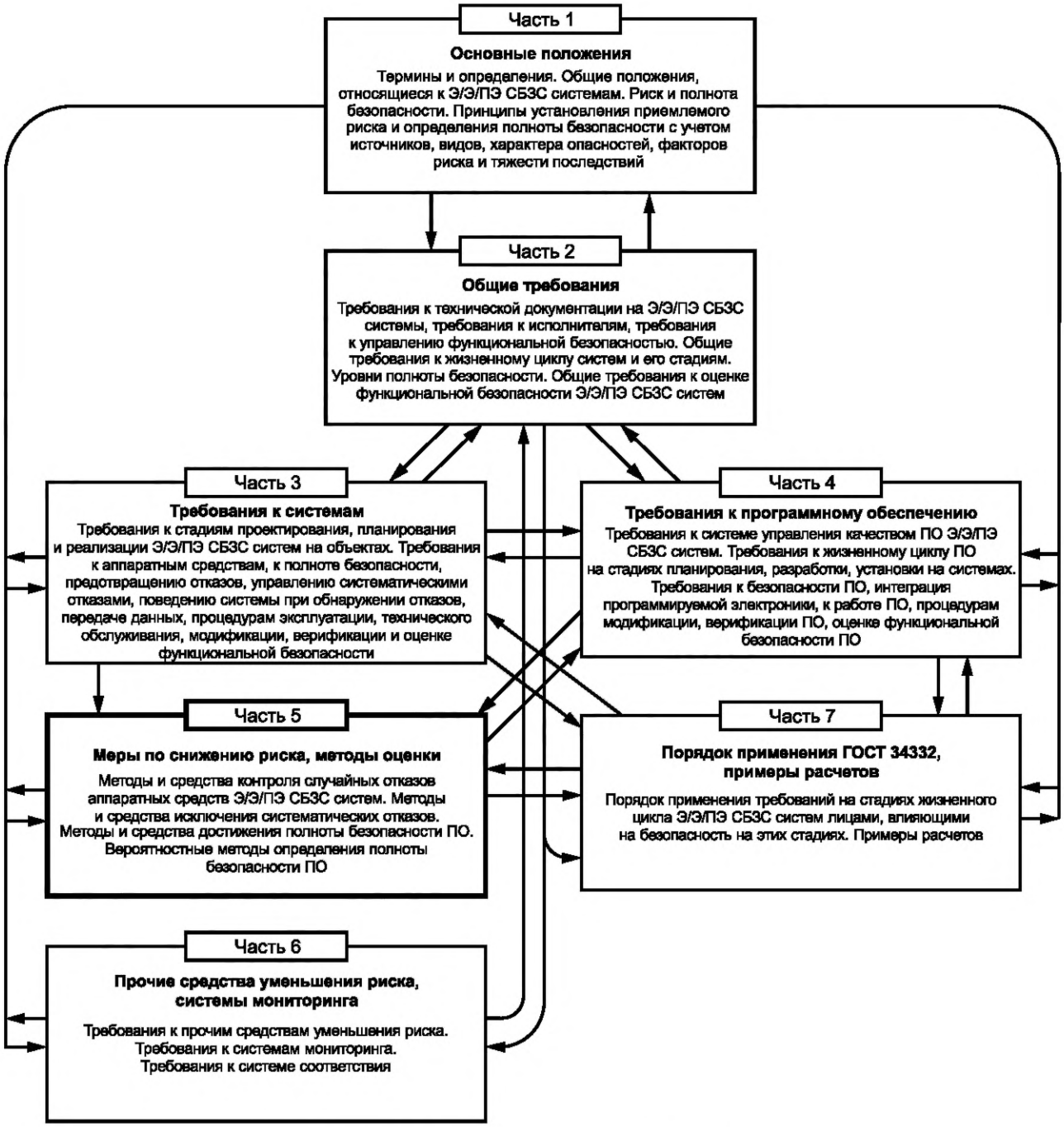


Рисунок 1 — Структура комплекса ГОСТ 34332

Поправка к ГОСТ 34332.5—2021 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 5. Меры по снижению риска, методы оценки

В каком месте	Напечатано	Должно быть		
Предисловие. Таблица согласования	—	Казахстан	KZ	Госстандарт Республики Казахстан

(ИУС № 4 2022 г.)

Поправка к ГОСТ 34332.5—2021 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 5. Меры по снижению риска, методы оценки

В каком месте	Напечатано	Должно быть
Предисловие. Таблица согласования	—	Азербайджан | AZ | Азстандарт

(ИУС № 8 2023 г.)

М Е Ж Г О С У Д А Р С Т В Е Н Н Ы Й С Т А Н Д А Р Т

БЕЗОПАСНОСТЬ ФУНКЦИОНАЛЬНАЯ СИСТЕМ, СВЯЗАННЫХ С БЕЗОПАСНОСТЬЮ ЗДАНИЙ И СООРУЖЕНИЙ

Часть 5

Меры по снижению риска, методы оценки

Functional safety of building/construction safety-related systems. Part 5. Measures for risk reduction, methods of assessment

Дата введения — 2022—01—01

1 Область применения

1.1 Настоящий стандарт:

- применяется к электрическим, электронным, программируемым электронным (Э/Э/ПЭ), связанным с безопасностью зданий и сооружений (СБЗС) системам и подсистемам (далее Э/Э/ПЭ СБЗС системы и подсистемы), их аппаратным средствам (АС) и программному обеспечению (ПО), установленным на объектах (зданиях и сооружениях) и являющихся их неотъемлемой частью.

Примечание — К Э/Э/ПЭ СБЗС системам относятся системы, перечисленные в ГОСТ 34332.1—2017 (приложение А, раздел А.2);

- применяется к Э/Э/ПЭ системам, подсистемам и средствам промышленного производства, которые используют в качестве комплектующих технических средств при проектировании и реализации Э/Э/ПЭ СБЗС систем на объекте.

Примечание — Под реализацией систем понимается их установка на объекте, монтаж, пусконаладка, оценка и подтверждение соответствия;

- охватывает полный жизненный цикл (ЖЦ) Э/Э/ПЭ СБЗС систем, начиная с общей концепции, включая проектирование, реализацию, эксплуатацию и техническое обслуживание систем вплоть до вывода их из эксплуатации и утилизации.

1.2 Настоящий стандарт:

- устанавливает цели основных методов/средств, используемых для выполнения требований ГОСТ 34332.3 и ГОСТ 34332.4, и методы оценки соответствия;
- содержит краткие описания методов/средств, рекомендуемых в ГОСТ 34332.3 и ГОСТ 34332.4 и применяемых на различных стадиях жизненных циклов СБЗС систем, их АС и ПО для снижения рисков, а также ссылки на источники с полным описанием этих методов/средств.

Примечание — Под «методами/средствами» в настоящем стандарте понимаются методы и/или средства. В большинстве методов/средств, описанных в приложениях А—Ж, метод состоит в применении того или иного аппаратного, программного или аппаратно-программного средства, или средств, в применении логических или математических действий (которые выполняются с использованием средств информатики и математики).

1.3 Настоящий стандарт не распространяется на одиночные Э/Э/ПЭ СБЗС системы, способные осуществить необходимое снижение риска и требуемая полнота безопасности которых ниже самого низкого уровня полноты безопасности (УПБ 1), определенного в ГОСТ 34332.2—2017 (таблицы 1 и 2). Он не распространяется также на здания и сооружения, оснащенные такими системами или не имеющие никаких связанных с безопасностью систем.

Издание официальное

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты:

ГОСТ 19.701 (ИСО 5807—85) Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения

ГОСТ 14254 (IEC 60529:2013) Степени защиты, обеспечиваемые оболочками (Код IP)

ГОСТ 34332.1—2017 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 1. Основные положения

ГОСТ 34332.2—2017 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 2. Общие требования

ГОСТ 34332.3—2021 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 3. Требования к системам

ГОСТ 34332.4—2021 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 4. Требования к программному обеспечению

ГОСТ ISO 9000 Системы менеджмента качества. Основные положения и словарь

ГОСТ ISO 9001[1)] Системы менеджмента качества. Требования

ГОСТ IEC 60255-5 Реле электрические. Часть 5. Координация изоляции измерительных реле и защитных устройств. Требования и испытания

ГОСТ IEC/TS 61000-1-2 Электромагнитная совместимость (ЭМС). Часть 1-2. Общие положения. Методология достижения функциональной безопасности электрических и электронных систем, включая оборудование, в отношении электромагнитных помех

ГОСТ IEC 61000-6-5 Электромагнитная совместимость (ЭМС). Часть 6-5. Общие стандарты. Помехоустойчивость оборудования, используемого в обстановке электростанции и подстанции

ГОСТ IEC 61082-1 Документы, используемые в электротехнике. Подготовка. Часть 1. Правила

ГОСТ IEC TR 61340-5-2 Электростатика. Защита электронных устройств от электростатических явлений. Руководство по применению

Примечание — При пользовании настоящим стандартом целесообразно проверить действие ссылочных стандартов и классификаторов на официальном интернет-сайте Межгосударственного совета по стандартизации, метрологии и сертификации (gost.e-ecolog.ru по указателям национальных стандартов, издаваемых в государствах, указанных в предисловии, или на официальных сайтах соответствующих национальных органов по стандартизации. Если на документ дана недатированная ссылка, то следует использовать документ, действующий на текущий момент, с учетом всех внесенных в него изменений. Если заменен ссылочный документ, на который дана датированная ссылка, то следует использовать указанную версию этого документа. Если после принятия настоящего стандарта в ссылочный документ, на который дана датированная ссылка, внесено изменение, затрагивающее положение, на которое дана ссылка, то это положение применяется без учета данного изменения. Если ссылочный документ отменен без замены, то положение, в котором дана ссылка на него, применяется в части, затрагивающей эту ссылку.

3 Термины и определения

В настоящем стандарте применены термины по ГОСТ 34332.1—ГОСТ 34332.4, а также следующие термины с соответствующими их определениями.

3.1 **антивалентные сигналы** (antivalent signals): Два сигнала с одинаковым информационным содержанием, один из которых передается по каналу связи в прямой форме, а другой — в инверсной форме (аналоговые сигналы — в противофазе, цифровые сигналы — с инверсией логических «0» в логические «1» или наоборот).

3.2 **константная неисправность** (stuck-at fault): Неисправность аппаратного средства, вызванная переходом элемента устройства в одно из неизменяемых состояний, например, при «залипании» (сварке) контактов реле.

3.3 **константный отказ** (stuck-at failure): Отказ аппаратного средства и/или программного обеспечения, приводящий к переходу аппаратного средства в одно из неизменяемых состояний и/или выдаче на выходе неизменяемых данных или неизменяемой(ых) команды (команд).

3.4 **постепенный отказ** (drift failure): Отказ аппаратного средства из-за постепенного выхода его характеристик за допустимые пределы.

[1)] В Российской Федерации действует ГОСТ Р ИСО 9001—2015.

3.5 **самоустраняющийся отказ** (transient failure): Отказ, обусловленный переходными процессами, устраняющийся по их завершении.

3.6 **условная тревога** (conditional alarm): Состояние, близкое к тревожному, но еще не влекущее опасных последствий.

Примечание — Термин относится к Э/Э/ПЭ СБЗС системам, в которых предусмотрено ступенчатое реагирование на постепенно развивающиеся тревожные события.

3.7 **чрезвычайное действие** (emergency action): Действие, требующее выполнения при возникновении чрезвычайной ситуации для снижения риска причинения вреда.

4 Сокращения и обозначения

В настоящем стандарте применены следующие сокращения и обозначения:

АГТП — автоматический генератор тестовых примеров;
АС СБЗС — аппаратное(ые) средство(а), связанное(ые) с безопасностью зданий и сооружений;
ЖЦ — жизненный цикл;
ИС — интегральная(ые) микросхема(ы);
КМОП — комплементарная структура металл—оксид—полупроводник;
ОЗУ — оперативное запоминающее устройство (устройство памяти с произвольным доступом);
ПЗУ — постоянное запоминающее устройство;
ППВМ — полевая программируемая вентильная матрица;
САПР — система автоматизированного проектирования;
СБС — связанная с безопасностью система;
СВА — статический временной анализ;
СИС — специализированная интегральная схема;
УО — управляемое оборудование;
УПБ — уровень полноты безопасности;
ФСЗ — формат стандартной задержки;
ФСЗ-файл — файл в формате стандартной задержки;
ЦПУ — центральное процессорное устройство;
Ada — язык программирования для встраиваемых систем, разработанный в 1979—1980 годах в США и названный в честь Ады Лавлейс;
ADT — данные абстрактного типа;
CASE — набор методов и средств программной инженерии для проектирования программного обеспечения;
CCS — метод/средство расчета соединяющихся систем;
CHAZOP — метод/средство анализа опасности и работоспособности систем управления;
CHAZOPs — метод/средство анализа опасности работоспособности компьютеров;
CIRCAL — метод/средство расчета критических цепей;
CORE — метод/средство выражения контролируемых требований;
CRC — циклический избыточный код коррекции ошибок;
CSP — метод/средство описания последовательных коммуникационных процессов;
DMA — прямой доступ к памяти;
FMEA — процедура анализа типа отказа и его последствий;
FTA — метод анализа на основе дерева отказов;
HAZOP — метод/средство анализа опасности и работоспособности;
HOL — наименование языка логики высшего порядка;
INMOS — наименование английской фирмы, специализирующейся на производстве транспьютеров;
JSD — наименование структурного метода разработки программных систем Джексона;
LCSAJ — обозначение последовательности линейного кода и перехода, применяемой при тестировании ПО;

LOTOS — язык для описания спецификаций, упорядоченных во временно́й области;
MCDC — охват решения модифицированными условиями;
MMU — блок управления памятью;
MTBF — среднее время наработки на отказ;
OBJ — язык для алгебраического описания спецификаций;
OCCAM — язык параллельного программирования высокого уровня, используемый для транспьютеров;
PROM — программируемое постоянное запоминающее устройство;
UML — унифицированный язык моделирования;
SADT — метод/средство структурного анализа и проектирования;
VDM — один из методов разработки компьютерных систем на основе формального языка;
VDM++ — расширенная версия метода VDM;
VDM-SL — формальный язык для описания спецификаций, разрабатываемых с использованием метода VDM;
Z — нотация языка для описания спецификаций последовательных систем.

5 Меры по снижению риска

5.1 Основными мерами (методами/средствами) по снижению риска являются:
- контроль случайных отказов АС Э/Э/ПЭ СБЗС систем;
- исключение систематических отказов на различных стадиях ЖЦ Э/Э/ПЭ СБЗС систем;
- методы/средства, реализуемые на различных этапах стадий ЖЦ для снижения риска и достижения полноты безопасности ПО Э/Э/ПЭ СБЗС систем.

5.2 Методы/средства для контроля случайных отказов АС Э/Э/ПЭ СБЗС систем, их краткое описание, а также ссылки на источники с подробным описанием приведены в приложении А.

5.3 Методы и средства для предотвращения систематических отказов на различных стадиях ЖЦ Э/Э/ПЭ СБЗС систем, их краткие описания, анализ, а также ссылки на источники с полным описанием приведены в приложении Б.

5.4 Методы и средства для достижения полноты безопасности ПО Э/Э/ПЭ СБЗС систем, реализуемые на различных этапах стадий ЖЦ ПО, их краткое описание, а также ссылки на источники с подробным описанием приведены в приложении В.

5.5 Краткое описание методов и средств для проектирования СИС приведено в приложении Г.

5.6 Краткое описание методов и средств для разработки связанного с безопасностью объектно-ориентированного ПО приведено в приложении Д.

6 Методы оценки

6.1 Методы оценки функциональной безопасности Э/Э/ПЭ СБЗС систем, ПО этих систем, их краткое описание, а также ссылки на источники с подробным описанием приведены в приложении В.

6.2 Методы оценки полноты безопасности предварительно разработанных программных средств, применяемых для Э/Э/ПЭ СБЗС систем, основанные на вероятностном подходе, приведены в приложении Г.

6.3 Вероятностный подход определения полноты безопасности предварительно разработанного ПО приведен в приложении Е.

6.4 Методы и средства по определению свойств стадий ЖЦ ПО приведены в приложении Ж.

Приложение А
(справочное)

Методы и средства для контроля случайных отказов аппаратных средств Э/Э/ПЭ СБЗС систем

А.1 Электромеханические компоненты

А.1.1 Обнаружение отказов посредством мониторинга в режиме онлайн

Цель — обнаружение отказов посредством мониторинга (в режиме онлайн) Э/Э/ПЭ СБЗС системы в процессе нормального функционирования управляемого оборудования.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.2 и А.14).

Описание. При определенных условиях отказы могут быть обнаружены с помощью информации о поведении УО во времени. Например, если переключатель, который является частью Э/Э/ПЭ СБЗС системы нормально активизируется УО и если при этом переключатель не изменяет состояния в предполагаемое время, то этот отказ может быть обнаружен. Обычными методами невозможно локализовать такой отказ.

А.1.2 Мониторинг контактов реле

Цель — обнаружение отказов [например, из-за сварки («залипания»)] контактов реле.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.2 и А.14).

Описание. Активизируемые контактные реле (или переключаемые контакты в реле) проектируют таким образом, чтобы их поводки контактов были жестко связаны между собой. Пусть имеются два набора переключаемых контактов *a* и *b* (рисунок А.1). Если нормально разомкнутый контакт *b* оказался приваренным («залипшим»), то нормально замкнутый контакт *a* не может замкнуться при обесточивании обмотки реле. Следовательно, контроль замыкания нормально замкнутого контакта *a* при обесточенной обмотке реле может быть использован для указания того, что нормально разомкнутый контакт *b* действительно разомкнут. Отказ замыкания нормально замыкаемого контакта *a* указывает на отказ контакта *b*. Таким образом, схема контроля обеспечивает надежное отключение или продолжение отключения при любом управлении оборудования контактом *b*.



Рисунок А.1 — Контакты реле

А.1.3 Компаратор

Цель — оперативное обнаружение (не одновременное) отказов в независимом модуле обработки или в компараторе.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.2—А.4).

Описание. Сигналы независимых модулей обработки (процессоров) сравнивают циклически или непрерывно с помощью компаратора (АС). Сам компаратор может быть внешне тестируемым или в нем может быть использована самоконтролируемая технология. При обнаружении компаратором различия в поведении процессоров независимых модулей формируется информация для сообщений об отказах. На рисунке А.2 представлена схема компаратора в двухканальной системе.

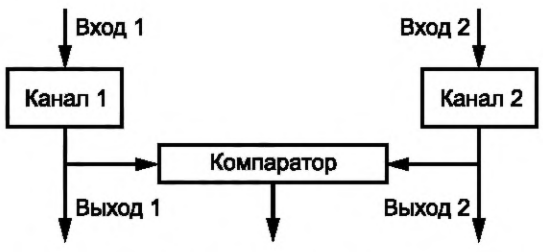


Рисунок А.2 — Компаратор в двухканальной системе

На рисунке А.3 представлена схема компаратора в одноканальной системе, реализуемая с использованием ПО. Сложная обработка осуществляется с помощью двух независимых наборов данных и прикладных программ. Поскольку существует только один блок обработки, двойная обработка производится последовательно в интервалах времени *Т*1, *Т*2, показанных на рисунке. Независимые выходные результаты проверяются программным компаратором в интервале времени *Т*3. При применении двух различных прикладных программ достигается высокий диапазон охвата отказов.

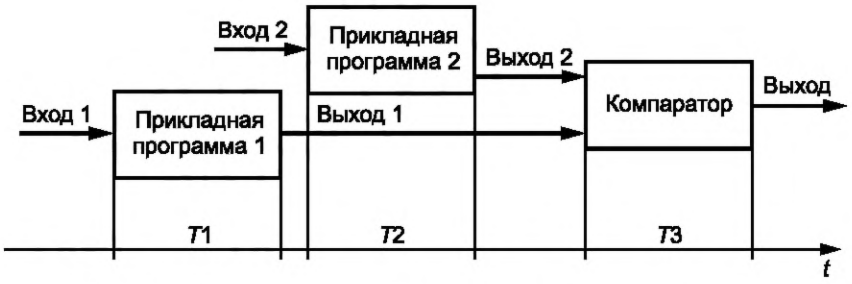


Рисунок А.3 — Компаратор одноканальной системы, реализуемый с помощью ПО

А.1.4 Схема голосования по мажоритарному принципу

Цель — обнаружение и парирование отказов по меньшей мере в одном из трех каналов АС.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.2—А.4).

Описание. Для обнаружения и маскирования отказов применяют модуль голосования, использующий мажоритарный принцип (2 из 3, 3 из 3 или *m* из *n*). Для работы схемы голосования может быть использовано внешнее тестирование, или в самой схеме могут быть использованы самоконтролируемые технологии. Схема голосования по мажоритарному принципу показана на рисунке А.4.



Рисунок А.4 — Голосование по мажоритарному принципу

Подробное описание данного метода/средства приведено в [4] и [5].

А.1.5 Отсутствие электропитания

Цель — выполнение функции безопасности при отключении или отсутствии электропитания.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица 16).

Описание. Функция безопасности выполняется, если контакты разомкнуты и ток не поступает в АС. Например, при использовании торможения для останова опасного вращения двигателя тормоза отпускаются замыкающими контактами в Э/Э/ПЭ СБЗС системах и включаются размыкающими контактами.

Подробное описание данного метода/средства приведено в [5].

А.2 Электроника

Главная цель — управление отказами в твердотельных компонентах.

А.2.1 Тестирование избыточными аппаратными средствами

Цель — обнаружение отказов с использованием избыточных АС, то есть с использованием дополнительных АС, не требующихся для реализации функций обработки.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.3, А.15, А.16 и А.18).

Описание. Избыточные АС могут быть использованы для тестирования на соответствующей частоте запросов к заданным функциям безопасности. Такой подход обычно требуется для реализации положений пунктов А.1.1 или А.2.2.

А.2.2 Динамическая обработка сигналов

Цель — обнаружение статических отказов путем динамической обработки сигналов.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.3).

Описание. Для обнаружения статических отказов в компонентах используют принудительное изменение параметров сторонних статических сигналов (генерируемых внешними и внутренними источниками). Этот метод часто применяют в отношении электромеханических компонентов.

Подробное описание данного метода/средства приведено в [6].

А.2.3 Стандартный тестовый порт доступа и архитектура граничного сканирования

Цель — управление и наблюдение за происходящим на каждом контакте интегральной схемы (ИС).

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.3, А.15 и А.18).

Описание. Тестирование граничного сканирования представляет собой метод построения ИС, который повышает тестируемость ИС, разрешая проблему доступа к внутренним точкам тестируемой схемы. В типичной сканируемой по границам ИС, содержащей внутренние логические схемы, а также входные и выходные буферы, между внутренними логическими схемами и входными/выходными буферами, соединенными с внешними контактами ИС, размещается ступень регистра сдвига. Содержимое каждого регистра сдвига находится в ячейке граничного сканирования. Ячейка граничного сканирования может управлять и наблюдать за происходящим на каждом входном и выходном контакте ИС через стандартный тестовый порт доступа. Тестирование внутренних логических схем ИС проводится путем отключения размещенных на кристалле (ядре) внутренних логических схем от входных сигналов, получаемых от окружающих компонентов, и последующего выполнения внутреннего тестирования. Эти тесты могут быть использованы для обнаружения отказов в ИС.

Подробное описание данного метода/средства приведено в [7] и [8].

А.2.4 Избыточный контроль

Цель — обнаружение отказов посредством создания нескольких функциональных модулей и контроля поведения каждого из них для обнаружения отказов и последующего инициирования перехода в безопасное состояние при обнаружении какого-либо несоответствия в поведении.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.3).

Описание. Функция безопасности выполняется по меньшей мере двумя аппаратными каналами. Выходы этих каналов контролируются, и безопасное состояние инициируется при обнаружении отказа (в случае, если выходные сигналы из всех каналов не идентичны).

Подробное описание данного метода/средства приведено в [9].

А.2.5 Электрические/электронные компоненты с автоматической проверкой

Цель — обнаружение отказов посредством периодической проверки способности выполнения функции безопасности.

Описание. Аппаратные средства тестируются до запуска процесса и затем тестируются повторно через соответствующие интервалы. УО продолжает работу только при условии успешного прохождения каждого теста.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 34332.3—2021 (приложение А, таблица А.3).

А.2.6 Текущий контроль аналоговых сигналов

Цель — повышение достоверности результатов измерений аналоговых сигналов.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.3 и А.13).

Описание. Везде, где возможно, используют аналоговые, а не цифровые АС. В аналоговых приборах отключение или безопасные состояния представляются уровнями аналоговых сигналов обычно с непрерывным контролем выхода за допуски уровней этих сигналов. При этом обеспечиваются непрерывный контроль и высокая степень достоверности передачи сигнала и снижается необходимая частота контроля функции чувствительности датчиков передатчика. Внешние интерфейсы, например, линии с передачей импульсных сигналов, также нуждаются в контроле.

А.2.7 Снижение максимальных значений

Цель — повышение надежности компонентов АС.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (подпункт 8.3.2.14).

Описание. Компоненты АС успешно выполняют свои функции при значениях напряжений, которые определены при проектировании системы и которые ниже максимальных значений, установленных в спецификации

компонентов АС. Снижение этих значений является обычной практикой, гарантирующей, что при всех нормальных условиях эксплуатации компоненты будут успешно функционировать при уровнях напряжений ниже их максимальных значений.

А.3 Модули обработки

Главная цель — обнаружение отказов, которые приводят к неправильным результатам в модулях обработки.

А.3.1 Программное самотестирование (одноканальное): предельное количество комбинаций

Цель — оперативное обнаружение отказов в модулях обработки.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.4).

Описание. АС создают с использованием стандартных методов, не учитывающих специальных требований к безопасности. Обнаружение отказов осуществляется с помощью дополнительных программных функций, которые выполняют самотестирование с использованием по меньшей мере двух дополнительных комбинаций данных (например, 55*hex* и *AAhex*).

А.3.2 Программное самотестирование (одноканальное): «блуждающий бит»

Цель — оперативное обнаружение отказов в устройствах памяти (например, в регистрах) и дешифраторе команд процессора.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.4).

Описание. Обнаружение отказов полностью реализуется с помощью дополнительных программных функций, которые выполняют самотестирование устройств памяти (регистров данных, адресных регистров) с использованием комбинации данных (например, комбинации «блуждающих битов»). Однако охват диагностикой в этом случае составляет только 90 %.

А.3.3 Самотестирование, обеспечиваемое АС (одноканальным)

Цель — оперативное обнаружение отказов в процессоре с использованием специального АС, которое увеличивают скорость и расширяют область обнаружения отказов.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.4).

Описание. Дополнительное специальное АС реализует функцию самотестирования для обнаружения отказов. Например, таким средством может быть аппаратный модуль, который циклически контролирует выход на наличие конкретной битовой комбинации с использованием механизма сторожевой схемы.

А.3.4 Запрограммированная обработка (одноканальная)

Цель — оперативное обнаружение отказов в процессоре.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.4).

Описание. Процессоры могут быть спроектированы со встроенными специальными функциями обнаружения или исправления отказов. Например, такую функцию может выполнять аппаратный модуль, циклически контролирующий выход определенной битовой комбинации в соответствии с принципом действия сторожевой схемы (рисунок А.5).

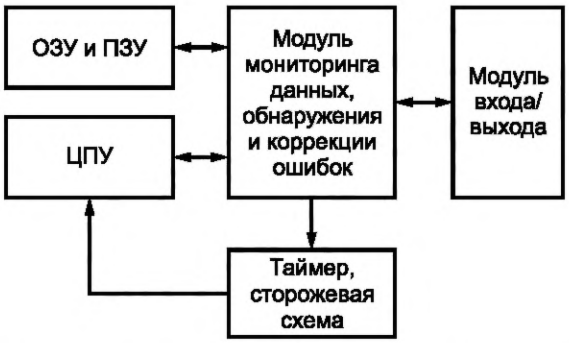


Рисунок А.5 — Структурная схема процессорного модуля с самотестированием и обнаружением отказов

До сих пор такие специальные функции применялись только в относительно простых схемах и не получили широкого распространения. Однако такие функции не следует исключать в будущих разработках.

Подробное описание данного метода/средства приведено в [10]—[12].

А.3.5 Программное обнаружение несовпадений

Цель — оперативное обнаружение отказов (сбоев) в процессоре посредством динамического программного сравнения.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.4).

Описание. Два модуля взаимно обмениваются данными (включая результаты, промежуточные результаты и тестируемые данные). Если при сравнении данных, выдаваемых с использованием программных средств в каждом модуле, обнаруживаются различия, то формируется сообщение об отказе.

А.4 Постоянное запоминающее устройство

Главная цель — выявление изменения информации в ПЗУ.

А.4.1 Защита слов многобитовой избыточностью

Цель — обнаружение всех однобитовых отказов, всех двухбитовых отказов и некоторых отказов во всех битах в 16-битовом слове.

Примечания

1 На данный метод приведена ссылка в ГОСТ 34332.3—2021 (приложение А, таблица А.5).

2 См. также А.5.6 приложения А и В.3.2 приложения В.

Описание. Каждое слово при записи в ПЗУ дополняется несколькими избыточными битами, например, для формирования модифицированного кода Хэмминга с кодовым расстоянием, равным 4 (по меньшей мере). При каждом считывании слова в результате проверки избыточных битов может быть выявлено, произошло ли искажение. При обнаружении искажения вырабатывается сообщение об ошибке. Данный метод может быть также использован для обнаружения ошибок адресации посредством вычисления избыточных битов для объединения слова данных с его адресом.

Подробное описание данного метода/средства приведено в [13]—[16].

А.4.2 Модифицируемая контрольная сумма

Цель — обнаружение всех ошибочных нечетных битов, то есть приблизительно 50 % всех возможных битовых ошибок.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.5).

Описание. Контрольная сумма блока памяти образуется с помощью соответствующего алгоритма обработки всех слов в блоке памяти. Контрольная сумма может храниться как дополнительное слово в ПЗУ, либо может быть добавлена как дополнительное слово в блок памяти для того, чтобы алгоритм контрольной суммы выработал заранее заданное значение. При последней проверке памяти контрольная сумма создается снова с использованием того же алгоритма, и результат сравнивается с запомненным или заданным значением. При обнаружении различий формируется сообщение об ошибке.

Подробное описание данного метода/средства приведено в [13].

А.4.3 Сигнатура из одного слова (8 бит)

Цель — обнаружение всех однобитовых ошибок и всех многобитовых ошибок в слове, а также приблизительно 99,6 % всех возможных битовых ошибок.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.5).

Описание. Содержимое блока памяти сжимается (с использованием АС или ПО) в одно слово памяти с использованием алгоритма контроля с помощью циклического избыточного кода (CRC). В типичном алгоритме CRC все содержимое блока памяти рассматривается как побайтовый или побитовый последовательный поток данных, в котором выполняется непрерывное полиномиальное деление с использованием полиномиального генератора. Остаток от деления сохраняется и представляет собой сжатое содержимое памяти — «сигнатуру» памяти. Сигнатура вычисляется каждый раз при последующем тестировании и сравнивается с уже запомненным значением. При обнаружении различий формируется сообщение об ошибке.

А.4.4 Сигнатура из двух слов (16 бит)

Цель — обнаружение всех однобитовых ошибок и всех многобитовых ошибок в слове составляет примерно 99,998 % всех возможных битовых ошибок.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.5).

Описание. В данном методе предусматривается вычисление сигнатуры с использованием алгоритма контроля с помощью циклического кода коррекции ошибок (CRC), однако длина результирующего значения составляет по меньшей мере два слова. Расширенная сигнатура заносится в память, повторно вычисляется и сравнивается как

одно слово. При обнаружении различий между сохраненной и повторно вычисленной сигнатурами формируется сообщение об ошибке.

А.4.5 Дублирование блока

Цель — обнаружение всех битовых ошибок.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.5).

Описание. В данном методе предусматривается дублирование данных в двух областях памяти (например, сдвоенное ПЗУ с аппаратным или программным сравнением данных). Первая область памяти работает обычным образом. Вторая содержит ту же информацию и доступна параллельно с первой. Их выходы сравниваются, и при обнаружении различий формируется сообщение об ошибке. Для обнаружения некоторых видов битовых ошибок данные должны запоминаться инверсно в одной из двух областей памяти и инвертироваться обратно при чтении.

А.5 Изменяемые диапазоны памяти

Главная цель — обнаружение отказов во время адресации, записи, хранения и считывания.

Примечание — Так называемые исправимые ошибки, перечисленные в ГОСТ 34332.3—2021 (приложение А, таблица А.1), являются отказами, которые должны быть обнаружены в процессе эксплуатации или должны быть проанализированы при выводе доли безопасных отказов. Причинами случайных ошибок являются: альфа-частицы, образовавшиеся в результате процесса распада, нейтроны, внешний источник электромагнитного излучения и внутренние перекрестные помехи. Внешний источник электромагнитного излучения должен соответствовать другим требованиям настоящего стандарта.

Результаты воздействия альфа-частиц и нейтронов могут быть обработаны функционирующими средствами обеспечения полноты безопасности. Но такие средства обеспечения полноты безопасности эффективны для случайных отказов АС и не эффективны для случайных сбоев, например, тесты для ОЗУ, такие как «блуждающая траектория», GALPAT и т. д., не являются эффективными, тогда как методы, использующие контроль четности и коды с исправлением ошибок, возвращающие содержимое ячеек памяти, являются эффективными.

Случайный сбой происходит, когда излучение вызывает такой заряд, который может изменить состояние или переключить с низкого уровня напряжения на высокий ячейку полупроводниковой памяти, регистр, защелку или триггер. Такую случайную ошибку называют «исправимой», потому что сама схема излучением не повреждается. Такие ошибки разделяют на однобитовые нарушения (SBU) или однособытийные нарушения (SEU) и многобитовые нарушения (MBU).

Если схема, в которой произошел сбой, является запоминающим элементом, таким как ячейка памяти или триггер, то ее состояние сохранится до следующей (намеченной) операции записи. Новые данные будут храниться правильно. В комбинаторной схеме это приведет скорее к незначительному сбою, потому что существует постоянный поток энергии из компонента, управляющего этим узлом. Влияние на соединительные провода и линии связи также может быть незначительным. Однако из-за большей емкости воздействие на них альфа-частиц и нейтронов считают незначительным.

Такие случайные сбои могут происходить в переменной памяти любого вида, то есть в динамическом ОЗУ, статическом ОЗУ, регистровой памяти в микропроцессоре, кэш-памяти, конвейерах, регистрах конфигурации устройств, таких как аналого-цифровой преобразователь, DMA, MMU, контроллер прерываний, сложные таймеры. Чувствительность к альфа-частицам и нейтронам зависит от напряжения питания и геометрии. Небольшие конфигурации с напряжением питания 2,5 В и особенно ниже 1,8 В потребуют более серьезной оценки и более эффективных мер защиты.

Интенсивность случайных сбоев для (встроенной) памяти находится в диапазоне от 700 до 1200 «фреймблоков» (Fit) или Мбит. Это эталонное значение для сравнения с данными, полученными для устройств, реализованных на основе кремниевой технологии.

Подробное описание данного метода/средства приведено в [17].

А.5.1 Тесты «шахматная доска» или «марш» для памяти с произвольным доступом

Цель — обнаружение преимущественно статических битовых ошибок.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.6).

Описание. Сформированную в шахматном порядке битовую комбинацию нулей и единиц записывают в ячейки памяти с битовой организацией. Затем эти ячейки анализируют попарно с тем, чтобы убедиться в их одинаковости и правильности. Адрес первой ячейки такой пары является переменным, а адрес второй ячейки этой пары образуется путем битового инвертирования первого адреса. При первом прохождении диапазон адресов памяти проходят в направлении более высоких переменных адресов, а при втором прохождении — в направлении более низких адресов. После этого оба прохождения повторяют с заранее заданным инвертированием. При обнаружении какого-либо различия формируется сообщение об отказе.

При «маршевом» тестировании ОЗУ ячейки памяти с битовой организацией заполняют унифицированным потоком битов. При первом прохождении ячейки анализируют в нисходящей последовательности; проверяют правильность содержимого каждой ячейки и ее содержимое инвертируют. Содержимое, созданное при первом прохождении, рассматривают при втором прохождении в убывающем порядке и также обрабатывают. Первые прохождения повторяют с предварительно инвертированными значениями в третьем и четвертом прохождениях. При обнаружении различий формируется сообщение об отказе.

Подробное описание данного метода/средства приведено в [17], [18].

А.5.2 Тест «блуждающая траектория» для памяти с произвольным доступом

Цель — обнаружение статических и динамических битовых ошибок, а также перекрестных помех между ячейками памяти ОЗУ.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.6).

Описание. Тестируемую область памяти ОЗУ инициализируют (заполняют) унифицированным потоком битов. Затем содержимое первой ячейки инвертируют, и остальную часть памяти анализируют на правильность. После этого содержимое первой ячейки повторно инвертируют для возврата в исходное значение, и всю процедуру повторяют для следующей ячейки. Второе прохождение «модели блуждающего бита» осуществляют при инверсии всех первоначально назначенных значений памяти. При обнаружении различий формируется сообщение об ошибке.

Подробное описание данного метода/средства приведено в [19].

А.5.3 Тесты «GALPAT» и «Прозрачный GALPAT» для памяти с произвольным доступом

Цель — обнаружение статических битовых ошибок и большой части динамических связей.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.6).

Описание. При тестировании ОЗУ «попарной записью—считыванием» (тест «GALPAT») выбранную область памяти сначала инициализируют унифицированно (то есть все логические «0» или все «1»). После этого первую ячейку памяти тестируют и затем инвертируют, и все остальные ячейки анализируют на правильность содержимого. После каждого доступа по чтению к одной из оставшихся ячеек инвертированную ячейку также проверяют. Эту процедуру повторяют для каждой ячейки в выбранной области памяти. Второе прохождение выполняют противоположно первому. Любые различия приводят к формированию сообщения об ошибке.

Тестирование «прозрачной попарной записью—считыванием» (тест «прозрачный GALPAT») представляет собой вариацию описанной выше процедуры: вместо инициализации всех ячеек в выбранной области памяти существующее содержимое остается неизменным, а для сравнения содержимого набора ячеек используют контрольные суммы («сигнатуры»). Выбирают первую тестируемую ячейку области памяти, вычисляют и сохраняют сигнатуру $S1$ всех оставшихся ячеек области. Затем тестируемые ячейки инвертируют и повторно вычисляют сигнатуру $S2$. (После каждого доступа по чтению к одной из оставшихся ячеек инвертируемую ячейку также проверяют.) Сигнатуру $S2$ сравнивают с сигнатурой $S1$, и при любом различии выдается сообщение об ошибке. Тестируемую ячейку повторно инвертируют для повторного установления исходного содержимого, сигнатуру $S3$ всех оставшихся ячеек повторно вычисляют и сравнивают с сигнатурой $S1$. Любые различия приводят к выдаче сообщения об ошибке. Все ячейки памяти в выбранной области тестируют тем же способом.

Подробное описание данного метода/средства приведено в [19].

А.5.4 Тест «Абраам» для памяти с произвольным доступом

Цель — обнаружение всех постоянных отказов и отказов в соединениях между ячейками памяти.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.6).

Описание. При использовании теста «Абраам» диагностический охват выше, чем при тесте «попарная запись—считывание». Число операций, необходимых для выполнения всего тестирования памяти, составляет примерно 30 n, где n — число ячеек памяти. Тестирование может быть «прозрачным» при выполнении запоминания и тестирования в различных временных сегментах в периоде рабочего цикла.

Подробное описание данного метода/средства приведено в [19], [20].

А.5.5 Бит четности

Цель — обнаружение 50 % всех возможных битовых ошибок в тестируемой области памяти.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.6).

Описание. При контроле памяти с произвольным доступом с помощью бита четности каждое слово в памяти расширяют на один бит (бит четности), который дополняет каждое слово до четного или нечетного числа логических единиц. Четность слова данных проверяют при каждом чтении. При обнаружении ложного числа единиц формируется сообщение об ошибке. Выбор четности или нечетности следует осуществлять так, чтобы всякий раз в случае отказа не выдавалось ничего, кроме нулевого (все «0») и единичного (все «1») слова, вырабатывалось уведомление о том, что это слово неправильно закодировано. Контроль четности также может быть использован для обнаружения ошибок адресации, если четность определяется для объединения слова данных с его адресом.

Подробное описание данного метода/средства приведено в [21]—[23].

А.5.6 Контроль памяти с произвольным доступом с помощью модифицированного кода Хэмминга или обнаружение ошибок данных с помощью кодов обнаружения и исправления ошибок

Цель — обнаружение всех нечетных битовых отказов, всех двухбитовых отказов, некоторых трехбитовых отказов и некоторых многобитовых отказов.

Примечание — См. также А.4.1 и В.3.2. Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.6).

Описание. Каждое слово в памяти расширяют несколькими избыточными битами для формирования модифицированного кода Хэмминга с расстоянием Хэмминга, равным по меньшей мере 4. При каждом считывании слова результаты проверки избыточных битов могут показать, произошло ли искажение или нет. При обнаружении различий формируется сообщение об ошибке. Эта процедура может быть также использована для обнаружения ошибок адресации при вычислении избыточных битов для объединения данных с его адресом.

Подробное описание данного метода/средства приведено в [24].

А.5.7 Дублирование со сравнением ОЗУ с АС или ПО и тестирование чтением/записью

Цель — обнаружение всех битовых ошибок.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.6).

Описание. Адресное пространство содержит две части. Первая часть памяти функционирует в нормальном режиме. Вторая часть памяти содержит ту же информацию и доступна параллельно с первой. Выходы этих частей памяти сравнивают. При обнаружении различий формируется сообщение об ошибке. Для обнаружения некоторых видов битовых ошибок данные следует сохранять инверсно в одной из двух частей памяти и обратно инвертироваться при чтении.

А.6 Устройства ввода-вывода и интерфейсы (внешний обмен)

Главная цель — обнаружение отказов на устройствах ввода и вывода (цифровые, аналоговые, последовательные или параллельные) и предотвращение дальнейшей передачи недопустимых выходных данных.

А.6.1 Тестирующая комбинация

Цель — обнаружение константных (статических) отказов и перекрестных помех.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.7, А.13 и А.14).

Описание. При использовании этого метода реализуют независимое от потока данных циклическое тестирование входных и выходных элементов. В нем используют определенную тестирующую комбинацию для сравнения наблюдаемых значений с соответствующими ожидаемыми значениями (рисунок А.6). Информация тестирующей комбинации, считывание и оценка тестирующей комбинации должны быть выбраны независимыми друг от друга. Тестирующая комбинация должна быть выбрана такой, чтобы не оказывать неблагоприятного влияния на операции УО.

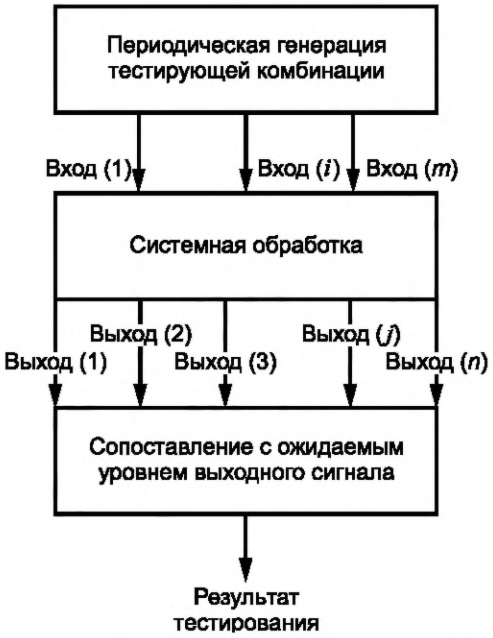


Рисунок А.6 — Применение тестирующей комбинации для обнаружения отказов

А.6.2 Кодовая защита

Цель — обнаружение случайных отказов АС и систематических ошибок в потоке ввода/вывода данных.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.7, А.15, А.16 и А.18).

Описание. Процедура, реализующая кодовую защиту, предназначена для защиты вводимой и выводимой информации от систематических и случайных отказов АС. Применение кодовой защиты обеспечивает зависимое от потока данных обнаружение отказов входных и выходных модулей с использованием избыточности информации и/или временной избыточности. Обычно избыточная информация налагается на входные и/или выходные данные; тем самым обеспечиваются средства для мониторинга правильности операций входных и выходных схем. Возможно применение многих методов. Например, сигнал несущей частоты может быть наложен на выходной сигнал датчика. После этого логический модуль может проверить наличие частоты несущей, либо на выходе канала могут быть добавлены избыточные кодовые биты для контроля действительности прохождения сигнала между логическим модулем и оконечным исполнительным устройством.

Подробное описание данного метода/средства приведено в [23].

А.6.3 Многоканальное параллельное выходное устройство

Цель — обнаружение случайных отказов АС (константных отказов), отказов, обусловленных внешними воздействиями, временных сбоев, отказов адресации, постепенных отказов и самоустраняющихся отказов.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.7).

Описание. Это зависимое от потока данных многоканальное параллельное выходное устройство с независимыми выходами для обнаружения случайных отказов АС. Обнаружение отказов осуществляется с помощью внешних компараторов. При появлении отказа УО непосредственно отключается. Данная мера эффективна только в случае, если поток данных изменяется в интервале диагностического тестирования.

А.6.4 Контролирование выходов

Цель — обнаружение случайных отказов, отказов, обусловленных внешними воздействиями, временных сбоев, отказов адресации, постепенных отказов (для аналоговых сигналов) и самоустраняющихся отказов.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.7).

Описание. Это зависимое от потока данных сравнение выходных данных с независимыми входными данными для оценки соответствия определенному диапазону допуска (время, значение). Обнаруженный отказ не всегда связан с неправильными выходными данными. Данная мера эффективна только в случае, если поток данных изменяется в интервале диагностического тестирования.

А.6.5 Сравнение/голосование данных на входе

Цель — обнаружение случайных отказов, отказов, обусловленных внешними воздействиями, временными сбоями, сбоями адресации, сбоями из-за дрейфа (для аналоговых сигналов) и сбоями из-за переходных процессов.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.7 и А.13).

Описание. Это зависящее от потока данных сравнение независимых входных данных для проверки соответствия определенному диапазону допуска (время, значение). Реализуемая избыточность может быть 1 из 2, 2 из 3 или более высокая. Данная мера эффективна только в случае, если поток данных изменяется в интервале диагностического тестирования.

А.7 Маршруты данных (внутренний обмен)

Главная цель — обнаружение отказов, обусловленных искажениями при передаче информации.

А.7.1 Однобитовая аппаратная избыточность

Цель — обнаружение всех нечетных битовых ошибок, то есть 50 % всех возможных битовых ошибок в потоке данных.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.8).

Описание. Шину расширяют на одну дорожку (бит) и эту дополнительную дорожку (бит) используют для обнаружения отказов путем проверки на четность.

А.7.2 Многобитовая аппаратная избыточность

Цель — обнаружение отказов в процессе передачи по шине и в последовательных каналах связи.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.8).

Описание. Шину расширяют на две или более дорожек (битов) и эти дополнительные дорожки (биты) используют для обнаружения отказов с применением кода Хэмминга.

А.7.3 Полная аппаратная избыточность

Цель — обнаружение отказов в процессе передачи данных путем сравнения сигналов двух шин.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.8).

Описание. Шину дублируют и дополнительную шину используют для обнаружения отказов.

А.7.4 Анализ с использованием тестирующих комбинаций

Цель — обнаружение константных отказов и перекрестных помех.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.8).

Описание. Осуществляют независимое от потока данных циклическое тестирование маршрутов данных. Используют определенную тестирующую комбинацию для сравнения наблюдаемых значений с соответствующими ожидаемым значениями.

Считывание информации тестирующей комбинации и ее оценку следует осуществлять независимо друг от друга. Тестирующая комбинация должна быть выбрана такой, чтобы не оказывать влияния на операции УО.

А.7.5 Избыточность при передаче

Цель — обнаружение самоустраняющихся отказов при передаче по шине.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.8).

Описание. Информацию передают последовательно несколько раз. Повторение осуществляют только для обнаружения самоустраняющихся отказов.

А.7.6 Информационная избыточность

Цель — обнаружение отказов при передаче по шине.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.8).

Описание. Данные передают блоками вместе с вычисленной контрольной суммой для каждого блока. Затем в приемнике осуществляется вычисление контрольной суммы полученных данных и сравнение с принятой контрольной суммой.

А.8 Источник питания

Главная цель — обнаружение отказов или устойчивость к отказам, вызванных дефектом источника питания.

А.8.1 Защита от перенапряжения с защитным отключением

Цель — защита СБ систем от перенапряжения.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.9).

Описание. Перенапряжение обнаруживается достаточно рано с тем, чтобы все выходы могли быть переключены в безопасное состояние путем отключения электропитания или переключения на второй источник питания.

Подробное описание данного метода/средства приведено в [25].

А.8.2 Контроль напряжений вторичного источника питания

Цель — контролирование напряжений вторичных источников питания и инициирование перехода в безопасное состояние, если значение напряжения не находится в заданном диапазоне.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.9).

Описание. Напряжение вторичного источника питания контролируется. Электропитание отключается либо происходит переключение на второй блок питания, если напряжение не находится в заданном диапазоне.

А.8.3 Отключение системы при снижении напряжения питания

Цель — отключение электропитания с сохранением всей критически важной информации.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.9).

Описание. Заранее выявляют перенапряжение или слишком низкое напряжение питания, в пределах которых обеспечивается возможность сохранения внутреннего состояния в энергозависимой памяти и перевод всех выходов в безопасное состояние с помощью процедуры отключения электропитания, либо переключения на второй блок питания.

А.9 Временной и логический контроль последовательности выполнения программ

Главная цель — обнаружение искаженных программных последовательностей. Искаженная программная последовательность появляется, если отдельные элементы программы (например, программные модули, подпрограммы или команды) обрабатываются в неправильной последовательности, или в несоответствующий период времени, или если сбилась тактовая частота процессора.

Примечание — Ссылки на данную группу методов/средств приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.15, А.16 и А.18).

А.9.1 Контрольный датчик времени с отдельной временной базой без временного окна

Цель — контролирование поведения и последовательности выполнения программ.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.10 и А.11).

Описание. Внешние средства определения времени с отдельной базой времени (например, контрольный датчик времени) периодически переключаются для контроля поведения компьютера и последовательности выполнения программ. Важно, чтобы моменты переключения были правильно расположены в программе. Контрольный датчик времени не переключается с определенным фиксированным периодом, однако для него задают максимальный интервал.

А.9.2 Контрольный датчик времени с отдельной временной базой и временным окном

Цель — контроль поведения и последовательности выполнения программ.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.10 и А.11).

Описание. Внешние средства определения времени с отдельной базой времени (например, контрольный датчик времени) периодически переключаются для контроля поведения компьютера и последовательности выполнения программ. Важно, чтобы моменты переключения были правильно расположены в программе. Для контрольного датчика задают нижнюю и верхнюю границы его действия. Если последовательности программ выполняются больше или меньше ожидаемого времени, то осуществляется действие чрезвычайного случая.

А.9.3 Логический контроль последовательности выполнения программ

Цель — контроль правильной последовательности выполнения отдельных частей программы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.10 и А.11).

Описание. Правильная последовательность выполнения отдельных частей программы контролируется с помощью программных средств (процедур учета, ключевых процедур) или с использованием внешних средств контроля. Важно, чтобы точки контроля располагались в программе правильно.

А.9.4 Комбинация временного и логического контроля последовательности выполнения программ

Цель — контроль поведения и правильной последовательности выполнения отдельных частей программы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.10 и А.11).

Описание. Средство определения времени (например, контрольный датчик времени), контролирующее последовательность разделов (модулей) программы, вновь запускается только в случае, если последовательность разделов (модулей) программы выполняется правильно.

А.9.5 Временная проверка при пуске системы

Цель — контролирование поведения и правильности последовательности выполнения отдельных разделов программы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.10 и А.11).

Описание. При пуске Э/Э/ПЭ СБЗС системы проводят временную проверку. Пуск возможен только в случае, если временная проверка прошла успешно. Например, датчик температуры может быть проверен с помощью резистора, нагретого при пуске системы.

А.10 Вентиляция и подогрев

Главная цель — контролирование сбоев в системах вентиляции и подогрева и/или мониторинг систем, если это связано с безопасностью.

Примечание — Ссылки на данную группу методов/средств приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.16 и А.17).

А.10.1 Датчик температуры

Цель — обнаружение перегрева или недогрева АС до того, как Э/Э/ПЭ СБЗС система начнет действовать за пределами заданных требований.

Описание. Датчик температуры контролирует температуру в наиболее критических точках Э/Э/ПЭ СБЗС системы. Прежде чем температура выйдет за пределы заданного диапазона осуществляется чрезвычайное действие.

А.10.2 Контроль вентиляции

Цель — обнаружение неправильной работы вентилятора.

Описание. Работа вентиляторов контролируется. Если вентилятор не работает должным образом, то проводят его техническое обслуживание или осуществляют чрезвычайные действия.

А.10.3 Защитное отключение с использованием плавкого предохранителя

Цель — отключение Э/Э/ПЭ СБЗС системы до того, как она выйдет за пределы заданных температурных режимов.

Описание. Для выключения Э/Э/ПЭ СБЗС системы используют плавкий предохранитель. В системах защиты выключение осуществляют с применением процедуры отключения питания, в которой предусматривается сохранение информации, необходимой при аварийных действиях.

А.10.4 Ступенчатые сообщения от термодатчиков и сигнал условной тревоги

Цель — показать (предупредить), что Э/Э/ПЭ СБЗС система, работает за пределами заданных допусков по температуре.

Описание. Измеряется температура, и при ее выходе из заданного диапазона выдается сигнал условной тревоги (предтревоги).

А.10.5 Соединение устройства принудительного охлаждения воздуха и индикатора состояния

Цель — недопущение перегрева АС с помощью принудительного воздушного охлаждения.

Описание. Измеряется температура. Если температура превышает заданный предел, то включается принудительное воздушное охлаждение. Пользователь информируется об измеренном значении температуры.

А.11 Обмен данными и запоминающее устройство большой емкости

Главная цель — контролирование отказов в процессе обмена данными между внешними источниками и запоминающим устройством большой емкости.

А.11.1 Разделение линий электрического питания и линий передачи информации

Цель — минимизация перекрестных помех, наводимых сильноточными линиями электропитания в информационных линиях.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.16).

Описание. Линии электропитания отделяют от информационных линий. Электрическое поле, которое может индуцировать скачки напряжения на информационных линиях, уменьшается с увеличением расстояния.

А.11.2 Пространственное разделение многофункциональных линий

Цель — минимизация перекрестных помех, индуцируемых сильными токами в многофункциональных линиях.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.16).

Описание. Линии с дублирующими сигналами пространственно отделяют друг от друга. Электрическое поле, которое могут индуцировать броски напряжений в многофункциональных линиях, уменьшается с увеличением расстояния. Такое отделение линий снижает также отказы по общей причине.

А.11.3 Повышение устойчивости к электромагнитным воздействиям

Цель — минимизация электромагнитного влияния на Э/Э/ПЭ СБЗС систему.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.16 и А.18).

Описание. Применение таких мер, как экранирование и фильтрация, для уменьшения чувствительности Э/Э/ПЭ СБЗС систем к электромагнитным полям, которые могут наводиться на линии электропитания или сигнальные линии либо возникать в результате электростатических разрядов.

Подробное описание данного метода/средства приведено в ГОСТ IEC TR 61340-5-2, ГОСТ IEC 61000-6-5, [26].

А.11.4 Передача антивалентных сигналов

Цель — обнаружение одинаковых индуцированных напряжений в групповых линиях передачи сигналов.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.7 и А.16).

Описание. Вся дублируемая информация передается с антивалентными сигналами (например, логическими «1» и «0»). Ошибки по общей причине (например, вызванные электромагнитными излучениями) могут быть обнаружены антивалентным компаратором.

Подробное описание данного метода/средства приведено в ГОСТ IEC/TS 61000-1-2.

А.12 Датчики

Главная цель — управление отказами в датчиках СБС.

А.12.1 Эталонный датчик

Цель — обнаружение отказа датчика.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.13).

Описание. Для контроля работоспособности датчика используют независимый эталонный датчик. Все входные сигналы в подходящие временные интервалы проверяются эталонным датчиком для обнаружения отказов в работе проверяемого датчика.

Подробное описание данного метода/средства приведено в [27], [28].

А.12.2 Положительно управляемый переключатель

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.14).

Цель — размыкание контакта с помощью непосредственного механического соединения между кулачком переключателя и контактом.

Описание. Положительно управляемый переключатель размыкает свои обычно замкнутые контакты непосредственным механическим соединением между кулачком переключателя и контактом. Разомкнутость контактов переключателя обеспечивается всякий раз, когда кулачок переключателя находится в рабочем положении.

Подробное описание данного метода/средства приведено в [29].

А.13 Исполнительные элементы (приводы)

Главная цель — управление отказами в исполнительных элементах Э/Э/ПЭ СБЗС систем.

А.13.1 Мониторинг

Цель — обнаружение отказа исполнительного элемента.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.14).

Описание. Операции исполнительного элемента (например, положительно управляемыми контактами реле; см. контроль контактов реле в А.1.2). Избыточность, вносимая этим контролем, может быть использована для переключения в аварийный режим.

Подробное описание данного метода/средства приведено в [30].

А.13.2 Перекрестный контроль групповых приводов

Цель — обнаружение отказов в приводах посредством сравнения результатов контроля.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение А, таблица А.14).

Описание. Каждый групповой привод контролируют своим аппаратным каналом. При обнаружении различий вырабатывается чрезвычайное действие.

А.14 Меры защиты от окружающей среды

Цель — предотвратить влияние физической окружающей среды (влажности, пыли, коррозийных субстанций), вызывающей отказы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.16 и А.18).

Описание. Корпус оборудования проектируют и изготавливают таким образом, чтобы он выдерживал воздействие ожидаемой окружающей среды.

Подробное описание данного метода/средства приведено в ГОСТ 14254.

Приложение Б
(справочное)

Методы и средства по предотвращению систематических отказов Э/Э/ПЭ СБЗС систем

Настоящее приложение содержит краткое описание методов/средств по предотвращению систематических отказов Э/Э/ПЭ СБЗС систем, на которые даны ссылки в ГОСТ 34332.3 и ГОСТ 34332.4, а также ссылки на источники с подробным описанием данных методов и средств.

Многие методы/средства, представленные в настоящем приложении, относятся и к ПО, но в приложении В они не описаны.

Б.1 Общие методы и средства

Б.1.1 Управление проектами

Цель — избегание отказов посредством принятия организационной модели, правил и мер по разработке и тестированию Э/Э/ПЭ СБЗС систем.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1—Б.6).

Описание. Наиболее значимыми и лучшими мерами являются:

- создание организационной модели, в основном, для обеспечения качества, не противоречащей требованиям ГОСТ ISO 9001;
- установление правил и определение средств для создания и подтверждения соответствия СБ систем в руководствах по взаимосвязанным и отдельным проектам.

Для управления проектами установлены следующие важные базовые принципы:

- при выборе проектной организации определяют:
  - задачи и ответственность подразделений конкретной организации,
  - полномочия подразделений по обеспечению качества,
  - независимость гарантии качества (при выполнении внутренней проверки) от разработки;
- в план последовательных действий (модель действий) включают позиции:
  - определение действий по выполнению проекта, включая внутренние проверки и график их проведения,
  - обновление проекта;
- в стандартную последовательность действий для внутренней проверки включают:
  - планирование, проведение и контроль проверки (теория проверки),
  - использование различных механизмов проверок для составных частей,
  - сохранение результатов повторных проверок;
- в управление конфигурацией включают действия:
  - администрирование и проверка версий,
  - выявление результатов модификаций,
  - проверки согласованности после модификаций;
- вводят количественные оценки для средств обеспечения качества в виде:
  - установления требований,
  - статистики отказов;
- применяют автоматизированные универсальные методы, инструменты и средства обучения персонала.

Подробное описание данного метода/средства приведено в ГОСТ ISO 9001, [31]—[37].

Б.1.2 Документация

Цель — предотвращение отказов и облегчение оценки безопасности Э/Э/ПЭ СБЗС системы с помощью документирования каждого шага процесса проектирования.

Примечания

1 Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1—Б.6).

2 См. также ГОСТ 34332.2 (раздел 5 и приложение А).

Описание. Во время оценки безопасности должны быть продемонстрированы эксплуатационные возможности и безопасность Э/Э/ПЭ СБЗС системы, а также внимание, уделяемое разработке всеми сторонами, вовлеченными в процесс проектирования. Чтобы иметь возможность продемонстрировать степень внимания, уделяемого проектированию, а также для гарантирования проверки доказательств безопасности в любое время, особое внимание уделяют документации. Основными общими подходами к документированию являются введение руководящих принципов создания документов и использование автоматизации, т. е.:

- вводят руководящие принципы:
  - которые определяют структуру документа,
  - в которых предусмотрены таблицы контрольных проверок для формирования содержания документа,
  - которые определяют формат документа;

- применяют автоматизацию управления документированием и создают структурированную библиотеку проекта.

К конкретным методам создания документов относятся:

- разделение в документации описаний:
  - требований,
  - системы (документация пользователя),
  - проектирования (включая внутреннюю проверку);
- группирование проектной документации в соответствии с жизненным циклом Э/Э/ПЭ СБЗС системы;
- определение стандартизованных модулей документации, из которых могут быть скомпилированы документы;
- ясная идентификация составных частей документа;
- формализованное обновление версий;
- выбор ясных и понятных средств описания:
  - формализованной нотации для определений,
  - естественного языка для введений, обоснований и представления намерений,
  - графического представления для описания примеров,
  - семантических определений для графических элементов,
  - терминологических справочников.

Подробное описание данного метода/средства приведено в [38].

Б.1.3 Разделение систем, связанных с безопасностью, и систем, не связанных с безопасностью

Цель — предотвращение влияния систем, не связанных с безопасностью, на части систем, связанных с безопасностью, в непредвиденных ситуациях.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1 и Б.6).

Описание. В спецификации должно быть определено, возможно ли разделение систем, связанных и не связанных с безопасностью. Должны быть установлены четкие спецификации взаимодействия между системами, связанными и не связанными с безопасностью. Четкое их разделение снижает затраты на тестирование систем, связанных с безопасностью.

Подробное описание данного метода/средства приведено в [30].

Б.1.4 Разнообразие аппаратных средств

Цель — обнаружение систематических отказов во время работы УО с использованием разнообразных компонентов с различными частотами и типами отказов.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.15, А.16 и А.18).

Описание. Для разных каналов Э/Э/ПЭ СБЗС системы используют различные типы компонентов. Это снижает вероятность отказов по общей причине (например, перенапряжение, электромагнитные помехи) и повышает вероятность обнаружения таких отказов.

Существование различных средств выполнения требуемой функции, например, применение других физических принципов, предполагает возможность использования других методов решения задачи обнаружения систематических отказов. Существует несколько типов разнообразия. Для получения функционального разнообразия используют различные подходы для достижения одного и того же результата.

Подробное описание данного метода/средства приведено в [30].

Б.2 Спецификация требований к проектированию Э/Э/ПЭ СБЗС системы

Главная цель — создание спецификации требований к Э/Э/ПЭ СБЗС системе, которая по возможности была бы полной, свободной от ошибок, противоречий и простой для проверки.

Б.2.1 Структурирование спецификации

Цель — уменьшение сложности посредством создания иерархической структуры частичных требований. Предотвращение ошибок взаимосвязи между требованиями.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1 и Б.6).

Описание. В данном методе разделяют функциональную спецификацию на частичные требования так, чтобы между ними существовали по возможности простейшие отношения. Этот метод применяют последовательно до тех пор, пока не будут получены небольшие четкие частичные требования. В результате получают иерархическую структуру частичных требований, которая создает основу для спецификации полных требований. В данном методе подчеркиваются взаимосвязи между частичными требованиями, и он особенно эффективен при его использовании для исключения ошибок в этих взаимосвязях.

Б.2.2 Формальные методы

Цель — приложение формальных принципов математического обоснования к созданию спецификации и реализации технических СБ систем для повышения полноты, согласованности или правильности спецификации, или реализации систем.

Примечания

1 Подробные сведения о конкретных формальных методах приведены в Б.2.4.

2 Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1, Б.2 и Б.6).

Описание. Формальные методы предоставляют средства разработки описания системы на конкретном этапе создания ее спецификации или проектирования. Такие формальные описания являются математическими моделями функции и/или структуры системы.

Применение формальных методов позволяет однозначно описать систему (например, любое состояние автомата может быть описано его начальным состоянием, входами и уравнениями перехода автомата из одного состояния в другое), что способствует пониманию системы.

Выбор подходящего формального метода является трудной задачей, требующей полного понимания системы, ее процесса разработки и ряда математических моделей, пригодных для использования (см. примечания).

Примечания

1 Теоремы модели (описывающие свойства) гарантированно представляют описание системы, которое обеспечивает гораздо большее доверие, чем моделирование, заключающееся в наблюдении отдельных действий системы.

2 К недостаткам формальных методов относятся:

- фиксированный уровень абстракции;
- ограничения в получении всей функциональности, которая относится к данному этапу;
- трудность понимания модели инженерами, которые ее реализуют;
- значительные усилия, необходимые для разработки, анализа и поддержки модели на на всех стадиях ЖЦ системы;
- недостаток эффективных инструментов, которые поддерживают создание и анализ модели;
- недостаток персонала, способного разрабатывать и анализировать модель.

3 Интерес представителей, занимающихся формальными методами, был явно направлен на моделирование целевой функции системы, часто с преуменьшением роли проблемы отказоустойчивости системы. Поэтому следует выбирать соответствующие формальные методы, включающие в себя возможность решения проблемы отказоустойчивости системы.

Подробное описание данного метода/средства приведено в [39].

Б.2.3 Полуформальные методы

Цель — четкое и последовательное выражение частей спецификации системы для обнаружения некоторых ошибок, упущений и неправильного поведения.

Примечания

1 Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1, Б.2 и Б.6) и в ГОСТ 34332.4—2021 (приложение А, таблицы А.1, А.2, А.4; приложение Б, таблицы Б.1, Б.7; приложение В, таблицы В.2, В.4 и В.17).

2 В ГОСТ 34332.4—2021 (приложение Б, таблица Б.7) представлен список полуформальных методов, расширенный с помощью других полуформальных методов, относящихся к ПО, которые перечислены ниже:

- логические диаграммы/диаграммы функциональных блоков, см. ГОСТ 34332.4;
- диаграммы последовательности, описанные в ГОСТ 34332.4;
- диаграммы потоков данных, см. Б.2.2;
- конечные автоматы/диаграммы переходов состояний, см. Б.2.3.2;
- временные сети Петри, см. Б.2.3.3;
- модели данных «сущность—связь—атрибут», см. Б.2.4.4;
- диаграммы последовательности сообщений, см. Б.2.14;
- таблицы решений/таблицы истинности, см. Б.6.1.

Подробное описание данного метода/средства приведено в [40], [41].

Б.2.3.1 Общие положения

Цель — убедиться в том, что проект соответствует своей спецификации.

Описание. Полуформальные методы представляют собой методы/средства создания описания системы на стадиях ее создания (например, подготовки спецификации, проектирования или кодирования). В некоторых случаях описание может быть проанализировано на компьютере, или для отображения различных аспектов поведения системы может быть применена анимация. Применение анимации придает дополнительную уверенность в том, что система соответствует реальным требованиям и требованиям, установленным в спецификации.

Описание двух полуформальных методов приведено в Б.2.3.2 и Б.2.3.3.

Б.2.3.2 Конечные автоматы/диаграммы переходов

Цель — моделирование, проверка, задание или реализация структуры управления системы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.5, Б.7; приложение В, таблицы В.15 и В.17).

Описание. Многие системы могут быть описаны в терминах их состояний, входов и действий. Так, система, находящаяся в состоянии *S*1, при получении информации на входе может выполнить действие *A* и перейти в состояние *S*2. Полное представление системы возможно с помощью описания действий системы в каждом ее состоянии для каждого ее входа. Такую модель системы называют машиной с конечными состояниями (или конечными автоматами). Ее часто представляют в виде так называемой диаграммы переходов, в которой отображается, как система переходит из одного состояния в другое, или в виде матрицы, в которой для каждого состояния и входа задаются действия по переходу в новое состояние.

Если система сложна или имеет естественную структуру, то это может быть отражено в многоуровневой структуре конечного автомата. Диаграмма состояний — это тип диаграммы переходов, в которой разрешены вложенные состояния (состояние объекта может разделяться на два или больше число подсостояний, которые могут развиваться параллельно, и, возможно, в некоторый момент времени опять объединиться в одно состояние). Применение данного метода увеличивает дополнительные возможности описания переходов, но увеличивает и сложность, нежелательную для Э/Э/ПЭ СБЗС системы. Диаграммы состояний имеют формальную (математическую) спецификацию. Диаграммы переходов могут быть применены ко всей системе или к ее некоторому объекту или элементу.

Спецификация или проект системы, представленной в виде конечного автомата, могут быть проверены:

- на полноту (система или объект должны иметь действие и новое состояние для каждого входа в каждом состоянии);
- согласованность (возможно только одно состояние для каждой пары состояние/вход);
- достижимость (независимо от того, возможно или нет перейти из одного состояния в другое с помощью некоторой последовательности входов);
- отсутствие бесконечных циклов и тупиковых состояний и т. д.

Эти свойства важны для критических систем. Инструменты для обеспечения таких проверок легко разработать, используя различные модели, основанные на теории конечных автоматов (формальные языки, сети Петри, марковские цепи и т. д.). Существуют также алгоритмы, позволяющие автоматически генерировать тестовые примеры для верификации реализаций конечных автоматов или анимации модели конечного автомата. Диаграммы переходов и диаграммы состояний широко поддерживаются инструментальными средствами, которые позволяют сформировать и проверить диаграммы, а также сгенерировать программный код для реализации описанного конечного автомата.

Они также могут быть применены для вычислений вероятности отказа, см. Б.6 и В.6.

Подробное описание данного метода/средства приведено в [42]—[44].

Б.2.3.3 Моделирование во времени сетями Петри

Цель — моделирование соответствующих аспектов поведения Э/Э/ПЭ СБЗС системы, оценка и, возможно, повышение безопасности и эксплуатационных возможностей системы путем использования анализа и повторного проектирования.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение В, таблица Б.5, приложение В, таблицы В.7, В.15 и В.17).

Описание. Метод сетей Петри является частным случаем метода конечных автоматов. Сети Петри относятся к классу моделей, описываемых теорией графов, и используются для представления информации и управления потоками в системах, в которых наблюдается параллелизм и асинхронное поведение.

Сеть Петри — это сеть позиций и переходов. Позиции могут быть маркированными или немаркированными. Переход считают «активизированным», если все его входы маркированы. В активизированном состоянии позиции разрешается (но не требуется) быть «возбужденной». Если позиция «возбуждена», то вход, поступающий на переход, становится немаркированным, а вместо него каждый выход из перехода оказывается маркированным.

Потенциальные опасности могут быть представлены в виде конкретных состояний (маркировок) в модели сети Петри. Модель может быть расширена с тем, чтобы обеспечить возможности моделирования систем во времени. Несмотря на то, что «классические» сети Петри применяют преимущественно для моделирования потоков управления, существуют некоторые расширения модели сети Петри для моделирования потоков данных.

Подробное описание данного метода/средства приведено в [45], [46].

Б.2.4 Автоматизированные средства разработки спецификации

Б.2.4.1 Общие положения

Цель — использование формальных технических методов для упрощения автоматического обнаружения неоднозначностей и полноты спецификации.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.1, А.2; приложение В, таблицы В.1 и В.2) и ГОСТ 34332.4—2021 (приложение Б, таблицы Б.1 и Б.6).

Описание. Данный метод позволяет создать спецификацию Э/Э/ПЭ СБЗС системы в виде базы данных, которая может автоматически анализироваться для оценки согласованности и полноты. Инструмент спецификации позволяет пользователю использовать анимацию различных аспектов заданной системы. В общем случае данный метод пригоден для поддержания не только стадии создания спецификации требований к Э/Э/ПЭ СБЗС системы, но и стадии проектирования, а также других стадий ЖЦ системы. Инструменты спецификаций могут быть классифицированы в соответствии со следующими пунктами настоящего приложения.

Б.2.4.2 Инструменты, не ориентированные на конкретный метод

Цель — оказание помощи пользователю в составлении правильной спецификации Э/Э/ПЭ СБЗС системы с применением подсказки и формированием связи между соответствующими частями.

Описание. Применение инструмента для составления спецификаций освобождает пользователя от некоторой рутинной работы и поддерживает управление проектами. Инструмент не представляет собой какой-либо конкретный метод разработки спецификаций. Относительная независимость пользователей от метода позволяет быть более свободными при выборе конкретного метода, но незначительно помогает при составлении спецификаций, что усложняет создание системы.

Б.2.4.3 Процедура, ориентированная на модель с иерархическим анализом

Цель — предотвратить неполноту, неоднозначность и противоречивость в спецификации, например, помогая пользователю в создании правильной спецификации, обеспечении согласованности между описаниями процессов и данных на различных уровнях абстрагирования.

Описание. Данный метод дает функциональное представление о необходимой системе (структурный анализ) на различных уровнях абстракции (степени точности). Существует огромный арсенал таких моделей: конечные автоматы — класс таких моделей, широко используемых для описания построения дискретных/цифровых систем. Дифференциальные уравнения позволяют описать непрерывные/аналоговые системы. Структурный анализ проводят на различных уровнях абстракции процессов и данных. Оценка неоднозначности и полноты возможна между иерархическими уровнями, а также между двумя функциональными единицами (модулями) на одном и том же уровне (например, любое состояние модели системы описывается ее начальным состоянием, входами и уравнениями перехода автомата из одного состояния в другое).

Примечание — Вопросами рассмотрения при описании, основанном на моделях, могут быть: уровень абстракции; ограничения для получения всей функциональности, относящейся к данному этапу; трудности понимания модели практиками (от чтения синтаксиса до ее правильной интерпретации); значительность усилий, затрачиваемых на разработку, анализ и поддержку модели на всем ЖЦ системы; доступность эффективных инструментов, поддерживающих создание и анализ модели (разработка таких инструментов, конечно, требует больших усилий) и наличие специалистов, способных разрабатывать и анализировать модели.

Подробное описание данного метода/средства приведено в [47].

Б.2.4.4 Модели данных «сущность — связь — атрибут»

Цель — оказание помощи пользователю в создании правильной спецификации, сосредоточив внимание на объектах внутри системы и отношений между ними.

Описание. Рассматриваемую систему описывают как совокупность объектов и отношений между ними, применяя инструмент, позволяющий определить, какие отношения могут быть интерпретированы системой. В общем случае отношения между объектами позволяют описывать иерархическую структуру объектов, поток данных, отношения между данными и данные, зависимые от конкретных производственных (технологических) процессов. Этот классический подход расширяют посредством применения управления процессами. Возможности метода и поддержка пользователя зависят от разнообразия проиллюстрированных отношений. Однако множество возможностей представления системы усложняет применение этого метода.

Подробное описание данного метода/средства приведено в [48].

Б.2.4.5 Стимул и отклик

Цель — оказание помощи пользователю в создании правильной спецификации Э/Э/ПЭ СБЗС системы посредством идентификации взаимоотношений «стимул — отклик».

Описание. Взаимоотношения между объектами системы определяют в нотации «стимулы» и «отклики». Используют простой и легко расширяемый язык, который содержит элементы языка, представляющие объекты, взаимоотношения, характеристики и структуры.

Б.2.5 Таблица контрольных проверок

Цель — рассмотрение и управление критическими оценками всех важных аспектов Э/Э/ПЭ СБЗС системы на стадии ЖЦ, обеспечивая исчерпывающий охват аспектов без установления точных требований.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1, Б.2 и Б.6) и в ГОСТ 34332.4—2021 (приложение А, таблица А.10; приложение Б, таблица Б.8; приложение В, таблицы В.10 и В.18).

Описание. Специалист, заполняющий таблицу контрольных проверок, должен дать ответ на ряд вопросов. Многие вопросы носят общий характер, и оценщик должен интерпретировать их как наиболее подходящие к конкретной оцениваемой Э/Э/ПЭ СБЗС системе. Таблицы контрольных проверок допускается использовать на всех стадиях полного ЖЦ Э/Э/ПЭ СБЗС системы и ЖЦ СБЗС ПО. Такие таблицы, в частности, полезны в качестве инструмента для оценки функциональной безопасности.

Для сокращения широкого разнообразия проходящих подтверждение соответствия систем большинство таблиц контрольных проверок содержат вопросы, которые применимы ко многим типам систем. Поэтому в используемой таблице контрольных проверок может оказаться множество вопросов, которые не уместны для конкретной системы и должны быть игнорированы. Кроме того, может также возникнуть необходимость дополнить стандартную таблицу контрольных проверок вопросами, специально ориентированными на конкретную систему.

Использование таблицы контрольных проверок в большей степени зависит от экспертной оценки и суждения специалиста, который выбирает и применяет таблицу контрольных проверок. Принятые им решения относительно выбранных(ой) таблиц(ы) контрольных проверок и любые дополнительные или игнорируемые вопросы должны быть полностью обоснованы и документально оформлены. Необходимо стремиться к тому, чтобы при пересмотре таблиц контрольных проверок гарантировалось получение одних и тех же результатов при использовании одних и тех же критериев.

Описание системы в заполненной таблице контрольных проверок должно быть максимально кратким. При необходимости исчерпывающего обоснования оно должно быть дано в виде ссылок на дополнительные документы. Для документирования результатов каждого вопроса следует использовать ответ «успешно», «не успешно» или «недостаточно убедительно», либо аналогичный набор ответов. Такая лаконичность значительно упрощает процедуру оформления общего заключения результатов оценки в виде таблицы контрольных проверок.

Подробное описание данного метода/средства приведено в [30], [49]—[54].

Б.2.6 Экспертиза спецификации

Цель — исключение некомплектности и противоречивости спецификации.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.1 и Б.6).

Описание. Экспертиза — это общий метод анализа спецификации Э/Э/ПЭ СБЗС системы для ее разносторонней оценки, осуществляемая независимой группой экспертов. Эксперты такой группы задают вопросы разработчику, который должен дать им удовлетворительные ответы. Анализ должен (по возможности) проводиться группой экспертов, которая не принимала участия в создании спецификации. Требуемая степень независимости оценки определяется УПБ, задаваемыми для системы. Группа независимых экспертов должна быть способна реконструировать эксплуатационную функцию системы без ссылок на любые последующие спецификации. Специалисты группы независимых экспертов должны также убедиться в том, что охвачены все уместные аспекты безопасности и технические аспекты эксплуатационных и организационных мер. Общий метод экспертизы спецификации доказал на практике свою высокую эффективность.

Подробное описание данного метода/средства приведено в [51], [55].

Б.3 Проектирование и разработка Э/Э/ПЭ СБЗС системы

Главная цель — создание надежного проекта Э/Э/ПЭ СБЗС системы в соответствии со спецификацией.

Б.3.1 Соблюдение руководящих материалов и стандартов

Цель — учет требований стандартов секторов применения (не рассматриваемых в настоящем стандарте).

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение Б, таблица Б.2).

Описание. Во время проектирования Э/Э/ПЭ СБЗС системы должны быть составлены руководящие материалы, применение которых должно приводить к созданию систем, практически не имеющих сбоев и упрощающих последующее подтверждение соответствия. Такие руководящие материалы могут быть универсальными, специфичными для проекта или только для отдельного этапа проекта.

Подробное описание данного метода/средства приведено в [30].

Б.3.2 Структурное проектирование

Цель — снижение сложности проектирования Э/Э/ПЭ СБЗС системы посредством создания иерархической структуры частичных требований; исключение ошибок взаимосвязей между требованиями; упрощение верификации.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.2 и Б.6).

Описание. При проектировании АС следует использовать конкретные критерии или методы. Например, может потребоваться:

- проектирование иерархически структурированных схем;
- использование изготовленных и проверенных частей схем.

При проектировании ПО использование структурных схем также позволяет создать однозначную структуру программных модулей. В данной структуре отображаются взаимосвязь модулей друг с другом, конкретные данные, которые передаются между модулями, и конкретное управление, существующее между модулями.

Подробное описание данного метода/средства приведено в ГОСТ IEC 61082-1, [52]—[54], [56].

Б.3.3 Использование достоверно испытанных компонентов

Цель — снижение риска многих оригинальных и необнаруживаемых отказов Э/Э/ПЭ СБЗС системы посредством использования компонентов с конкретными характеристиками.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.2 и Б.6).

Описание. Выбор достоверно испытанных компонентов для целей безопасности выполняется производителем в соответствии с надежностью компонентов (например, использование проверенных тестированием физических модулей для удовлетворения высоких требований безопасности или хранение относящихся к безопасности программ только в безопасной памяти). Обеспечение безопасности памяти может относиться к недопущению несанкционированного доступа к памяти, влиянию несанкционированной среды (электромагнитная совместимость, радиация и т. п.), а также к отклику компонентов в случае возникновения отказов.

Подробное описание данного метода/средства приведено в [57].

Б.3.4 Модульное проектирование

Цель — уменьшить сложность и избежать сбоев Э/Э/ПЭ СБЗС системы, связанных с взаимодействием между подсистемами.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.2 и Б.6).

Описание. Каждая подсистема на всех уровнях проектирования четко определена и ограничена по размеру (только небольшим набором функций). Интерфейсы между подсистемами выполняют максимально простыми и пересечения (разделяемые данные, обмен информацией) минимизируют. Сложность отдельных подсистем также ограничивают.

Подробное описание данного метода/средства приведено в [56], [58].

Б.3.5 Средства автоматизированного проектирования

Цели — более систематическое выполнение процессов проектирования Э/Э/ПЭ СБЗС системы; включение в проект подходящих автоматически сконструированных элементов, уже созданных и проверенных.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.2 и Б.6) и в ГОСТ 34332.4—2021 (приложение А, таблица А.4).

Описание. В процессе проектирования АС и ПО следует использовать средства автоматизированного проектирования САПР, если они доступны и их использование обосновано сложностью системы. Корректность применения таких средств должна быть продемонстрирована предоставлением результатов конкретных испытаний, описания обширной предыстории успешного использования либо результатов независимой верификации проектируемой СБС.

Для их интеграции необходимо выбирать инструменты поддержки. Инструменты поддержки выбирают в соответствии с их уровнем интегрируемости. Инструменты считают интегрируемыми, если они совместно работают так, что выходные данные одного инструмента по содержанию и формату подходят для автоматического ввода в следующий инструмент, что приводит к минимизации возможности внесения ошибки человеком в процессе его работы с промежуточными результатами.

Подробное описание данного метода/средства приведено в [59], [60].

Б.3.6 Моделирование

Цель — проведение систематических и полных проверок функционирования Э/Э/ПЭ СБЗС системы для корректного задания функциональных и размерных характеристик их компонентов.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.2, Б.5 и Б.6).

Описание. Функцию схемы, реализующую Э/Э/ПЭ СБЗС систему, имитируют на компьютере с помощью запрограммированной модели ее поведения. Поведение каждого компонента схемы моделируют отдельно, и отклик схемы, в которую он входит, анализируют при задании предельных значений параметров для каждого компонента.

Б.3.7 Проверка (обзор и анализ)

Цель — выявление рассогласования между спецификацией и реализацией Э/Э/ПЭ СБЗС системы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.2 и Б.6).

Описание. Проверяют заданные функции Э/Э/ПЭ СБЗС системы. Оценивают соответствие требованиям, приведенным в спецификации. Любые вопросы, связанные с созданием и использованием продукции, документи-

руют, чтобы они могли быть разрешены. В отличие от сквозного контроля во время процедуры проверки проектировщик системы пассивен, а эксперт активен.

Подробное описание данного метода/средства приведено в [55], [58], [59].

Б.3.8 Сквозной контроль

Цель — выявление рассогласования между спецификацией и реализацией Э/Э/ПЭ СБЗС системы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение Б, таблица Б.6).

Описание. Проверяют заданные функции Э/Э/ПЭ СБС. Оценивают соответствие системы требованиям, приведенным в спецификации. Все вызывающие сомнение ситуации при реализации и использовании изделий документируют в целях их последующего разрешения. В отличие от процедуры проверки (Б.3.7) во время сквозного контроля проектировщик должен быть активен, а эксперт — пассивен.

Подробное описание данного метода/средства приведено в [59].

Б.4 Процедуры эксплуатации и технического обслуживания Э/Э/ПЭ СБЗС системы

Главная цель — разработка процедур, которые исключают ошибки во время эксплуатации и обслуживания Э/Э/ПЭ системы.

Б.4.1 Инструкции по эксплуатации и техническому обслуживанию

Цель — исключение ошибок во время эксплуатации и технического обслуживания Э/Э/ПЭ СБЗС системы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение Б, таблица Б.4).

Описание. Инструкции пользователя содержат важную информацию о способах использования и технического обслуживания систем. В особых случаях эти инструкции могут содержать также примеры общих способов установки СБ систем. Все инструкции должны быть выполнены легко воспринимаемыми. Для описания сложных процедур и зависимостей следует использовать рисунки и схемы.

Подробное описание данного метода/средства приведено в [30].

Б.4.2 Удобство для пользователя

Цель — снижение сложности эксплуатации Э/Э/ПЭ СБЗС систем.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение Б, таблица Б.4).

Описание. Правильность эксплуатации Э/Э/ПЭ СБЗС систем в определенной степени зависит от оператора. Рассматривая конкретный проект системы и рабочего места, проектировщик Э/Э/ПЭ СБЗС системы должен предусмотреть:

- необходимость минимального вмешательства человека;
- наиболее простой способ необходимого вмешательства;
- минимизацию причинения вреда из-за ошибок оператора;
- эргономические требования при проектировании средств вмешательства и индикации;
- простые, имеющие четкую маркировку и удобные для использования средства оператора;
- обеспечение неперенапряженности оператора даже в экстремальной ситуации;
- адаптацию обучения процедурам и средствам вмешательства оператора в процесс к уровням его знаний и мотивации.

Б.4.3 Удобство технического обслуживания

Цель – упрощение процедуры технического обслуживания Э/Э/ПЭ СБЗС системы и проектирование необходимых средств для эффективной диагностики и ремонта.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение Б, таблица Б.4).

Описание. Техническое обслуживание и ремонт часто проводят в сложных условиях ограничения предельных сроков их выполнения. Поэтому проектировщик Э/Э/ПЭ СБЗС системы должен предусмотреть:

- чтобы средства, относящиеся к техническому обслуживанию, требовались как можно реже или вообще не требовались;
- чтобы использовались достаточно чувствительные и легко управляемые диагностирующие средства для неизбежных ремонтов, включающие в себя все необходимые интерфейсы;
- чтобы было достаточно времени (если отдельные средства диагностики необходимо разработать или приобрести).

Б.4.4 Сокращение работ на стадии эксплуатации

Цель — снизить эксплуатационные возможности для обычного пользователя Э/Э/ПЭ СБЗС системы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.4 и Б.6).

Описание. При этом подходе снижают эксплуатационные возможности системы посредством:
- ограничения операций в рабочих режимах, например, коммутаторами ключей;
- ограничения числа используемых в работе элементов;
- ограничения числа возможных в общем случае рабочих режимов.

Подробное описание данного метода/средства приведено в [30].

Б.4.5 Эксплуатация только квалифицированным оператором

Цель — исключение отказов, обусловленных ошибками оператора.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.4 и Б.6).

Описание. Оператор Э/Э/ПЭ СБЗС систем должен быть обучен до степени, соответствующей уровню сложности и УПБ этой системы. В обучение входит изучение основ процесса эксплуатации систем и взаимосвязей между Э/Э/ПЭ СБЗС системами и УО.

Подробное описание данного метода/средства приведено в [30].

Б.4.6 Защита от ошибок оператора

Цель — защита Э/Э/ПЭ СБЗС системы от всех видов ошибок оператора.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.4 и Б.6).

Описание. Ложные входные сообщения (значение, время и т. д.) обнаруживают проверками их достоверности или с помощью контролера УО. Для того чтобы объединить эти средства в проекте, необходимо на самом раннем этапе определить, какие из входных сообщений возможны и какие допустимы.

Б.4.7 Защита от модификаций

Цель — защита Э/Э/ПЭ СБЗС системы от модификаций АС техническими способами.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.17 и А.18).

Описание. Модификации или манипуляции обнаруживаются автоматически, например, проверками достоверности сигналов датчиков, обнаружением техническим процессом и автоматическими тестами пуска. Если обнаружена модификация, выполняется экстренное действие.

Б.4.8 Подтверждение ввода

Цель — обнаружение ошибок самим оператором во время работы Э/Э/ПЭ СБЗС системы, до активизации УО.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблицы А.17 и А.18).

Описание. Информация, вводимая из Э/Э/ПЭ СБЗС системы в УО, представляется оператору до передачи в УО с тем, чтобы оператор имел возможность обнаружить и исправить ошибки. Систему проектируют так, чтобы она могла реагировать на неправильные, самопроизвольные действия оператора и учитывать нижние/верхние пределы скорости и направление реакции оператора. Это позволяет исключить, например, более быстрое, чем предполагается, нажатие клавиш оператором, и настроить систему на восприятие двойного нажатия клавиши как одинарное или двойное за счет того, что система (изображение на экране) слишком медленно реагирует на разовое нажатие клавиши. Последовательное нажатие одной и той же клавиши при вводе критических данных должно восприниматься системой как одноразовое; нажатие клавиш «Ввод» (Enter) или «Да» (Yes) неограниченное число раз не должно приводить к нарушению безопасности системы.

Должны быть предусмотрены процедуры формирования временных пауз с возможностью выбора разных ответов (да/нет и т. п.) с тем, чтобы обеспечить резерв времени для размышления оператору, а системе — режим ожидания.

Любая перезагрузка ПЭ СБЗС системы делает эту систему уязвимой, если АС и ПО не спроектированы с учетом данной ситуации.

Б.5 Интеграция Э/Э/ПЭ СБЗС системы

Главная цель — исключение отказов Э/Э/ПЭ СБЗС системы на стадии интеграции и обнаружение любых отказов во время этой и предыдущей стадий.

Б.5.1 Функциональное тестирование

Цель — обнаружение отказов на стадиях создания спецификации и проектирования Э/Э/ПЭ СБЗС системы; исключение отказов во время реализации и интеграции ее АС и ПО.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.3 и Б.5) и в ГОСТ 34332.4—2021 (приложение А, таблицы А.5—А.7 и приложение Б, таблицы Б.5—Б.7).

Описание. В процессе функционального тестирования определяют, достигнуты ли заданные характеристики системы. В систему поступают входные данные, которые адекватно характеризуют обычное выполнение опера-

ций. Наблюдаемые выходные результаты сравнивают с заданными в спецификации. Отклонения от спецификации и указания на неполноту спецификации документально оформляют.

Функциональное тестирование электронных компонентов, предназначенных для многоканальной архитектуры, обычно включает в себя промышленные компоненты, каждый из которых поставщик уже протестировал и предварительно подтвердил соответствие. Помимо этого, рекомендуется, чтобы покупные промышленные компоненты были протестированы в сочетании с другими компонентами поставщика из той же партии, чтобы выявить неисправности группового типа, которые в противном случае остались бы не выявленными.

О достаточных рабочих возможностях системы см. также руководящие материалы (см. В.5.20 приложения В).

Подробное описание данного метода/средства приведено в [58], [61].

Б.5.2 Тестирование методом «черного ящика»

Цель — проверка динамического поведения Э/Э/ПЭ СБЗС системы в реальных условиях функционирования; выявление несоответствия функциональной спецификации и оценка ее полезности и устойчивости.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.3, Б.5 и Б.6) и в ГОСТ 34332.4—2021 (приложение А, таблицы А.5–А.7 и приложение Б, таблицы В.5—В.7).

Описание. Функции системы или программы выполняются в заданном окружении с заданными данными тестирования, которые систематически формируются из спецификации в соответствии с установленными критериями. Это позволяет сравнить поведение системы с ее спецификацией. При проведении тестирования не используют сведения о внутренней структуре системы. Основная цель состоит в том, чтобы определить, правильно ли выполняют функциональный модуль функции, требуемые спецификацией. Примером критерия тестирования данных методом «черного ящика» служит метод формирования эквивалентных классов. Массив входных данных подразделяют на конкретные диапазоны входных значений (эквивалентные классы) на основе спецификации. После этого формируют тестовые примеры, применяя:

- данные из допустимых диапазонов;
- данные из недопустимых диапазонов;
- данные предельных значений диапазонов;
- экстремальные значения;
- комбинации из перечисленных выше классов.

Могут оказаться эффективными также другие критерии выбора тестовых примеров в различных режимах тестирования (модуля, интеграции и системы). Например, критерий «экстремальные эксплуатационные условия» используют при тестировании системы в процессе подтверждения соответствия.

Подробное описание данного метода/средства приведено в [58], [62], [63].

Б.5.3 Статистическое тестирование

Цель — проверка динамического поведения Э/Э/ПЭ СБЗС системы и оценка ее полезности и устойчивости.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.3, Б.5 и Б.6).

Описание. При этом подходе тестируют систему или программу с входными данными, выбранными в соответствии с предполагаемым статистическим распределением реальных эксплуатационных входных данных — эксплуатационным профилем.

Подробное описание данного метода/средства приведено в [64].

Б.5.4 Полевые испытания

Цель — использование результатов полевых испытаний из различных областей применения в качестве одного из средств исключения сбоев во время интеграции Э/Э/ПЭ СБЗС системы и/или в процессе подтверждения ее соответствия.

Примечания

1 Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.3, Б.5 и Б.6).

2 См. также приложение В, В.2.10 — аналогичные средства, а в приложении Г — статистический подход — то и другое в контексте программного обеспечения.

Описание. Использование компонентов или подсистем, которые на практике показали отсутствие ошибок или незначительное их число, практически не изменяемое в течение продолжительного периода времени в многочисленных различных применениях. В частности, для сложных компонентов с множеством функций (например, операционной системы, интегральных схем) проектировщик должен обратить внимание на функции, которые были фактически протестированы в ходе полевых испытаний. Например, должны быть рассмотрены подпрограммы самотестирования для обнаружения сбоев, поскольку при отсутствии сбоев АС в период эксплуатации нет уверенности в том, что они протестированы, поскольку подпрограммы никогда не выполняли функций обнаружения своих собственных сбоев.

При проведении полевых испытаний должны быть соблюдены следующие требования:

- неизменность спецификации;

- наличие не менее 10 истем в различных применениях;
- продолжительность работы не менее 10^5 час и техническое обслуживание не реже одного раза в год.

Примечание — В стандартах различных областей применения могут быть определены другие значения этих параметров.

Полевые испытания документируются поставщиком, проектировщиком и эксплуатирующей организацией. В документацию включают, по меньшей мере:
- точное обозначение системы и ее компонентов, включая управление версиями АС;
- сведения о пользователях и времени применения;
- отработанное время в часах;
- описание процедур выбора системы и прикладные программы, использованные при испытаниях;
- описание процедур обнаружения и регистрации сбоев, а также процедуры устранения их последствий и причин возникновения.

Подробное описание данного метода/средства приведено в [30], [65].

Б.6 Подтверждение соответствия Э/Э/ПЭ СБЗС системы

Главная цель — подтвердить, что Э/Э/ПЭ СБЗС система соответствует спецификации требований к системе и спецификации требований к ее проектированию.

Б.6.1 Функциональные испытания в условиях окружающей среды

Цель — оценка защищенности Э/Э/ПЭ СБЗС системы от типовых воздействий окружающей среды.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.3—2021 (приложение Б, таблица Б.5).

Описание. Систему помещают в различные условия окружающей среды (например, в соответствии со стандартами на воздействие внешних факторов или стандартов на электромагнитную совместимость) и оценивают способности системы выполнять функции безопасности (на соответствие требованиям указанных стандартов).

Подробное описание данного метода/средства приведено в [26], [66].

Б.6.2 Испытания на устойчивость к перекрестным помехам

Цель — проверка способности Э/Э/ПЭ СБЗС систем выдерживать пиковые воздействия.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6).

Описание. В систему загружают типичную прикладную программу и все периферийные линии (цифровые, аналоговые и последовательные интерфейсы, шины, источники питания и т.д.) подвергают воздействию стандартных шумовых сигналов. Для того чтобы получить их количественную оценку, целесообразно внимательно подходить к предельным значениям пиковых воздействий. Класс помех считается выбранным неверно, если функция системы не выполняется.

Подробное описание данного метода/средства приведено в ГОСТ IEC 60255-5.

Б.6.3 Статический анализ

Цель — исключение систематических дефектов, которые могут приводить к отказам в испытываемой Э/Э/ПЭ СБЗС системе вначале либо после продолжительной эксплуатации.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6) и в ГОСТ 34332.4 (приложение А, таблица А.9; приложение Б, таблица Б.8; приложение В, таблицы В.9 и В.18).

Описание. Этот систематический и, возможно, автоматизированный метод позволяет исследовать конкретные статические характеристики прототипов системы для обеспечения полноты, согласованности, отсутствия неоднозначностей в рассматриваемых требованиях (например, в руководящих материалах по принципам построения, спецификациях и техническом паспорте системы). Статический анализ должен быть воспроизводимым и применимым к прототипу, который доведен до четко определенной завершающей стадии. Ниже приведены некоторые примеры статического АС и ПО:
- анализ согласованности потока данных (например, при тестировании, если данные об объекте интерпретируются как имеющие одно значение);
- анализ управления потоком (например, определение маршрутов, кода недоступности);
- анализ интерфейсов (например, исследование передачи переменных между различными программными модулями);
- анализ потока данных для обнаружения вызывающих сомнения последовательностей для переменных: создание — использование для обращения — удаление;
- проверка строгого соблюдения конкретных руководящих материалов (например, по вопросам: длина пути утечки тока и зазоры, расстояние между группами модулей, физическое расположение модулей, механически чувствительные физические модули, индивидуальное использование физических модулей при их внедрении).

Подробное описание данного метода/средства приведено в [67].

Б.6.4 Динамический анализ и тестирование

Цель — обнаружение ошибок в спецификации путем исследования динамического поведения прототипа Э/Э/ПЭ СБЗС системы на завершающих стадиях.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6) и в ГОСТ 34332.4—2021 (приложение А, таблицы А.5, А.9; приложение Б, таблица Б.2; приложение В, таблицы В.5, В.9 и В.12).

Описание. Динамический анализ систем проводят при подаче на вход прототипа Э/Э/ПЭ СБЗС системы, входных данных, которые типичны для заданного эксплуатационного окружения. Анализ считают удовлетворительным, если наблюдаемое поведение СБ системы соответствует требуемому поведению. Любой отказ системы должен быть устранен, после чего должны быть проанализированы новые варианты эксплуатации системы.

Подробное описание данного метода/средства приведено в [68].

Б.6.5 Анализ отказов

Б.6.5.1 Анализ видов и последствий отказов

Цель — проведение анализа проекта Э/Э/ПЭ СБЗС системы с систематическим исследованием всех возможных причин отказов компонентов системы и определением влияния этих отказов на поведение и безопасность системы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6).

Описание. Анализ обычно проводят экспертным методом. Каждый компонент системы анализируют по очереди с тем, чтобы выявить набор режимов отказов для компонента, их причины и последствия (на локальном уровне и на уровне всей системы), процедуры обнаружения и рекомендации. При выдаче рекомендаций их документально оформляют в виде корректирующих действий.

Подробное описание данного метода/средства приведено в [30], [69].

Б.6.5.2 Причинно-следственные диаграммы

Цель — моделирование Э/Э/ПЭ СБЗС системы с помощью причинно-следственных диаграмм, которые позволяет представить проект системы в виде последовательности комбинаций базовых событий.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.3, Б.4 и приложение В, таблицы В.13 и В.14).

Описание. Данный метод может рассматриваться как комбинация процедур анализа с помощью дерева отказов и дерева событий. Начиная с критического (начального) события, граф последствий просматривают в прямом направлении, используя логические элементы «ДА»/«НЕТ», описывающие успех и неудачу некоторых операций. Это позволяет сформировать последовательность событий, ведущую или к аварии, или к корректной ситуации. Затем для каждого отказа строят графы причин (то есть деревья отказов). Прохождение в обратном направлении эквивалентно дереву отказов, где критическое событие представлено в виде события, описанного на верхнем уровне. Прохождение в прямом направлении позволяет определять возможные последствия, возникающие из события. В узле графа могут быть символы, описывающие условия распространения причин по различным ветвям от этого узла. Могут быть также учтены временные задержки. Эти условия распространения причин также могут быть описаны с помощью деревьев отказов. Для того чтобы диаграмма выглядела более компактной, пути распространения причин могут быть объединены с логическими символами. Должен быть определен набор стандартных символов для использования в причинно-следственных диаграммах. Такие диаграммы могут быть использованы для генерации деревьев отказов и для вычисления вероятности появления определенных критических последовательностей. Они также могут быть использованы для генерации деревьев событий.

Подробное описание данного метода/средства приведено в [70].

Б.6.5.3 Анализ дерева событий

Цель — моделирование Э/Э/ПЭ СБЗС системы с помощью диаграмм последовательности событий, которые могут произойти в системе после появления инициализирующего события и указать на возможные опасные последствия. Дерево событий трудно создать с нуля, поэтому полезно использовать схему последовательности событий.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблица Б.4 и приложение В, таблица В.14).

Описание. В верхней части диаграммы записывают последовательность условий, относящихся к формированию последовательности событий, следующих за инициализирующим событием. Начиная с инициализирующего события, являющегося целью анализа, проводят прямую линию к первому условию последовательности. Наличие ветвей «ДА» и «НЕТ» диаграммы указывает на зависимость будущего события от условий. Каждую из двух ветвей продолжают до следующего условия. Однако не все условия выполняются на этих ветвях. Какая-то из них продолжится до окончания последовательности условий, но каждая ветвь дерева, построенная таким способом, представляет возможную последовательность событий. Если условия в последовательностях независимы, дерево

событий может быть использовано для вычисления вероятностей различных последовательностей, основываясь на значениях вероятностей условий и их числе в последовательности. Поскольку условия редко бывают полностью независимыми, такие вычисления необходимо тщательно анализировать, и анализ должен выполняться квалифицированными аналитиками.

Подробное описание данного метода/средства приведено в [70].

Б.6.5.4 Анализ видов, последствий и критичности отказов

Цель — ранжирование критичности компонентов, которые могут вызвать нарушения, повреждения или ухудшение работы системы при одиночных ошибках в целях определить, для определения, каким компонентам может потребоваться особое внимание и какие средства управления необходимы в процессе проектирования или эксплуатации.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблица А.10; приложение Б, таблица Б.4; приложение В, таблицы В.10 и В.14).

Описание. Этот метод сопоставим с методом FMEA, представленном в [73], но отличается наличием одного или нескольких столбцов для описания критичности, которая может быть ранжирована многими методами. Наиболее сложный метод описан Обществом автомобильных инженеров (Society for Automotive Engineers — SAE) в ARP 926. В этом методе значение критичности для любого компонента определяется числом отказов конкретного вида, предполагаемым в процессе выполнения каждого миллиона операций, реализуемых в критическом режиме. Критичность является функцией девяти параметров, большинство из которых должны быть измерены. Очень простой метод определения критичности состоит в умножении вероятности отказа компонента на величину ущерба, который может быть при этом причинен; этот метод аналогичен простой оценке показателя риска.

Подробное описание данного метода/средства приведено в [69], [71], [72].

Б.6.5.5 Анализ дерева отказов

Цель — оказание помощи в анализе событий или комбинации событий, которые вызывают угрозы или опасные последствия, и в выполнении вычисления вероятности главного события.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблица Б.4 и приложение В, таблица В.14).

Описание. Начиная с главного события, которое может непосредственно вызвать угрозу или опасные последствия («событие вершины дерева»), выполняют анализ для идентификации причины этого события. Комбинации причины описывают логическими операторами («И», «ИЛИ» и т. д.). Затем анализируют промежуточные причины тем же способом и т. д., возвращаясь к базовым событиям, где анализ прекращается.

Данный метод является графическим, и для изображения дерева отказов используют набор стандартизованных символов. В результате анализа дерево отказов представляет собой логическую функцию, объединяющую базовые события (обычно отказы компонентов) с главным событием (полный отказ системы). Рассматриваемый метод предназначен в основном для анализа АС, но допускается также применять его к анализу ошибок ПО. Этот метод может быть использован для качественного анализа отказов (идентификация сценариев отказа: минимальные сечения или простые импликанты), полуколичественного анализа (оценивая сценарии их вероятностями) и количественного анализа для вычислений вероятности главного события (см. В.6).

Подробное описание данного метода/средства приведено в [70], [73].

Б.6.5.6 Модели Маркова

Цель — моделирование поведения Э/Э/ПЭ СБЗС системы с использованием графа состояний—переходов и оценка общесистемных параметров [ненадежность, неготовность, среднее время наработки на отказ (MTTF, MUT, MDT) и др.] системы.

Описание. Модель — это конечный автомат (см. Б.2.3.2), представленный направленным графом. Узлы (кружки) представляют состояния, а ребра (стрелки) между узлами представляют переходы (отказы, ремонты и т. д.), происходящие между состояниями. Ребра имеют весовые коэффициенты, соответствующие частотам отказов или частотам восстановлений. Фундаментальное свойство однородных процессов Маркова заключается в том, что будущее состояние зависит только от настоящего состояния, то есть переход из состояния N к последующему состоянию $N + 1$ не зависит от состояния, предшествующего состоянию $N + 1$. Это означает, что все вероятностные законы моделей экспоненциальны.

События, состояния и частоты отказов могут быть детализированы так, что может быть получено точное описание системы, например, обнаруженные или необнаруженные отказы, обнаружение наибольшего отказа и т. п. Интервалы контрольных проверок также могут быть смоделированы должным образом при помощи так называемых многофазных процессов Маркова, где вероятности состояний в конце одной фазы (например, как раз перед контрольным испытанием) могут быть использованы для вычисления начальных условий для следующей фазы (например, вероятности различных состояний после того, как контрольная проверка была выполнена).

Метод Маркова подходит для моделирования многих систем, уровень избыточности которых изменяется со временем вследствие нахождения компонента в состоянии отказа или восстановления. Другие классические методы, например, анализ видов и последствий отказов (FMEA) и FTA, не могут быть адаптированы к моделированию влияний отказов в течение ЖЦ системы, поскольку не существует простой комбинаторной формулы для вычисления соответствующих вероятностей.

Как правило, такую формулу, описывающую вероятности системы, можно найти в литературе или вывести самостоятельно. В более сложных случаях существуют методы упрощения (то есть сокращение числа состояний).

Однородный граф Маркова описывается системой линейных дифференциальных уравнений с постоянными коэффициентами. В результате серьезного анализа таких систем уравнений для их решения были разработаны доступные мощные алгоритмы. Поэтому с увеличением размера модели очень эффективно использовать вышеупомянутые алгоритмы, которые реализованы в виде различных пакетов ПО.

Размер графа растет экспоненциально с числом компонентов, это так называемый комбинаторный взрыв. Поэтому данный метод применим без аппроксимаций только для небольших систем.

Если законы распределения неэкспоненциальные (полумарковские процессы), то необходимо использовать метод моделирования Монте-Карло (см. Б.6.5.8).

Подробное описание данного метода/средства приведено в [74].

Б.6.5.7 Структурные схемы надежности

Цель — моделирование Э/Э/ПЭ СБЗС системы в форме структурных схем наборов событий, которые должны происходить, и условий, которые должны быть удовлетворены для успешного выполнения операций системы или задач. Данный метод в большей степени является методом представления, чем методом анализа.

Примечание — Данный метод/средство используется в В.6.4 «Структурные схемы надежности»; на него дана ссылка в ГОСТ 34332.4—2021 (приложение В, таблица В.10).

Описание. Данный метод позволяет сформировать успешный маршрут, состоящий из блоков, линий и логических переходов. Такой успешный маршрут начинается от одной стороны структурной схемы и проходит через блоки и логические переходы до другой ее стороны. Блок представляет собой условие или событие; маршрут проходит через него, если условие истинно или событие произошло. Когда маршрут подходит к логическому переходу, то его продолжают, если критерий логического перехода выполняется. Если маршрут достигает какой-либо вершины, то он может быть продолжен по всем исходящим из нее путям. Если существует по меньшей мере один успешный маршрут через всю структурную схему, то цель анализа считают достигнутой.

Данный метод позволяет сформировать структурное представление моделируемой системы. Эта структура напоминает электрическую схему, в которой ток протекает от входа к выходу, что означает, что моделируемая система работает должным образом. Если в схеме есть разрыв, то это означает, что в моделируемой системе произошел отказ. В результате появляется концепция наборов минимальных сечений, которые представляют комбинации отказов (т. е. места, где структурная схема надежности имеет «разрыв»), приводящих к отказу моделируемой системы.

Математически данный метод подобен методу дерева отказов. Он представляет собой логическую функцию, связывающую состояния отдельных компонентов (не работающих или работающих) с состоянием всей системы (не работающей или работающей). Поэтому вычисления подобны тем, которые описаны для дерева отказов.

Подробное описание данного метода/средства приведено в [75].

Б.6.5.8 Моделирование методом Монте-Карло

Цель — моделирование реальных ситуаций методом генерации случайных чисел, когда аналитические методы не применимы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.4).

Описание. Моделирование методом Монте-Карло используют для решения двух классов задач:

- вероятностных, при которых для генерации стохастических явлений используют случайные числа;
- детерминистических, при которых появление явлений математически преобразуется в эквивалентную вероятностную форму (например, интегральные вычисления).

При моделировании методом Монте-Карло используют случайные числа для анимации модели поведения правильно и неправильно функционирующей исследуемой системы. Такие поведенческие модели реализуют моделями переходов состояний (граф Маркова, сети Петри, формальные языки и т. д.). Моделирование Монте-Карло позволяет получить большую статистическую выборку, из которой формируются статистические результаты.

При использовании моделирования Монте-Карло необходимо принять меры по обеспечению разумных перекосов, допусков и шумов. Этим необходимо управлять через доверительный интервал, который легко может быть получен из моделирования. В отличие от аналитических методов моделирование Монте-Карло является самоаппроксимирующимся. Для упрощения модели незначительные события просто не рассматривают без необходимости их идентификации.

Общие принципы моделирования методом Монте-Карло заключаются в переформулировании проблемы так, чтобы полученные результаты были как можно более точными, что позволяет отказаться от решения проблемы в ее исходной постановке.

В контексте настоящего стандарта моделирование Монте-Карло может быть использовано для вычислений УПБ и учета неопределенности данных о надежности. Используя современные компьютеры, можно легко выполнить моделирование системы с УПБ 4.

Подробное описание данного метода/средства приведено в [76].

Б.6.5.9 Модели дерева отказов

Цель — построение логической функции, связывающей базовые события (виды отказов) с главным событием (нежелательное событие) на основе систематического нисходящего графического (следствие — причина) подхода.

Примечание — Применение дерева отказов в качестве средства подтверждения соответствия безопасности уже было описано в Б.6.5.5. Данный метод также широко используется для анализа отказов и вероятностных расчетов.

Описание. Это одновременно и метод анализа, помогающий аналитику шаг за шагом разработать модель, и математическая модель для вероятностных расчетов. Данный метод позволяет выполнять:

- качественный анализ путем выявления и сортировки сценариев отказов (минимальные сечения или простейшие импликанты);
- полукачественный анализ путем ранжирования сценариев в соответствии с их вероятностями возникновения;
- количественный анализ путем расчета вероятности главного события.

Подобно структурным схемам надежности, дерево отказов представляет собой логическую (булеву) функцию, связывающую состояния индивидуальных компонентов (не работает или работает) с состоянием всей системы (не работает или работает). Если компоненты являются независимыми, то вероятностные расчеты для логической функции могут быть выполнены только с учетом вероятностных свойств базовых компонентов. Это непросто, поскольку это статическая модель в основном работает только с постоянными вероятностями. Расчет вероятностей, зависимых от времени, должен быть проведен особенно внимательно. Например, PFDavg СБС, включающих периодическое контрольное тестирование компонентов, не может быть рассчитан непосредственно, кроме того, еще более сложно рассчитать PFH для систем безопасности, работающих в непрерывном режиме. Поэтому расчеты значений неготовность/PFD и ненадежность/PFH с помощью данного метода должны проводить только инженеры по надежности с глубоким пониманием математики, лежащей в основе данного метода.

Для очень простых деревьев отказов расчеты могут быть проведены вручную, однако за последние 50 лет было разработано и реализовано довольно большое количество алгоритмов для решения сложных логических уравнений. Наиболее современным на текущий момент является метод двоичных диаграмм решений (Binary Decision Diagrams, BDD), который основан на технике компактного кодирования логических уравнений в памяти компьютера. В настоящее время это единственный метод, способный выполнять вероятностные расчеты без приближений для систем промышленных размеров. Он также достаточно эффективен для обработки неопределенностей при моделировании методом Монте-Карло.

Подробное описание данного метода/средства приведено в [70], [77], [78].

Б.6.5.10 Обобщенные стохастические модели сетей Петри

Цель — графическое построение модели поведения правильно и неправильно функционирующей Э/Э/ПЭ СБЗС системы, максимально приближенной к реальной моделируемой системе для обеспечения эффективной поддержки моделирования методом Монте-Карло.

Примечание — Метод сетей Петри был описан в Б.2.3.3 как полуформальный метод. Данный метод также может быть эффективно использован для анализа полноты безопасности АС.

Описание. В методе применен асинхронный конечный автомат, описанный в Б.2.3.3, за исключением того, что хорошее свойство, отслеживаемое при полуформальном подтверждении соответствия, не существует, когда моделируется поведение неправильно функционирующей системы безопасности. Так называемые позиции (изображаются кружками) представляют возможные состояния, а так называемые переходы (изображаются прямоугольниками) представляют события, которые могут произойти. Кроме маркирования позиций (см. Б.2.3.3), могут быть использованы сообщения или предикаты для подтверждения соответствия (активизации) переходов, а продолжительность задержки между активизацией перехода и его «возбуждением» может быть детерминированной или стохастической величиной. Поэтому такие сети Петри называются «обобщенными стохастическими» сетями Петри.

Сети Петри являются гибкими поведенческими моделями, которые подтверждают свою высокую эффективность для поддержки моделирования методом Монте-Карло (см. Б.6.5.8). Кроме точности самого метода Монте-Карло, которая всегда известна, преодолеваются все ограничения других методов (зависимости, комбинаторный взрыв, неэкспоненциальность законов распределения и т. д.). Применение метода с использованием современных компьютеров позволяет решить проблемы даже для оценки УПБ 4.

Подробное описание данного метода/средства приведено в [79]—[81].

Б.6.6 Анализ наихудшего случая

Цель — исключение систематических ошибок, возникающих в результате неблагоприятных сочетаний условий окружающей среды и допусков на параметры компонентов Э/Э/ПЭ СБЗС системы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6).

Описание. Эксплуатационные возможности системы и параметры компонентов исследуют или вычисляют теоретически. При этом для условий окружающей среды задают их допустимые предельные значения. Анализируют и сопоставляют со спецификацией наиболее существенные характеристики системы.

Подробное описание данного метода/средства приведено в [82].

Б.6.7 Расширенное функциональное тестирование

Цель — обнаружение отказов на стадиях разработки спецификации, проектирования и разработки Э/Э/ПЭ СБЗС системы; проверка поведения системы в случае редких или неучтенных входов.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6).

Описание. Расширенное функциональное тестирование применяют для проверки функционального поведения Э/Э/ПЭ СБЗС системы как реакцию на входные условия, которые ожидаются только в редких случаях (например, глобального отказа) или не охватываются спецификацией СБС (например, некорректные операции). Для редко встречающихся условий наблюдаемое поведение системы сравнивают со спецификацией. В тех случаях, когда реакция системы не указана, следует убедиться в том, что заданная безопасность сохраняется в наблюдаемой реакции системы.

Подробное описание данного метода/средства приведено в [58], [83].

Б.6.8 Анализ наихудших случаев

Цель — проверка ситуаций, специфицированных при анализе наихудших случаев.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6).

Описание. Эксплуатационные возможности Э/Э/ПЭ СБЗС системы и параметры компонентов проверяют при анализе наихудших случаев. При этом для условий окружающей среды задают их предельно допустимые значения. Анализируют и сопоставляют со спецификацией наиболее существенные характеристики системы.

Б.6.9 Испытания с введением неисправностей

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение Б, таблицы Б.5 и Б.6).

Цель — внесение или имитация неисправностей в АС Э/Э/ПЭ СБЗС системы и документирование реакции системы.

Описание. Это качественный метод оценки надежности. Для описания местоположения и типа неисправностей, а также способа их внесения предпочтительно используются детализированные функциональные блоки, схемы и схемные диаграммы: например, питание может не поступать на различные модули; линии питания, линии общей шины или адресные линии могут быть разомкнуты/коротко замкнуты; компоненты или их порты могут быть разомкнуты или закорочены; реле могут быть замкнуты или разомкнуты, либо их действия могут выполняться в несоответствующие моменты времени и т. д. Возникающие в результате отказы системы классифицируют. Обычно вводят одиночные неисправности в устойчивом состоянии системы. Однако в случае, если неисправность не обнаруживается тестом встроенной диагностики или оказывается неочевидной, она может сохраниться в системе и вызвать следующую неисправность. При этом количество неисправностей может быстро возрасти многократно.

Такие испытания проводятся многопрофильным коллективом специалистов. Проектировщик (поставщик) системы должен при этом присутствовать и получать рекомендации. Для отказов, приводящих к опасным последствиям, вычисляют и оценивают среднее время наработки на отказ. Если это время мало, необходима модификация системы.

Подробное описание данного метода/средства приведено в [69], [84].

Приложение В
(справочное)

Методы и средства достижения полноты безопасности программного обеспечения

В.1 Общие положения

В настоящем приложении приведено краткое описание методов и средств достижения полноты безопасности ПО Э/Э/ПЭ СБЗС систем, на которые даны ссылки в ГОСТ 34332.4, дан их анализ, а также приведены ссылки на источники с подробным описанием данных методов и средств. Настоящее приложение не должно рассматриваться как полное или исчерпывающее.

В.2 Требования и детальное проектирование

Цель — формулирование методов/средств, применимых на стадиях подготовки требований (спецификации) и детального проектирования.

Примечание — Соответствующие методы и средства приведены в Б.2 приложения Б.

В.2.1 Структурные методы

Цель — формулирование методов/средств, применительно к структурным методам.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблицы А.2 и А.4).

В.2.1.1 Общие положения

Цель — основная цель методов анализа структуры (структурных методов) состоит в обеспечении качества разработки программного обеспечения. Данные методы в основном используются на ранних стадиях ЖЦ создаваемой Э/Э/ПЭ СБЗС системы. Структурные методы используют как точные, так и интуитивные процедуры, и нотации (поддерживаемые компьютерами), а также определяют и позволяют документально оформлять требования и возможности реализации в логической последовательности и структурированным способом.

Описание. Существует достаточно много структурных методов. Некоторые из них созданы для выполнения традиционных функций обработки данных и транзакций, другие в большей степени ориентированы на процессы управления и задачи реального времени (для систем, реализующих такие задачи, характеристика безопасности является более критичной, чем для других систем). Унифицированный язык моделирования UML (см. В.3.12) содержит много примеров структурированных нотаций.

Структурные методы можно считать «интеллектуальными инструментами», предназначенными для обобщенного восприятия и структуризации конкретной проблемы или системы. К их основным свойствам относятся:

- использование логики в рассуждениях и выводах, декомпозиция сложной проблемы на управляемые стадии;
- анализ и документальное представление всей системы, включая окружающую среду, а также разрабатываемую систему;
- декомпозиция данных и функций в разрабатываемой системе;
- использование контрольных таблиц, то есть списков типов объектов, нуждающихся в анализе;
- малая интеллектуальная перегрузка — простота, интуитивность и практичность при представлении проблемы или системы;
- акцентирование внимания на разработке структурной модели создаваемой системы с поддержкой CASE-средств для полноты метода.

Нотации, используемые для анализа и документирования проблем и объектов системы (например, на основе процессов и потоков данных), ориентированы на строгость представления, однако нотации для выражения функций обработки, выполняемых этими объектами, являются более неформальными. В то же время некоторые методы частично используют формальные нотации (например, регулярные выражения или конечные состояния автоматов). Увеличение точности нотации не только повышает уровень понимания, но и обеспечивает возможность автоматизированной обработки.

Другим преимуществом структурных нотаций является их наглядность, которая позволяет пользователю интуитивно проверять возможности спецификации или проекта при неполной информации.

Настоящий краткий обзор описывает несколько структурных методов более подробно.

Подробное описание данного метода/средства приведено в [56].

В.2.1.2 Управляемое представление требований (CORE)

Цель — обеспечение того, чтобы все требования были определены и выражены.

Описание. Данный метод предназначен для устранения разрыва между потребителем/конечным пользователем и аналитиком. Он не основан на математически строгой теории, а является средством коммуникации. Метод CORE создан для представления требований, а не для спецификаций. Данный метод является структурированным, все его представления проходят через различные уровни уточнений. Метод CORE используется для широкого

круга задач, учитывает сведения об окружающей среде, в которой система функционирует, а также различные точки зрения разных категорий пользователей. Метод CORE содержит руководящие материалы и тактические подходы для упрощения сложных проектов. Такое упрощение может быть скорректировано либо явным образом идентифицировано и документально оформлено. Таким образом, спецификации могут быть неполными, однако выявленные нерешенные проблемы и области высокого риска должны быть рассмотрены при последующем проектировании.

Подробное описание данного метода/средства приведено в [56], [85].

В.2.1.3 Метод разработки системы по Джексону (JSD)

Цель — разработка метода, охватывающего создание программных систем от стадии формирования требований до стадии кодирования, специально для систем реального времени.

Описание. Метод разработки системы по Джексону (JSD) представляет собой поэтапную процедуру разработки, в которой разработчик моделирует поведение реального мира, которое представляется функциями системы, определяет эти функции, вводит их в модель и преобразует образовавшуюся в результате спецификацию, которая реализуема в планируемой среде. Поэтому данный метод охватывает традиционные этапы, такие как создание спецификаций, проектирование и разработка, но несколько отличается от традиционных методов и не является методом нисходящего проектирования.

В данном методе большое внимание уделяется выявлению на ранней стадии сущностей реального мира, относящихся к создаваемой системе, а также моделированию этих сущностей и того, что может с ними произойти. Как только анализ «реального мира» будет выполнен и создана его модель, анализируют функции системы с тем, чтобы определить, как они вписываются в модель «реального мира». Модель результирующей системы дополняют структурным описанием всех процессов модели и затем преобразуют в программы, которые могут работать в заданной программно-аппаратной среде.

Подробное описание данного метода/средства приведено в [86], [87].

В.2.1.4 Метод Йордона для систем реального времени

Цель — спецификация и проектирование систем реального времени.

Описание. Метод Йордона применяют для реализации процесса разработки Э/Э/ПЭ системы, состоящего из трех этапов. На первом этапе создают «сущностную модель», которая описывает поведение системы в целом. На втором этапе строят модель реализации, описывающей структуры и механизмов, которые, являясь реализованными, отражают требуемое поведение системы. На третьем этапе осуществляют фактическое построение АС и ПО системы. Три этапа строго соответствуют традиционным этапам — разработке спецификации, проектированию и разработке, но главное, что проектировщик на каждом этапе должен активно заниматься моделированием.

Сущностная модель состоит из двух частей:

- модели окружающей среды, содержащей описание границ между системой и ее окружением, а также внешних событий, на которые должна реагировать система;
- модели поведения, которая содержит схемы, описывающие преобразования, выполняемые системой в ответ на события, и описание данных, которые система должна содержать для выдачи откликов.

Модель реализации подразделяют на две подмодели, описывающие распределение отдельных процессов в процессорах и декомпозицию процессов на программные модули.

Для создания сущностной модели и модели реализации данный метод использует множество хорошо известных подходов: построение диаграмм потоков данных, преобразование графов, структурированный язык, диаграммы переходов состояний и сети Петри. Кроме того, данный метод содержит методики для моделирования, представленного из уже сформированных моделей проекта системы или вручную (на бумажном носителе), либо автоматически.

Подробное описание данного метода/средства приведено в [88].

В.2.2 Диаграммы потоков данных

Цель — программная поддержка описания потока данных в виде диаграмм.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.5 и Б.7).

Описание. Диаграммы потоков данных описывают преобразование входных данных в выходные для каждого компонента схемы, представляющего различные преобразования.

Диаграммы потоков данных состоят из трех компонентов:

- аннотированные стрелки — обозначают поток данных, входящих и исходящих из блоков преобразования, с кратким описанием этих данных;
- аннотированные кружки — обозначают блоки преобразования с кратким описанием преобразований;
- операторы (and, xor) — операторы, используемые для связи аннотированных стрелок.

Каждый аннотированный кружок на диаграмме потока данных может рассматриваться как самостоятельный блок, который при появлении на его входах данных преобразует их в выходные. Одним из основных преимуществ является то, что они показывают преобразования, не предполагая, как они реализуются. Чистая диаграмма потоков данных не включает в себя управляющую информацию или информацию о последовательности процесса, так как управление реализуется в расширениях для реального времени, как в методе Йордона для систем реального времени (см. В.2.1.4).

Создание диаграмм потока данных является наилучшим подходом при анализе систем в направлении от входов к выходам. Каждый кружок на диаграмме должен обозначать разное преобразование — его выходы должны отличаться от его входов. Не существует правил определения общей структуры диаграммы, и создание диаграммы потока данных является одним из творческих аспектов создания проекта системы в целом. Подобно всем проектам, процедура, уточняющая начальную диаграмму для создания конечной, является итеративной.

Подробное описание данного метода/средства приведено в ГОСТ 19.701, [59], [60], [89].

В.2.3 Структурные диаграммы

Цель — представление структуры программы в виде схемы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.5).

Описание. Структурные диаграммы дополняют диаграммы потоков данных. Они описывают программируемую систему и иерархию ее компонентов, а также отображают их графически в виде дерева. Структурные диаграммы описывают способ реализации элементов диаграммы потоков данных в виде иерархии программных модулей.

Структурная диаграмма показывает взаимоотношения между программными модулями, не указывая при этом порядок активизации программных модулей. Структурные диаграммы изображаются с использованием следующих четырех символов:

- прямоугольника с именем модуля;
- линии, соединяющей эти прямоугольники, формирующие структуру;
- стрелки, отмеченной незаштрихованным кругом, с именем данных, передаваемых в направлении элементов структурной диаграммы и обратно (как правило, такая стрелка изображается параллельно линиям, соединяющим прямоугольники схемы);
- стрелки, отмеченной заштрихованным кружком, с именем сигнала управления, проходящего в структурной диаграмме от одного модуля к другому, и эта стрелка также изображается параллельно линии, соединяющей два модуля.

Из любой нетривиальной диаграммы потока данных можно создать множество различных структурных диаграмм.

Диаграммы потоков данных отображают взаимоотношение между информацией и функциями системы. Структурные диаграммы отображают способ реализации элементов системы. Оба метода представляют собой обоснованные, хотя и различные точки зрения на конкретную систему.

Подробное описание данного метода/средства приведено в [59], [60], [90].

В.2.4 Формальные методы

Цель — формулирование методов/средств, применимых для формальных методов.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблицы А.1, А.2, А.4 и приложение Б, таблица Б.5).

В.2.4.1 Общие положения

Цель — разработка программных средств, основанных на математических принципах. К этим средствам относятся методы формального проектирования и формального кодирования.

Описание. На основе формальных методов разработаны средства описания системы для решения отдельных задач на этапах разработки спецификации, проектирования или реализации. Создаваемое в результате описание представляет собой строгую нотацию, математически анализируемую для обнаружения различных видов несогласованностей или некорректностей. Более того, такое описание может быть в некоторых случаях проанализировано автоматически по аналогии с проверкой компилятором синтаксиса исходной программы, или использована анимация в целях показать различные аспекты поведения описываемой системы. Анимация может дать дополнительную уверенность в том, что система соответствует как реальным, так и формально специфицированным требованиям, поскольку это улучшает восприятие человеком специфицированного поведения системы.

Формальный метод обычно предлагает нотацию (как правило, используется один из методов дискретной математики), метод вывода описания в данной нотации и различные методы анализа описания для проверки корректности различных свойств системы.

Ряд формальных методов CCS, CSP, HOL, LOTOS, OBJ, временная логика, VDM и Z описан в настоящего пункте. Другие методы, например, метод конечных автоматов и сети Петри (см. приложение В), в зависимости от корректности использования методами соответствующего математического аппарата, могут рассматриваться как формальные.

Подробное описание данного метода/средства приведено в [91]—[93].

В.2.4.2 Расчет взаимодействующих систем — CCS

Цель — описание и анализ поведения систем, реализующих параллельные коммуникационные процессы.

Описание. Расчет взаимодействующих систем (CCS) — это применение математического аппарата, описывающего поведение систем. Проект системы моделируют в виде сети независимых процессов, реализующихся последовательно или параллельно. Процессы могут взаимодействовать через порты (аналогичные каналам CSP), и взаимодействие осуществляется только при готовности обоих процессов. Может быть смоделировано отсутствие

детерминизма. Начиная с описания всей системы на высоком уровне абстрагирования (трассирование), можно выполнять пошаговое уточнение системы (стратегия сверху вниз) в рамках композиции взаимодействующих процессов, общее поведение которых формирует также поведение всей системы. В равной степени можно выполнять и стратегию «снизу-вверх», комбинируя процессы и получая в результате необходимые свойства формируемой системы, используя правила вывода композиционного типа.

Подробное описание данного метода/средства приведено в [94].

В.2.4.3 Взаимодействующие последовательные процессы — CSP

Цель — спецификация конкурирующих программных систем, то есть систем, процессы которых реализуются одновременно.

Описание. Метод взаимодействующих последовательных процессов (CSP) обеспечивает язык для спецификаций процессов системы и подтверждения соответствия реализации процессов их спецификациям (описанным как трасса, то есть допустимая последовательность событий).

Систему моделируют в виде сети независимых процессов, составленных последовательно или параллельно. Каждый независимый процесс описывают в терминах всех его возможных поведений. Независимые процессы могут взаимодействовать (синхронно или обмениваться данными) через каналы, и взаимодействие происходит только при готовности обоих процессов. Может быть промоделирована относительная синхронизация событий.

Теоретические положения метода CSP были непосредственно включены в архитектуру транспьютера INMOS, а язык OCCAM позволил непосредственно реализовывать на сетях транспьютеров системы, специфицированные в языке CSP.

Подробное описание данного метода/средства приведено в [95].

В.2.4.4 Логика высшего порядка — HOL

Цель — спецификация и верификация аппаратных средств.

Описание. Логика высшего порядка (HOL) представляет собой конкретную логическую нотацию и систему, которая ее автоматически поддерживает. Логическая нотация взята в основном из простой теории типов Черча, а машинная реализация основана на теории логики вычислимых функций (LCF).

Подробное описание данного метода/средства приведено в [96].

В.2.4.5 Язык упорядоченных во времени процессов LOTOS

Цель — описание и анализ поведения систем, реализующих параллельные коммуникационные процессы.

Описание. Язык для спецификации процессов, упорядоченных во времени (LOTOS), основан на CCS с дополнительными возможностями из близких алгебраических теорий CSP и CIRCAL (теория цепей). В языке LOTOS преодолены недостатки CCS в управлении структурами данных и представлении значений выражений благодаря объединению его с аспектами языка абстрактных типов данных ACT ONE. Процесс описания аспектов в LOTOS может быть также использован для других формальных методов при описании абстрактных типов данных.

Подробное описание данного метода/средства приведено в [97].

В.2.4.6 Язык спецификаций OBJ

Цель — обеспечение точной спецификации системы в процессе диалога с пользователем и подтверждение соответствия системы до ее реализации.

Описание. OBJ представляет собой алгебраический язык спецификаций. Пользователи определяют требования в терминах алгебраических выражений. Системные аспекты (поведение или конструктивы) специфицируются в терминах операций, действующих над абстрактными типами данных (ADT). ADT подобен языку Ada, где поведение оператора наблюдаемо, однако подробности реализации скрыты.

Спецификация OBJ и последующая пошаговая реализация подвергаются тем же формальным методам проверки, что и другие формальные методы. Более того, поскольку конструктивные аспекты спецификации OBJ автоматически исполнимы, существует непосредственная возможность подтверждения соответствия системы на основе самой спецификации. Исполнение — это оценка функций системы посредством подстановки выражений (перезаписыванием), которая продолжается до тех пор, пока не будут получены конкретные выходные значения. Эта исполнимость позволяет конечным пользователям рассматриваемой системы получать «облик» планируемой системы на этапе создания ее спецификации без необходимости знакомства с методами, лежащими в основе формальных спецификаций.

Как и все другие методы ADT, метод OBJ применим только к последовательным системам или к последовательным аспектам параллельных систем. Метод OBJ применяют для спецификации как малых, так и крупных промышленных применений.

Подробное описание данного метода/средства приведено в [98].

В.2.4.7 Временная логика

Цель — непосредственное выражение требований к безопасности и эксплуатации, а также формальное представление сохранения этих качеств на последующих этапах разработки.

Описание. Стандартная предикатная логика первого порядка не содержит концепций времени. Временная логика расширяет логику первого порядка добавлением модальных операторов (например, «с этого момента» и «случайно»). Эти операторы могут использоваться для уточнения суждений о системе. Например, свойства безопасности могут потребовать использовать модальный оператор «с этого момента», но может потребоваться, чтобы и другие необходимые состояния системы были достигнуты «случайно» из некоторого другого начального состо-

яния. Временные формулы интерпретируются последовательностями состояний (поведениями). Представление состояния зависит от выбранного уровня описания. Оно может относиться ко всей системе, системным элементам или компьютерной программе.

Квантифицированные временные интервалы и ограничения во временной логике явно не обрабатываются. Абсолютное время обрабатывается посредством образования дополнительных временных состояний, что является частью описания состояния.

Подробное описание данного метода/средства приведено в [99].

В.2.4.8 Программы реального времени VDM, VDM++ — метод разработки Vienna

Цель — систематическая спецификация и реализация последовательных (VDM) и параллельных (VDM++) программ реального времени.

Описание. VDM — это математический метод спецификации и уточнения реализаций, который позволяет доказать их корректность по отношению к спецификации.

В этом основанном на модели методе спецификации состояние системы моделируют в терминах теоретико-множественных структур, в которых описаны инварианты (предикаты), а операции над этим состоянием моделируют посредством их пред- и постусловий в терминах системных состояний. Операции могут быть проверены на сохранение системных инвариантов.

Выполнение спецификаций осуществляют путем реализации состояния системы в терминах структур данных в заданном языке и уточнения операций в терминах программы на заданном языке. Этапы реализации и уточнения позволяют логически вывести свойства, устанавливающие корректность этих этапов. Выполняются или нет эти свойства, определяет разработчик.

В принципе VDM используют на этапе создания спецификации, но может быть использован на этапах проектирования и реализации исходного кода. VDM может быть также применен к последовательно структурированным программам или к последовательным процессам в параллельных системах.

Объектно-ориентированное и параллельное для реального времени расширение VDM — VDM++ представляет собой язык формализованных спецификаций, основанный на языке VDM-SL, созданном в ИСО, и на объектно-ориентированном языке Smalltalk.

VDM++ имеет широкий диапазон конструкций, что позволяет пользователю формально специфицировать параллельные системы реального времени в объектно-ориентированной среде. В VDM++ полная формальная спецификация содержит совокупность спецификаций классов и отдельных характеристик рабочего пространства.

К средствам описания реального времени на языке VDM++ относятся:

- временные выражения, предусмотренные для представления как текущего момента, так и момента вызова метода внутри тела метода;

- выражение, описывающее синхронизирующий сигнал, которое может быть добавлено к методу для спецификации верхних (или нижних) пределов времени исполнения для корректности реализаций;

- переменные непрерывного времени, которые должны быть введены. С условными операторами и операторами действия допускается специфицировать отношения (например, дифференциальные уравнения) между этими временными функциями, что оказалось очень полезно при спецификации требований к системам, действующим в среде с непрерывным временем. Уточняющие шаги приводят к дискретным программным решениям для программ реального времени.

Подробное описание данного метода/средства приведено в [100], [101].

В.2.4.9 Язык Z

Цель — оказание помощи пользователю в применении нотации языка спецификаций Z для последовательных систем и метода проектирования, позволяющего разработчику выполнять работу, начиная со спецификации на языке Z до исполнительных алгоритмов, обеспечивая при этом доказательство их корректности по отношению к спецификации.

Язык Z в принципе используют на этапе создания спецификации, однако данный язык был разработан для использования от этапа составления спецификации до проектирования и реализации систем. Более всего он подходит для разработки последовательных систем, ориентированных на данные.

Описание. Как и в VDM, в реализованном языке Z спецификацию состояний системы моделируют в терминах теоретико-множественных структур, в которых описаны инварианты (с использованием предикат), а операции над этими состояниями моделируют посредством определения их пред- и постусловий в терминах системных состояний. Операции допускается проверять на сохранение системных инвариантов для демонстрации их согласованности. Формальная часть спецификации подразделяется на схемы, которые обеспечивают возможность структурирования спецификаций посредством их усовершенствования.

Обычно спецификация Z представляет собой сочетание формального текста на языке Z и неформального пояснительного текста на естественном языке. Формальный текст сам по себе может оказаться слишком сжатым для простого восприятия, и часто его смысл необходимо пояснять, тогда как неформальный, естественный язык может оказаться неоднозначным и неточным.

В отличие от VDM язык Z представляет собой скорее нотацию, чем завершенный метод. Был разработан близкий метод (метод B), который может быть использован в сочетании с языком Z. Метод B основан на принципе пошагового уточнения.

Подробное описание данного метода/средства приведено в [102], [103].

В.2.5 Программирование с защитой

Цель — создание программ, выявляющих во время их исполнения аномальные потоки управления, данных или значения данных и реагирующих на них заранее определенным и приемлемым способом.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.4).

Описание. В процессе разработки программ допускается использовать разные методы для проверки аномалий в потоках управления или данных. Эти методы могут быть применены систематически в процессе программирования системы для снижения вероятности ошибочной обработки данных.

Существуют два пересекающихся множества методов защиты. Внутренние методы защиты от ошибок проектируют в ПО для преодоления недостатков в процессе создания ПО. Эти недостатки могут быть обусловлены ошибками при проектировании или кодировании либо ошибочными требованиями. Ниже перечислены некоторые из рекомендаций по защите:

- проверка диапазона значений переменных;
- проверка достоверности значений переменных (если возможно);
- проверка типа, размерности и диапазона значений параметров процедур при вводе процедур.

Представленные три рекомендации помогают гарантировать допустимость значений, обрабатываемых в программах, как с точки зрения терминов программных функций, так и физических значений переменных.

Параметры «только для чтения» и параметры «для чтения—записи» должны быть разделены, и должен быть установлен и проконтролирован доступ к ним. В программных функциях должны быть рассмотрены все параметры в качестве параметров «только для чтения». Символьные константы не должны быть доступны для записи. Это помогает обнаруживать случайные перезаписи или ошибочное использование переменных.

Устойчивое к ошибкам ПО проектируют в «предположении», что ошибки существуют в его собственном окружении либо используются выходящие за номиналы значения или предполагаемые условия, но ПО ведет себя заранее определенным способом. В этом случае применяют следующие проверки:

- проверку на достоверность физических значений входных и промежуточных переменных;
- проверку влияния выходных переменных, предпочтительно путем прямого наблюдения соответствующих изменений состояния системы;
- проверку самим ПО своей конфигурации, включая наличие и доступность предполагаемых АС, а также завершенность ПО, что особенно важно для поддержки его целостности в процессе эксплуатации.

Некоторые из методов защиты программ, например, проверки последовательности потока управления, также справляются и с внешними отказами.

Подробное описание данного метода/средства приведено в [56].

В.2.6 Стандарты по проектированию и кодированию

В.2.6.1 Общие положения

Цель — упрощение верификации для поддержания группового объективного подхода и установления стандартного метода проектирования.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.4).

Описание. В самом начале между участниками проекта должны быть согласованы необходимые правила, охватывающие рассмотренные ниже методы проектирования и разработки (например, JSP, сети Петри и т. д.), а также соответствующие стандарты кодирования (см. В.2.6.2).

Эти правила создают для облегчения разработки, верификации, оценки и эксплуатации. При этом должны учитываться доступные инструментальные средства, в частности, для аналитиков, а также развитие средств проектирования.

Подробное описание данного метода/средства приведено в [49].

В.2.6.2 Стандарты кодирования

Цель — сократить вероятность ошибок в разрабатываемом коде, связанном с безопасностью, и упростить его верификацию.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.1).

Описание. Приведенные в таблице В.1 принципы указывают, как связанные с безопасностью правила кодирования (для любого языка программирования) могут помочь в выполнении нормативных требований ГОСТ 34332.4 и в достижении информативных «требуемых свойств» (см. приложение Ж). Необходимо уделить внимание доступным инструментам поддержки.

В.2.6.3 Отказ от динамических переменных или динамических объектов

Цель — исключение:

- нежелательных или необнаруживаемых наложений в памяти;
- узких мест ресурсов в процессе (связанном с безопасностью) выполнения программы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблица А.2 и приложение Б, таблица Б.1).

Таблица В.1 — Требования и рекомендации ГОСТ 34332.4 и рекомендации стандартов кодирования

Требования и рекомендации ГОСТ 34332.4	Рекомендации стандартов кодирования
Модульный подход (таблица А.2, пункт 7, таблица А.4, пункт 4)	Ограничение размера программного модуля (таблица Б.9, пункт 1) и управление сложностью ПО (таблица Б.9, пункт 2). Примеры: - определение «локальных» метрик: размера, метрик сложности и предельных размеров (для модулей); - определение «глобальных» метрик сложности и предельных размеров (для всей структуры модулей); - ограниченное число параметров/фиксированное число параметров под программы (таблица Б.9, пункт 4). Ограничение доступа/инкапсуляция информации (таблица Б.9, пункт 3), например, стимулы для того, чтобы использовать определенные функции языка. Полностью определенный интерфейс (таблица Б.9, пункт 6). Примеры: - точная спецификация сигнатуры функции; - программирование с проверкой ошибок (таблица А.2, пункт 3а) и верификация данных (7.9.2.7) с явной спецификацией предварительных условий и постусловий для функций, утверждений, инвариантов типов данных
Понятность кода: - содействие понятности кода (7.4.4.13); - читаемый, понятный и тестируемый код (7.4.6)	Соглашения о присвоении имен, продвигающие значащие, однозначные имена, например, предотвращение имен, которые можно перепутать (например, IO и I0). Символьные имена для числовых значений. Процедуры и руководства для документирования исходного кода (7.4.4.13). Например, в них: - объяснить, почему и зачем (а не только что) кодируется; - описать предостережения; - описать побочные эффекты. Где это возможно, в исходный код следует включать следующую информацию (7.4.4.13): - данные о юридическом лице [например, компания, автор(ы) и т. д.]; - описание кода; - входные и выходные данные; - предысторию управления конфигурацией. (См. также модульный подход.)
Верифицируемость и тестируемость: - обеспечение верификации и тестирования (7.4.4); - способствовать обнаружению ошибок при проектировании или программировании (7.4.4.10); - формальная верификация (таблица А.5, пункт 9); - формальное доказательство (таблица А.9, пункт 1)	Для инструментальных средств поддержки следует контролировать: - обертки для «критических» библиотечных функций для проверки пред- и постусловий; - стимулы для использования функций языка, которые могут выразить ограничения на использование определенных элементов данных или функций (например, констант); - для инструментов, поддерживающих верификацию: правила выполнения ограничений для выбранных инструментов (если это не вредит более существенным целям); - ограниченное использование рекурсии (таблица Б.1, пункт 6) и других форм циклических зависимостей. (См. также модульный подход.)
Статическая верификация соответствия специфицированному проекту (7.9.2.12)	Методические рекомендации по кодированию для реализации специфицированных в проекте концепций или ограничений. Например: - методические рекомендации по кодированию циклов с гарантируемым максимальным значением времени цикла (таблица А.2, пункт 13а); - методические рекомендации по кодированию архитектуры с временным распределением (таблица А.2, пункт 13б); - методические рекомендации по кодированию архитектуры, управляемой событиями, с гарантируемым максимальным временем ответа (таблица А.2, пункт 13в);

Окончание таблицы В.1

Требования и рекомендации ГОСТ 34332.4	Рекомендации стандартов кодирования
	- применение циклов со статически определенным максимальным числом итераций (за исключением бесконечного цикла, предусмотренного в проекте); - методические рекомендации по кодированию статического распределения ресурсов (таблица А.2, пункт 14) и предотвращение динамических объектов (таблица В.1, пункт 2); - методические рекомендации по кодированию статической синхронизации доступа к совместно используемым ресурсам (таблица А.2, пункт 15); - методические рекомендации по кодированию с соблюдением ограничений на использование прерываний (таблица Б.1, пункт 4); - методические рекомендации по кодированию, не использующему динамические переменные (таблица Б.1, пункт 3а); - проверка установки динамических переменных в неавтономном режиме (таблица Б.1, пункт 3б); - методические рекомендации по кодированию, обеспечивающему совместимость с другими используемыми языками программирования (7.4.4.10); - методические рекомендации по прослеживаемости в проекте
Подмножество языков (таблица А.3, пункт 3): - запрет опасных функций языка (7.4.4.13); - использование только определенных функций языка (7.4.4.10); - структурное программирование (таблица А.4, пункт 6), - строго типизированный язык программирования (таблица А.3, пункт 2); - отсутствие автоматического преобразования типов (таблица В.1, пункт 8)	Исключение функций языка, приводящих к неструктурированным проектам. Например: - ограниченное использование указателей (таблица Б.1, пункт 5); - ограниченное использование рекурсии (таблица В.1, пункт 6); - ограниченное использование С-подобных объединений; - ограниченное использование Ada-подобных или С++-подобных исключений; - неиспользование неструктурированного потока управления в программах на языках более высокого уровня (таблица Б.1, пункт 7); - использование одной точки входа/одной точки выхода в подпрограммах и функциях (таблица Б.9, пункт 5); - неиспользование автоматического преобразования типов; - ограниченное использование побочных эффектов, не очевидных из сигнатур функций (например, статических переменных). Недопущение никаких побочных эффектов при оценке (вычислении) условий и во всех формах утверждений. Ограниченное или только документально оформленное использование специфичных для компилятора функций. Ограниченное использование конструкций языка, которые могут ввести в заблуждение. Использование правил, которые должны применяться при использовании этих функций языка
Хорошая практика программирования (7.4.4.13)	Когда это применимо: - методические рекомендации по кодированию, обеспечивающие, в случае необходимости, оценивание выражений с плавающей точкой (запятой) в правильном порядке (например, «*a*-*b*+*c*» не всегда равно «*a*+*c*-*b*»), - в сравнениях с плавающей точкой (запятой) — использование только неравенств (меньше чем, меньше или равно, больше чем, больше или равно) вместо строгого равенства; - методические рекомендации, относящиеся к условной компиляции и «препроцессорной обработке»; - систематическая проверка условий возврата (успех/отказ). Документирование и по возможности автоматизация создания исполняемого кода (make-файлы). Предотвращение побочных эффектов, не очевидных из сигнатур функций. Когда такие побочные эффекты существуют, в соответствии с методическими рекомендациями их необходимо документально оформить. Заключение в скобки, когда приоритет операторов не абсолютно очевиден. Поиск и локализация предположительно невозможных ситуаций (например, ситуация «по умолчанию» в «переключателях» языка С). Использование «оберток» для критических модулей, в частности, для проверки пред- и постсостояний и состояний возврата. Соблюдение методических рекомендаций по кодированию в условиях известных ошибок компилятора и в пределах, установленных оценкой компилятора

Описание. В случае применения этого метода динамические переменные и динамические объекты получают устанавливаемые во время выполнения программы определенные и абсолютные адреса в памяти. Объем (размер) распределяемой памяти и ее адреса зависят от состояния системы в момент распределения памяти и не могут быть проверены компилятором или другим автономным инструментом.

Так как число динамических переменных и объектов и существующее свободное пространство памяти для размещения новых динамических переменных или объектов зависит от состояния системы в момент их размещения, то при размещении или при использовании переменных или объектов возможны сбои. Например, если объем свободной памяти, распределяемый системой, недостаточен, то содержимое памяти другой переменной может быть неумышленно стерто. Если динамические переменные или объекты не используются, то появление этих ошибок исключено.

Необходимы ограничения на использование динамических объектов, если динамическое поведение не может быть точно предсказано с помощью статического анализа (то есть перед выполнением программы), и поэтому не может быть гарантировано предсказуемое выполнение программы.

В.2.6.4 Проверка создания динамических переменных или динамических объектов при выполнении программы

Цель — убедиться в том, что память, в которой должны быть размещены динамические переменные и объекты, свободна до ее загрузки, а размещение в ней динамических переменных и объектов во время выполнения программы не повлияет на уже существующие в ней переменные, данные или коды.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.1).

Описание. В случае применения этих средств к динамическим переменным относят те переменные, которые имеют свои конкретные и абсолютные адреса в памяти, устанавливаемые во время выполнения программы (в этом смысле динамические переменные являются также атрибутами объектов).

Память проверяют аппаратными или программными средствами, чтобы определить, свободна ли она до размещения в ней динамических переменных или объектов (например, для того, чтобы исключить переполнение стека). Если размещение не разрешается (например, если емкости памяти по конкретному адресу недостаточно), должны быть предприняты соответствующие действия. После использования динамических переменных или объектов (например, после выхода из подпрограммы) вся используемая ими память должна быть освобождена.

Примечание — Альтернативным методом является демонстрация статического распределения памяти.

В.2.6.5 Ограниченное использование прерываний

Цель — сохранение верифицируемости и тестируемости ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.1).

Описание. Использование прерываний должно быть ограничено. Прерывания могут быть использованы, если они упрощают систему. Использование ПО для обработки прерываний должно быть запрещено в критических ситуациях для выполняемых функций (например, критичность по времени, критичность изменений данных). Если прерывания используют, то следует установить максимальное время вычисления, в течение которого прерывание запрещено. Использование прерываний и их маскирование следует подробно документировать.

В.2.6.6 Ограниченное использование указателей

Цель — исключение проблем, связанных с доступом к данным без предварительной проверки типа и диапазона указателя; обеспечение модульного тестирования и верификации программных средств; ограничение последствия отказов.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.1).

Описание. В прикладном ПО арифметика указателей может быть использована на уровне исходного кода только в случае, если тип и диапазон значений указателя данных (для гарантии того, что ссылка указателя находится внутри корректного адресного пространства) будут проверены перед доступом. Межпроцессное взаимодействие в прикладных программах не должно быть осуществлено прямым доступом между задачами. Обмен данными следует осуществлять с помощью операционной системы.

В.2.6.7 Ограниченное использование рекурсий

Цель — исключение неверифицируемого и нетестируемого использования вызовов подпрограмм.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.1).

Описание. Если используют рекурсию, то должен быть определен четкий критерий, который делает глубину рекурсии предсказуемой.

В.2.7 Структурное программирование

Цель — проектирование и реализация программы с использованием практического анализа программы без ее выполнения. Программа может содержать только абсолютный минимум статистически нетестируемого поведения.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.4).

Описание. Для минимизации структурной сложности программы следует применять следующие принципы:

- разделять программу на подходящие, небольшие, минимально связанные программные модули, все взаимодействия между которыми точно специфицированы;
- составлять поток управления программными модулями с использованием таких структурированных конструкций, как последовательности, итерации и выбор;
- обеспечивать небольшое число возможных путей через программные модули и возможно более простые отношения между входными и выходными параметрами;
- исключать сложные ветвления и, в частности, безусловные переходы (goto) при использовании языков высокого уровня;
- по возможности связывать ограничения цикла и ветвление с входными параметрами;
- исключать использование сложных вычислений в ветвлении и цикле.

Следует использовать свойства языков программирования, которые способствуют указанным выше принципам, предпочитая их другим свойствам, которые считают более эффективными, за исключением случаев, когда эффективность приобретает абсолютный приоритет (например, некоторые критичные к безопасности системы).

Подробное описание данного метода/средства приведено в [104]—[106].

В.2.8 Ограничение доступа/инкапсуляция информации

Цель — предотвращение непреднамеренного доступа к данным или процедурам и обеспечение тем самым качественной структуры ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.9).

Описание. Общедоступные для всех программных компонентов данные могут быть случайно или некорректно модифицированы любым из этих компонентов. Любые изменения этих структур данных могут потребовать подробной проверки программного кода и серьезных исправлений.

Ограничение доступа представляет собой общий метод к минимизации указанных выше проблем. Ключевые структуры данных «скрыты», и с ними можно работать только через конкретный набор процедур доступа, это позволяет модифицировать внутренние структуры данных или добавлять новые процедуры и при этом не оказывать влияния на функциональное поведение остальных программных средств. Например, каталог имен директорий может иметь процедуры доступа «вставить», «удалить» и «найти». Процедуры доступа и структуры внутренних данных могут быть изменены (например, при использовании различных методов просмотра или запоминании имен на жестком диске), не оказывая влияния на логическое поведение остальных программных средств, использующих эти процедуры.

В данном случае следует использовать концепцию абстрактных типов данных. Если непосредственная проверка не предусмотрена, может оказаться необходимым проверить, не было ли случайно разрушено абстрагирование.

Подробное описание данного метода/средства приведено в [104], [105].

В.2.9 Модульный подход

Цель — декомпозиция программной системы на небольшие законченные модули в целях сокращения сложности системы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблица А.4 и приложение Б, таблица Б.9).

Описание. Модульный подход, или модуляризация, включает в себя несколько различных правил для этапов ЖЦ проекта ПО (проектирования, кодирования и эксплуатации). Эти правила могут быть различными в зависимости от реализуемых методов проектирования. Большинство методов подчиняются следующим правилам:

- программный модуль (или подпрограмма, что одно и то же) должен быть составлен так, чтобы выполнял одну четко сформулированную задачу или функцию;
- связи между программными модулями должны быть ограничены и строго определены, уровень связности каждого программного модуля должен быть высоким;
- совокупности подпрограмм следует строить так, чтобы обеспечивать несколько уровней программных модулей;
- размеры подпрограмм следует ограничить некоторыми конкретными значениями, обычно от двух до четырех размеров экрана;
- подпрограммы должны быть выполнены только с одним входом и одним выходом;

- программные модули должны быть такими, чтобы взаимодействовали с другими программными модулями через свои интерфейсы, где используются глобальные или общие переменные, которые должны быть хорошо структурированы; доступ к ним должен находиться под контролем, и их использование в каждом конкретном случае должно быть обосновано;

- все интерфейсы программных модулей должны быть полностью документально оформлены;

- все интерфейсы программных модулей должны быть выполнены так, чтобы они содержали только необходимые для их функционирования параметры. Однако эта рекомендация усложнена тем, что язык программирования может иметь параметры по умолчанию или тем, что использован объектно-ориентированный подход.

Подробное описание данного метода/средства приведено в [58], [105].

В.2.10 Использование доверительных/проверенных элементов программного обеспечения

Цель — исключение такого проектирования программных модулей и элементов, которое вызывало бы необходимость их интенсивных повторных проверок или перепроектирования для каждого нового применения. Использование преимущества проектов, которые не были формально или строго проверены, но для которых имеется продолжительный опыт эксплуатации. Использовать преимущества уже существующих программных элементов, которые были проверены для различных применений и для которых существует совокупность доказательств подтверждения соответствия.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблицы А.2, А.4 и приложение В, таблицы В.2, В.4).

Описание. Данный метод позволяет проверить наличие в программных компонентах систематических ошибок проектирования и/или эксплуатационных отказов.

Строить сложную систему, используя элементарные компоненты, нецелесообразно. Как правило, используют основные узлы («модули»), которые были разработаны ранее для обеспечения некоторых полезных функций и которые могут быть использованы для реализации некоторой части новой системы.

Хорошо продуманные и структурированные программируемые электронные системы состоят из ряда программных модулей, которые различаются между собой и взаимодействуют друг с другом вполне определенными способами. Формирование библиотеки таких общепримени мых программных модулей, которые можно повторно использовать в нескольких применениях, позволяет большую часть ресурсов, необходимых для подтверждения соответствия проектов, распределять по нескольким применениям.

Однако для применений, связанных с безопасностью, важно иметь достаточную уверенность в том, что новая система, включающая эти уже существующие модули, имеет необходимую полноту безопасности, и что безопасность новой системы не нарушается некоторым некорректным поведением уже существующих модулей.

Существуют два подхода, как обрести уверенность в том, что поведение уже существующих модулей точно известно:

- провести всесторонний анализ опыта эксплуатации модуля, чтобы продемонстрировать, что модуль был «проверен в эксплуатации»;

- оценить совокупность доказательств подтверждения соответствия, которая была собрана для поведения модуля, чтобы определить, соответствует ли этот модуль требованиям настоящего стандарта.

В.2.10.1 Проверка в эксплуатации

Только в редких случаях проверки в эксплуатации будет достаточно в качестве единственного средства, гарантирующего для доверительного модуля ПО достижение им необходимого УПБ. Для сложных модулей со многими возможными функциями (например, операционной системы) важно установить, какая из функций достаточно проверена при ее использовании. Например, процедуру самотестирования для обнаружения ошибок, если в период ее эксплуатации не появилось отказов, нельзя рассматривать как проверенную в эксплуатации.

Программный модуль может быть признанным проверенным в эксплуатации, если он соответствует следующим критериям:

- спецификация не менялась;

- использовался в системах в различных областях применения;

- продолжительность срока его эксплуатации не менее года;

- продолжительность эксплуатации соответствует УПБ или соответствующему числу запросов; для демонстрации частоты отказов, не связанных с безопасностью, менее чем:

- 10^{-2} на один запрос (в год) с 95 %-ным уровнем доверия [необходимо 300 эксплуатационных прохождений (в год)];

- 10^{-5} на один запрос (в год) с 95 %-ным уровнем доверия [необходимо 690000 эксплуатационных прохождений (в год)].

Примечание — Математический аппарат, обеспечивающий числовые оценки данного метода, приведен в приложении Г. Аналогичный метод и статистический подход изложены также в Б.5.4;

- весь опыт эксплуатации связан с известным профилем запросов функций программного модуля для гарантии того, что увеличивающийся опыт эксплуатации действительно приводит к увеличению знаний о поведении программного модуля, связанного с соответствующим профилем запроса;

- его отказы не связаны с безопасностью.

Примечание — Отказ, некритичный для безопасности в одном контексте, может быть критичен для безопасности в другом контексте, и наоборот.

Для проверки соответствия программного модуля соответствующему критерию должны быть документально оформлены:

- точная идентификация о каждой системе и ее элементах, включая номера версий (как для программных, так и для аппаратных средств);
- идентификация пользователей и продолжительность их работы;
- продолжительность эксплуатации системы;
- процедура выбора систем, применяемых пользователями, и случаев их применения;
- процедуры обнаружения и регистрации отказов и устранения сбоев.

В.2.10.2 Оценка совокупности доказательств подтверждения соответствия

Уже существующий модуль ПО — это тот модуль, который уже существует и не был разработан специально для текущего проекта. Уже существующее ПО может быть коммерческим доступным продуктом, или оно может быть разработано какой-то организацией для предыдущего изделия или системы. Предварительно существующее ПО может или не может быть разработано в соответствии с требованиями настоящего стандарта.

Для оценки полноты безопасности новой системы, включающей уже существующие программы, необходима совокупность доказательств подтверждения соответствия для определения поведения уже существующего модуля. Она может быть получена из собственной документации модуля и описания процесса разработки модуля или может быть создана или дополнена дополнительными квалифицированными мероприятиями, выполненными разработчиком новой СБС или третьими лицами. Возможности и ограничения потенциально повторно используемого программного модуля определяются в руководстве по безопасности для применяемых изделий.

В любом случае должно существовать (или должно быть создано) руководство по безопасности для применяемых изделий, которое обеспечивает адекватную возможность выполнить оценку полноты безопасности конкретной функции безопасности, которая полностью или частично реализуется повторно используемым элементом. Если руководство отсутствует, то должен быть сделан консервативный вывод о том, что для модуля не подтверждена возможность его повторного использования в системе, связанной с безопасностью. (Это не означает, что для элемента вообще не подтверждена возможность его повторного использования, просто в данном конкретном случае не было найдено достаточно доказательств.)

Настоящий стандарт предъявляет особые требования к содержанию руководства по безопасности для применяемых изделий, см. ГОСТ 34332.3—2021 (приложение Г) и ГОСТ 34332.4—2021 (подпункты 7.4.2.12 и 7.4.2.13).

В руководстве по безопасности для применяемых изделий должно быть отражено, что:

- проект модуля известен и документирован;
- модуль подвергался проверке и подтверждению соответствия на основе систематического подхода с документально оформленной проверкой и анализом всех элементов модуля и кода модуля;
- неиспользуемые и ненужные функции модуля не помешают новой системе выполнения своих требований к безопасности;
- были выявлены все вероятные механизмы отказа модуля в новой системе и было выполнено их соответствующее смягчение.

При оценке функциональной безопасности новой системы должно быть установлено, что повторно используемый модуль применяется строго в пределах возможностей, которые для этого модуля были обоснованы доказательством и предположениями в руководстве по безопасности для применяемых изделий.

Подробное описание данного метода/средства приведено в [107]—[109].

В.2.11 Прослеживаемость

Цель — обеспечить согласованность между этапами ЖЦ Э/Э/ПЭ СБЗС системы.

Описание. Чтобы для ПО гарантировать, что результаты действий на этапах ЖЦ соответствуют требованиям корректной работы Э/ЭПЭ СБЗС системы крайне важно гарантировать обеспечение соответствия между этапами ЖЦ системы. Ключевым понятием здесь является «прослеживаемость» между действиями. Это выполнение анализа влияния, проверяющего, что решения, принятые на ранней стадии, адекватно реализованы на более поздних стадиях (прямая прослеживаемость), и что решения, принятые на более позднем этапе, действительно необходимы и санкционированы ранее принятыми решениями (обратная прослеживаемость).

Прямая прослеживаемость в основном связана с проверкой адекватности требований на более поздних этапах ЖЦ системы. Прямая прослеживаемость важна в нескольких точках ЖЦ Э/Э/ПЭ СБЗС системы следующая:

- от требований безопасности системы к требованиям безопасности ПО;
- от спецификации требований безопасности ПО системы к архитектуре ПО;
- от спецификации требований безопасности ПО к разработке ПО;
- от спецификации требований проектирования ПО к спецификации модуля и интеграционных тестов;
- от спецификации требований безопасности ПО к плану подтверждения соответствия;
- от спецификации требований безопасности ПО к плану модификации ПО (включая повторные оценку и подтверждение соответствия);
- от спецификации проекта ПО к плану верификации ПО (включая верификацию данных);
- от требований ГОСТ 34332.4—2021 (раздел 8), к плану оценки функциональной безопасности ПО.

Обратная прослеживаемость в основном связана с проверкой, насколько корректно любым требованием обосновывается каждое реализационное решение (реализация понимается в широком контексте, а не только реализация кода). Если такое обоснование отсутствует, то реализация будет содержать что-то не нужное, что приведет к увеличению сложности, но не обязательно удовлетворит любому реальному требованию к СБС. Обратная прослеживаемость важна в нескольких точках жизненного цикла системы безопасности:

- от требований безопасности к воспринимаемым потребностями безопасности;
- от архитектуры ПО к спецификации требований к ПО системы безопасности;
- от детального проекта ПО к архитектуре ПО;
- от программного кода к детальному проекту ПО;
- от плана подтверждения соответствия безопасности ПО к спецификации требований безопасности ПО;
- от плана модификации ПО к требованиям безопасности ПО;
- от плана верификации ПО (включая верификацию данных) к спецификации проекта ПО.

Подробное описание данного метода/средства приведено в [85].

В.2.12 Проектирование программного обеспечения без сохранения состояния (или с ограниченными состояниями)

Цель — ограничить сложность поведения ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2)

Описание. Рассмотрим программу, которая обрабатывает последовательность операций (переходов состояний): она получает последовательность входных данных и для каждого из них выдает выходные данные. Программа также способна запоминать некоторые или все состояния в процессе вычисления и может также запоминать свою историю «в состояниях» и учитывать это состояние при вычислении того, как реагировать на последующие входные данные.

Если выходные данные программы полностью определяются только входными данными, то считается, что такая программа работает без запоминания или является программой без сохранения состояния. Каждая операция преобразования входных/выходных данных считается полной в том смысле, что на любую операцию никак не влияет любая, более ранняя операция, и конкретные входные данные всегда приводят к одним и тем же связанным с ними выходным данным.

Если программа при вычислении входных данных учитывает, кроме входных данных, также и состояние, которое она запомнила в результате предыдущих вычислений, то такая программа обладает более сложным поведением, потому что в различных случаях она может давать различные выходные данные для одних и тех же входных данных. Результат для конкретных входных данных может зависеть от контекста (то есть от предыдущих входных и выходных данных), в котором они обрабатываются. Необходимо также отметить, что в некоторых приложениях (обычно коммуникационные системы) поведение программы может быть особенно чувствительно к изменениям в сохраненном состоянии, которые могут произойти или непреднамеренно, или злонамеренно.

Проектирование без сохранения состояния (или с ограниченными состояниями) является общим подходом, направленным на минимизацию возможной сложности поведения ПО, исключая или уменьшая использование информации о состоянии при проектировании ПО.

Подробное описание данного метода/средства приведено в [110].

В.2.13 Численный анализ в автономном режиме

Цель — гарантировать точность числовых вычислений.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.9).

Описание. Числовая погрешность может возникнуть при вычислении математической функции как следствие использования конечных представлений идеальных функций и чисел. Ошибка усечения появляется, когда функция аппроксимируется конечным числом членов бесконечного ряда, такого, как ряд Фурье. Для представления в реальном компьютере вещественных чисел с конечной точностью вводят погрешность их округления. Если выполняются какие-либо вычисления с плавающей запятой, кроме самых простейших, то должна быть проверена обоснованность вычисления, чтобы гарантировать, что точность, требуемая приложением, фактически достигнута.

Подробное описание данного метода/средства приведено в [111].

В.2.14 Диаграммы последовательности сообщений

Цель — помочь получению требований к системе на ранних стадиях проектирования ПО, включая стадии формирования требований и проектирования архитектуры ПО. В UML данный метод/средство называется «Диаграмма последовательности операций системы».

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблица Б.7 и приложение В, таблица В.17).

Описание. Диаграмма последовательности сообщений — это графический механизм для описания поведения системы с точки зрения коммуникаций между агентами системы (агентом может быть человек, компьютерная

система или элемент, либо объект программного обеспечения, в зависимости от стадии проектирования). Для каждого агента на диаграмме представлен вертикальный «жизненный путь», а стрелки между ними используют, чтобы представить сообщения. Действия по получении сообщений можно дополнительно показать на схемах в виде прямоугольников. Набор сценариев (описывающих и требуемое и нежелательное поведение) создается как спецификация необходимого поведения системы. Эти сценарии имеют несколько применений. Может быть проведена их анимация, чтобы продемонстрировать поведение системы конечным пользователям. Сценарии могут быть преобразованы в исполнимую реализацию системы. Они могут сформировать основу тестовых данных.

UML содержит расширения обычной концепции диаграммы последовательности сообщений в виде конструкций выбора и итерации, которые позволяют сценариям выполнять условные переходы и циклы, обеспечивая более компактную нотацию. Могут быть также определены подсхемы, на которые можно сослаться из нескольких диаграмм последовательностей более высокого уровня. Также могут быть представлены таймер и внешние события.

Подробное описание данного метода/средства приведено в [112], [113].

В.3 Архитектурное проектирование

В.3.1 Обнаружение и диагностика сбоев

Цель — обнаружение сбоев в Э/Э/ПЭ СБЗС системе, которые могут привести к отказам, и тем самым обеспечить основу для контрмер, направленных на минимизацию числа последующих сбоев.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. Обнаружение сбоев представляет собой действие по проверке системы на наличие ошибочных состояний (обусловленных сбоями в проверяемой системе). Основная цель обнаружения сбоев состоит в том, чтобы предотвратить появление неверных результатов. Система, действующая в сочетании с параллельно работающими компонентами, останавливающими управление, в случае, если она обнаруживает, что ее собственные результаты некорректны, называется самопроверяемой.

Обнаружение сбоев основывается на принципах избыточности [в основном при обнаружении сбоев АС (см. ГОСТ 34332.3—2021, приложение А)] и разнообразия (программные ошибки). Необходим один из способов голосования для определения корректности результатов. Применимы специальные методы, к которым относятся программирование утверждений, программирование *N*-версий и различные методы контроля. Для АС — введение дополнительных сенсоров; контуров регулирования; кодов, проверяющих ошибки, и др.

Обнаружение сбоев может обеспечиваться проверками в области значений или временной области на различных уровнях, особенно на физическом уровне (температура, напряжение и т. п.), логическом (коды, обнаруживающие ошибки), функциональном (утверждения) или внешнем (проверки достоверности). Результаты этих проверок могут быть сохранены и связаны с данными, на которые повлиял сбой, с тем чтобы обеспечить возможность отслеживания отказов.

Сложные системы состоят из подсистем. Эффективность обнаружения ошибок, диагностики и компенсации ошибок зависит от сложности взаимодействия между подсистемами, влияющими на распространение ошибок.

Диагностику ошибок следует применять на уровне самых малых подсистем, поскольку подсистемы меньших размеров допускают более детальную диагностику ошибок (обнаружение ошибочных состояний).

Интегрированные информационные системы уровня предприятия могут обычным способом передавать состояния безопасности системы, в том числе информацию диагностического тестирования, другим управляющим системам. При обнаружении некорректного поведения оно может быть выделено и использовано для запуска корректирующих действий до возникновения опасной ситуации. При появлении инцидента документирование такого некорректного поведения может способствовать его последующему анализу.

В.3.2 Коды обнаружения и исправления ошибок

Цель — обнаружение и исправление ошибок в чувствительной к ним информации.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. Для информации, состоящей из n битов, генерируется закодированный блок из k битов, который позволяет обнаруживать и исправлять r ошибок. Примерами могут служить код Хэмминга и полиномиальные коды.

Следует отметить, что в СБС лучше уничтожить ошибочные данные, чем пытаться исправлять их, поскольку лишь заранее определенная часть ошибок может быть исправлена.

Подробное описание данного метода/средства приведено в [114].

В.3.3 Программирование с проверкой ошибок

Цель — обнаружение ошибок, оставшихся при проектировании ПО, в процессе выполнения программ в целях предотвращения критичных для безопасности отказов систем, и продолжение правильного выполнения программы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.3—2021 (приложение А, таблица А.17) и в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. В методе программирования с проверкой ошибок уже заложена идея проверки предусловий (до выполнения последовательности операторов начальные условия проверяют на соответствие) и постусловий (проверяют результаты после выполнения последовательности операторов). Если предусловия или постусловия не соблюдаются, то выдается сообщение об ошибке.

Пример —
assert < pre-condition>;
action 1;
.....
action x;
assert < post-condition>;

Подробное описание данного метода/средства приведено в [115].

В.3.4 Методы контроля

Цель — защита от ошибок в ПО, не обнаруженных на этапах разработки спецификации и реализации, которые неблагоприятно влияют на безопасность.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. Различают два подхода реализации контроля:

- процесс контроля и контролируемая функция реализованы на одном компьютере с некоторой гарантией независимости между ними; и
- процесс контроля и контролируемая функция реализованы на разных компьютерах.

Метод, в котором процесс контроля и контролируемая функция реализованы на разных компьютерах (имеющих разную спецификацию), называется методом внешнего контроля. Данный метод направлен только на то, чтобы гарантировать, что основным компьютером выполняются безопасные, но не обязательно корректирующие действия. Метод внешнего контроля обеспечивает непрерывный контроль основного компьютера и предотвращает вхождение системы в опасное состояние. Кроме того, если обнаружится, что основной компьютер вошел в потенциально опасное состояние, система должна возвратиться обратно в безопасное состояние с помощью либо средств внешнего контроля, либо основного компьютера.

АС и ПО средств внешнего контроля следует классифицировать и квалифицировать в соответствии с подходящим УПБ.

В.3.5 Многовариантное программирование

Цель — обнаружение и наложение маски при выполнении программ на не выявленные на этапах проектирования и реализации ошибки программных средств для предотвращения критичных для безопасности отказов системы и продолжения ее правильной работы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. При многовариантном программировании заданную программную спецификацию проектируют и реализуют различными способами *N* раз. Одни и те же входные значения поступают в *N* версий и сравниваются результаты, выданные *N* версиями. Если результат определяется как правильный, он поступает на выходы компьютера.

Важным требованием является то, что в некотором смысле *N* версий независимы друг от друга, поэтому они не все одновременно перестают правильно работать по общей причине. Независимость версий, являющуюся основой для многовариантного программирования, на практике довольно трудно достичь и продемонстрировать.

N версий могут выполняться параллельно на различных компьютерах, либо все версии могут выполняться на одном компьютере с последующим сравнением полученных результатов на том же компьютере. Для этих *N* результатов могут быть использованы различные стратегии сравнения, и в зависимости от заданных требований применяются следующие стратегии:

- если система находится в безопасном состоянии, можно потребовать полного согласия (все *N* результатов одинаковы); в противном случае используется выходное значение, которое заставит систему перейти в безопасное состояние. Для простых пошаговых систем сравнение может обеспечить безопасность. В этом случае безопасное действие может быть разбито по шагам, если какая-либо версия реализует пошаговые операции. Этот подход обычно используют только для двух версий ($N = 2$);
- для систем, находящихся в опасном состоянии, могут быть реализованы стратегии мажоритарного сравнения. В случаях, если отсутствует общее согласие, могут быть использованы вероятностные подходы с тем, чтобы максимизировать вероятность выбора правильного значения, например, принять среднее значение, временно зафиксировать выходы, пока не будет достигнуто согласие и т. п.

Данный метод не устраняет ошибок, не выявленных при проектировании программ, а также ошибок в интерпретации спецификации, однако он является средством для обнаружения и маскирования ошибок, прежде чем они смогут повлиять на безопасность.

Подробное описание данного метода/средства приведено в [116]—[119].

В.3.6 Восстановление предыдущего состояния

Цель — обеспечение исправления функциональных операций при наличии одной или нескольких ошибок.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. При обнаружении ошибки система возвращается в первоначальное внутреннее состояние, правильность которого была подтверждена ранее. Данный метод предполагает частое сохранение внутреннего состояния в так называемых четко определенных контрольных точках. Сохранение может быть выполнено глобально (для всей базы данных) или частично (для изменений только между контрольными точками). После этого система должна устранить изменения, произошедшие за это время путем занесения в журнал (аудиторское отслеживание действий), компенсации (все результаты этих изменений аннулируются) или внешнего (ручного) способа.

Подробное описание данного метода/средства приведено в [120], [121].

В.3.7 Механизмы повторных попыток парирования сбоя

Цель — парирование обнаруженного сбоя с помощью механизмов повторных попыток.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. В случае обнаружения сбоя или ошибочного условия предпринимают попытки парирования сбоя или восстановления ситуации путем повторного выполнения того же кода. Восстановление с помощью повторной попытки может быть полным в виде перезагрузки и повторного пуска процедуры, либо небольшим в виде перепланирования и повторного пуска задачи после выполнения блокировки по времени программы или управляющего действия задачи. Методы повторной попытки широко используют при коммуникационных сбоях или при восстановлении от ошибок, и условия повторной попытки могут быть отделены флажками от ошибки протокола связи (контрольная сумма и т.д.) или от подтверждающего ответа блокировки по времени коммуникации.

Подробное описание данного метода/средства приведено в [122].

В.3.8 Постепенное отключение функций

Цель — обеспечение пригодности наиболее критичных системных функций, несмотря на отказы, путем игнорирования наименее критичных функций.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. Данный метод устанавливает приоритеты для различных функций, выполняемых системой. Проект создаваемой Э/Э/ПЭ СБЗС системы гарантирует, что в случае недостаточности ресурсов для выполнения всех системных функций функции высшего приоритета будут выполнены в предпочтение функциям более низкого приоритета. Например, функции регистрации ошибки и события могут оказаться задачей более низкого приоритета, чем системные функции управления, и в этом случае управление системой будет продолжаться, даже если АС из-за регистрации ошибки окажутся неработоспособными. Более того, если АС управления системой окажутся неработоспособными, а АС регистрации ошибок останутся работоспособными, то АС регистрации ошибок возьмут на себя функцию управления.

Данные соображения относятся в основном к АС, но они применимы также и к системе в целом, включая ПО. Данные соображения должны учитываться, начиная с самых ранних этапов проектирования.

Подробное описание данного метода/средства приведено в [123], [124].

В.3.9 Исправление ошибок методами искусственного интеллекта

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Цель — реализовать способность гибко реагировать на возможные угрозы безопасности с использованием сочетания методов и моделей процессов, а также некоторые способы безопасности в режиме онлайн и анализа надежности.

Описание. Для различных каналов связи Э/Э/ПЭ СБЗС системы прогнозирование (вычисление тенденций), исправление ошибок, обслуживание и контролирующие действия могут достаточно эффективно поддерживаться системами, основанными на методах искусственного интеллекта. Правила для таких систем могут быть созданы непосредственно из спецификаций и проверены на соответствие. С помощью методов искусственного интеллекта некоторые ошибки общего характера, попадающие в спецификации, для устранения которых уже существуют некоторые правила проектирования и реализации, могут быть исключены, особенно при представлении комбинаций моделей и методов функциональным или описательным способом.

Методы выбирают так, чтобы ошибки могли быть устранены и влияние отказов минимизировано для обеспечения требуемой полноты безопасности.

Примечание — Предупреждение об исправлении ошибочных данных (см. В.3.2) и об отрицательных рекомендациях применения данного метода [см. ГОСТ 34332.4—2021 (таблица А.2, пункт 5)].

Подробное описание данного метода/средства приведено в [125]—[127].

В.3.10 Динамическая реконфигурация

Цель — обеспечение функциональности Э/Э/ПЭ СБЗС системы, несмотря на внутренний отказ.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. Логическую архитектуру Э/Э/ПЭ СБЗС системы выбирают такой, чтобы ее можно было реализовать на подмножестве доступных средств системы. Логическая архитектура должна быть способна к обнаружению отказа в физических средствах и дальнейшей повторной реализации логической архитектуры на другом подмножестве доступных средств, остающихся функционирующими. Несмотря на то, что данный метод, в основном, традиционно ограничен только восстановлением отказавших модулей АС, он применим также к ошибкам в ПО при наличии достаточной «избыточности времени прогона» для повторного выполнения программы или при наличии достаточных избыточных данных, которые обеспечат незначительное влияние отдельного и изолированного отказа.

Данный метод следует учитывать на первом этапе проектирования системы.

Подробное описание данного метода/средства приведено в [128].

В.3.11 Безопасность и работа в жестком реальном времени. Архитектура с временным распределением

Цель — обеспечение компонуемости и простой реализации обеспечения отказоустойчивости в критических к безопасности системах, действующих в условиях жесткого реального времени.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. В архитектуре системы с временным распределением (TTA) все действия системы инициируются и выполняются под управлением глобальной (жесткой) системы синхронизации. Каждому приложению присвоен фиксированный временной слот на шине с временным распределением, во время которого происходит обмен сообщениями с другими приложениями, поэтому каждое приложение может выполнять обмен только согласно жестко определенному расписанию обмена. В управляемых событиями системах системные действия инициируются случайными событиями в непредсказуемые моменты времени. Главные преимущества архитектуры с временным распределением следующие:

- способность значительно уменьшать усилия, требуемые для тестирования и спецификации системы;
- простая реализация обеспечения отказоустойчивости, которая делает архитектуру очень востребованной для критических к безопасности приложений;
- применение глобально синхронизируемого времени облегчает проектирование распределенных систем реального времени.

Передача между узлами выполняется в соответствии с протоколом временного распределения ресурсов (TTP/C) согласно статическому расписанию обмена, в рамках которого решается, когда передать сообщение и является ли полученное сообщение важным для конкретного электронного модуля. Доступ к шине реализуется по схеме с определенным периодическим расписанием в режиме множественного доступа с разделением времени (TDMA), связанной с глобальным временем.

Протокол TTP/C гарантирует четыре базовые услуги (базовые службы) в сети узлов TTA системы:

- детерминированная передача сообщений в строго определенные моменты времени. Передача сообщений от выходного порта передающего элемента к входным портам получающих элементов в пределах априорно известного временного интервала. Отказоустойчивая транспортная служба, предлагаемая коммуникационной услугой с временным распределением, которая доступна через временной интерфейс с сетевым экраном, устраняет передачу ошибок управления проектом и минимизирует связь между элементами. Передача сообщений в строго определенные моменты времени, с минимальной задержкой и с минимальными флуктуациями, крайне важна для достижения устойчивости управления в приложениях реального времени;
- отказоустойчивая синхронизация. Коммуникационный контроллер (а не центральный сервер) генерирует отказоустойчивый синхронизирующий глобальный временной сигнал (с точностью до нескольких тактов), которым обеспечены подсистемы узлов;
- контроль целостности данных при сбоях в узлах (служба целостности). Коммуникационный контроллер сообщает каждому «минимальному элементу замены» (SRU) о состоянии остальных SRU в кластере с временной задержкой меньше одного раунда TDMA;
- строгая изоляция сбоя. Злонамеренно дефектная подсистема узла (включая ее программное обеспечение) может сформировать ошибочные выходные данные, но никогда не сможет вмешаться каким-либо способом в корректную работу остальной части кластера TTP/C. Невоспринимаемость сбоя во времени гарантируется поведением коммуникационного контроллера, реализующего временное распределение.

Примечание — Существуют другие протоколы с временным распределением: FlexRay и TT-Ethernet (Интернет с временным распределением).

Подробное описание данного метода/средства приведено в [129]—[131].

В.3.12 Универсальный язык программирования (UML)

Цель — обеспечение исчерпывающего набора нотаций для моделирования требуемого поведения сложных систем.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.7).

Описание. Универсальный язык программирования (UML) — это набор требований и нотаций проектирования, которые предназначены обеспечить всестороннюю поддержку процессу разработки ПО. Некоторые части UML основаны на нотациях, впервые появившихся в других методах (таких как диаграмма последовательностей и диаграммы переходов), а другие нотации уникальны в UML. UML очень близок к объектно-ориентированному языку, хотя некоторые из нотаций могут не использоваться в объектно-ориентированном программировании. UML поддерживается многими коммерчески доступными инструментами CASE, многие из которых способны автоматически генерировать программные коды из моделей UML.

Нотации UML, которые обычно применяют для составления спецификации и проектирования СБС, следующие:

- диаграммы классов;
- прецеденты;
- диаграммы действий;
- диаграммы переходов (диаграммы состояний);
- диаграммы последовательностей.

Другие нотации UML относятся к представлению проекта архитектуры ПО (структуры ПО), но в данном пункте не рассматриваются.

Диаграмма переходов описана в Б.2.3.2, а диаграмма последовательностей — в В.2.14. Остальные нотации описаны в следующих трех пунктах.

В.3.12.1 Диаграммы классов

Диаграммы классов определяют классы объектов, которые должны быть использованы в ПО. Они основаны на более ранних схемах атрибут — связь — сущность, но адаптированы к объектно-ориентированному проектированию. Каждый класс (из которого один или более экземпляров будут использоваться в качестве объектов во время выполнения) представлен прямоугольником, а различные отношения между классами показаны линиями или стрелками. К схеме могут быть добавлены операции или методы, предлагаемые каждым классом, и атрибуты данных каждого класса. Представляемые отношения состоят как из ссылочных отношений с указанием их кратности (экземпляр класса А может обратиться к одному или нескольким экземплярам класса В), так и из отношений специализации (класс X — уточнение класса Y) возможно, с дополнительными методами и атрибутами. Может быть изображено множественное наследование.

В.3.12.2 Прецеденты

Прецеденты обеспечивают текстовое описание требуемого поведения системы в соответствии с определенным сценарием, обычно с точки зрения внешних агентов, включая пользователей системы и внешних систем. Для представления дополнительного поведения, особенно в случаях ошибочных ответов, внутри данного прецедента могут быть использованы альтернативные подсценарии. Чтобы обеспечить достаточно полную спецификацию системных требований, разрабатывают набор прецедентов. Прецеденты могут быть начальной точкой для разработки более строгих моделей, таких как диаграммы последовательностей и диаграммы действий.

Диаграммы прецедентов обеспечивают пиктографическое представление системы и агентов, которые включены в прецеденты, но оно не строгое, так как для спецификации важным является только текстовое описание прецедента.

В.3.12.3 Диаграммы действий

Диаграммы действий показывают намеченную последовательность действий, выполняемых элементом ПО (часто объектом в объектно-ориентированном проекте), включая последовательное и итеративное поведение (некоторые аспекты очень похожи на блок-схему). Диаграммы действий позволяют описать параллельные действия для нескольких элементов с указанием взаимодействий между этими элементами стрелками на диаграмме. Точки синхронизации, в которых действие до начала его выполнения должно ожидать один или несколько входных потоков от других действий, обозначают символом, подобным узлу сети Петри.

Подробное описание данного метода/средства приведено в [113].

В.4 Инструменты разработки и языки программирования

В.4.1 Строго типизированные языки программирования

Цель — снижение вероятности ошибок путем использования языка, который обеспечивает высокий уровень проверки компилятором.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.3).

Описание. Если скомпилирован строго типизированный язык программирования, то проводится много проверок по использованию типов переменных, например, в вызовах процедур и доступе к внешним данным. Компи-

ляция может оказаться безуспешной, и будет выдано сообщение об ошибке при любом использовании типа переменных, которое не соответствует заранее установленным правилам.

Подобные языки обычно позволяют определять установленные пользователем типы данных на основе типов данных базового языка (например, целое число, вещественное число). Затем эти типы могут быть использованы так же, как и базовый тип. Вводят строгие проверки, чтобы гарантировать использование правильного типа. Эти проверки проводят для всей программы, даже если она построена из отдельных скомпилированных модулей. Данные проверки гарантируют также, что число и тип аргументов конкретной процедуры соответствуют числу и типу аргументов в ее вызове, даже если к ней обращаются из отдельно скомпилированных программных модулей.

Строго типизированные языки обычно обеспечивают другие аспекты проверенной на практике техники ПО, например, легко анализируемые структуры управления (if... then... else..., do... while... и т. п.), которые приводят к четко структурированным программам.

Типичными примерами строго типизированных языков являются PASCAL, Ada и MODULA 2.

Подробное описание данного метода/средства приведено в [104].

В.4.2 Подмножество языка

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.3).

Цель — снижение вероятности внесения программных ошибок и повышение вероятности обнаружения оставшихся ошибок.

Описание. Язык исследуют для определения программных конструкций, подверженных ошибкам либо сложных для анализа, например, при использовании методов статического анализа. После этого определяют языковое подмножество, которое исключает такие конструкции.

В.4.3 Сертифицированные средства и сертифицированные трансляторы

Цель — оказание помощи разработчику на различных этапах разработки ПО в использовании необходимых инструментальных средств, которые, где это возможно, должны быть сертифицированы с тем, чтобы обеспечить конкретную степень уверенности в корректности результатов.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.3).

Описание. Сертификацию инструментальных средств в общем случае допускается проводить независимо в национальных органах по сертификации по независимому набору критериев, находящемуся, как правило, в межгосударственных или международных стандартах. В идеальном случае инструментальные средства, применяемые на всех стадиях разработки (разработка спецификации, проектирование, кодирование, тестирование и оценка соответствия), и те из них, которые используют в управлении конфигурацией, должны быть сертифицированы.

В настоящее время регулярным процедурам сертификации подвергаются только компиляторы (трансляторы); сертификацию проводят национальными органами по сертификации, которая заключается в проверке компиляторов (трансляторов) на соответствие международным стандартам, например, для языков Ada или PASCAL.

Важно отметить, что сертифицированные инструментальные средства и сертифицированные трансляторы обычно сертифицируют только на соответствие стандартам на соответствующий язык или процесс. Обычно их никак не сертифицируют на соответствие стандартам по безопасности.

Подробное описание данного метода/средства приведено в [132].

В.4.4 Инструментальные средства и трансляторы — повышение уверенности на основании опыта использования

Цель — исключение любых проблем, обусловленных ошибками транслятора, которые могут появиться во время разработки, верификации и эксплуатации программного пакета.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.3).

Описание. Транслятор используют в тех случаях, где не было доказательств ненадлежащего исполнения многих предыдущих проектов. Если отсутствует опыт эксплуатации трансляторов или в них обнаружены любые известные серьезные ошибки, то от таких трансляторов следует отказаться, если только нет каких-либо других гарантий корректной работы транслятора (см. В.4.4.1).

Если в трансляторе выявлены небольшие недостатки, то соответствующие языковые конструкции в проектах, связанных с безопасностью, не применяют.

Другим вариантом исключения проблем, обусловленных ошибками транслятора, является ограничение языка до его общеиспользуемых конструкций.

Недоработанные трансляторы служат серьезным препятствием в любой разработке программ. Такие трансляторы в общем случае делают невозможной разработку СБ ПО.

В настоящее время отсутствуют методы подтверждения корректности всего транслятора или отдельных его частей.

В.4.4.1 Сравнение исходных программ и исполнимых кодов

Цель — убедиться в том, что инструменты, используемые для создания образа PROM, не вносят в него никаких ошибок.

Описание. Образ PROM обратно преобразуют в совокупность «объектных» модулей. Эти «объектные» модули преобразуют обратно в файлы ассемблера, которые затем, с помощью соответствующих методов, сравнивают с фактическими исходными файлами, используемыми первоначально для разработки PROM.

Основное преимущество данного метода состоит в том, что инструменты [компиляторы, редакторы связей (компоновщики) и т. п.], используемые для разработки образа PROM, не требуют подтверждения соответствия. Этим методом проверяют правильность преобразования исходного файла, используемого для конкретной СБС.

Подробное описание данного метода/средства приведено в [133].

В.4.5 Выбор подходящего языка программирования

Цель — обеспечение в максимальной степени требований настоящего стандарта для специального защищающего программирования, строгой типизации, структурного программирования и, возможно, суждений. Выбор подходящего языка программирования обеспечивает легко верифицируемый код и простые процедуры разработки, верификации и эксплуатации программ.

Примечание— Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.3).

Описание. Язык программирования должен быть полностью и однозначно определен. Язык должен быть ориентирован на пользователя или проблему, а не на процессор или платформу. Широко используемые языки программирования или их подмножества предпочтительнее языков специального применения.

Подходящий язык программирования также должен обеспечивать:

- блочную структуру организации программ;
- проверку времени трансляции;
- проверку во время работы программы типов и границ массивов.

Подходящий язык программирования должен включать в себя:

- использование небольших и управляемых программных модулей;
- ограничение доступа к данным в конкретных программных модулях;
- определение поддиапазонов переменных;
- любые другие типы конструкции, ограничивающие ошибки.

Если операции СБС зависят от ограничений реального времени, то язык программирования должен обеспечивать также обработку исключений или прерываний.

Желательно, чтобы язык программирования обеспечивался соответствующим транслятором, подходящими библиотеками с заранее созданными программными модулями, отладчиком и инструментами как для управления версиями, так и для разработки.

В настоящее время еще не ясно, будут ли объектно-ориентированные языки программирования предпочтительнее других общепринятых языков.

К свойствам, которые усложняют верификацию и поэтому должны быть исключены, относятся:

- безусловные переходы (за исключением вызовов подпрограмм);
- рекурсии;
- указатели, динамически распределяемые области памяти или любые типы динамических переменных или объектов;
- обработка прерываний на уровне исходного кода;
- множество входов или выходов в циклах, блоках или подпрограммах;
- неявная инициализация или объявление переменных;
- вариантные записи и эквивалентность;
- процедурная переменная в качестве параметра.

Языки программирования низкого уровня, в частности ассемблеры, обладают недостатками, связанными с их жесткой ориентацией на АС (процессор или определенную платформу).

Желательным свойством языка программирования состоит в том, чтобы его проектирование и использование приводило к созданию программ, выполнение которых предсказуемо. Если используется подходящий конкретный язык программирования, то в нем должно существовать подмножество, которое гарантирует, что выполнение программы предсказуемо. Это подмножество не может быть (в общем случае) статически определено, несмотря на то, что многие статические ограничения помогают гарантировать предсказуемое выполнение. Обычно может потребоваться демонстрация того, что индексы массива находятся в установленных пределах и что числовое переполнение не может возникнуть и т. п.

Рекомендации по конкретным языкам программирования приведены в таблице В.2.

Таблица В.2 — Рекомендации по конкретным языкам программирования

Язык программирования	УПБ 1	УПБ 2	УПБ 3	УПБ 4
1 Ada	ОР	ОР	Р	Р
2 Ada с подмножеством	ОР	ОР	ОР	ОР
3 Java	НР	НР	НР	НР
4 Java с подмножеством (без включения сборки мусора или с включением сборки мусора, которая не будет вызывать остановку прикладной программы в течение значительного промежутка времени). Руководящие указания по использованию объектно-ориентированных средств см. в приложении Ж	Р	Р	НР	НР
5 PASCAL (см. примечание 1)	ОР	ОР	Р	Р
6 PASCAL с подмножеством	ОР	ОР	ОР	ОР
7 FORTRAN 77	Р	Р	Р	Р
8 FORTRAN 77 с подмножеством	ОР	ОР	ОР	ОР
9 C	Р	- -	НР	НР
10 C с подмножеством и стандартом кодирования, а также использование инструментов статического анализа	ОР	ОР	ОР	ОР
11 C++ (Руководящие указания по использованию объектно-ориентированных средств см. в приложении Ж)	Р	- -	НР	НР
12 C++ с подмножеством и стандартом кодирования, а также использование инструментов статического анализа (Руководящие указания по использованию объектно-ориентированных средств см. в приложении Ж)	ОР	ОР	ОР	ОР
13 Ассемблер	Р	Р	- -	- -
14 Ассемблер с подмножеством и стандартом кодирования	Р	Р	Р	Р
15 Многоступенчатые диаграммы	Р	Р	Р	Р
16 Многоступенчатая диаграмма с определенным подмножеством языка	ОР	ОР	ОР	ОР
17 Диаграмма функциональных блоков	Р	Р	Р	Р
18 Диаграмма функциональных блоков с определенным подмножеством языка	ОР	ОР	ОР	ОР
19 Структурированный текст	Р	Р	Р	Р
20 Структурированный текст с определенным подмножеством языка	ОР	ОР	ОР	ОР
21 Последовательная функциональная диаграмма	Р	Р	Р	Р
22 Последовательная функциональная диаграмма с определенным подмножеством языка	ОР	ОР	ОР	ОР
23 Список команд	Р	- -	НР	НР
24 Список команд с определенным подмножеством языка	ОР	Р	Р	Р

Примечания

1 Пояснения к рекомендациям: Р — рекомендованный, ОР — особо рекомендованный, НР — нерекомендованный, «- -» — рекомендация отсутствует (см. ГОСТ 34332.4—2021, приложение А).

2 Системное программное обеспечение включает в себя операционную систему, драйверы, встроенные функции и программные модули, являющиеся частью системы. Программные средства обычно обеспечивают системой безопасности при поставке. Подмножество языка следует выбирать очень внимательно с тем, чтобы исключить сложные структуры, которые могут образоваться в результате ошибок реализации. Следует выполнять проверки для того, чтобы убедиться в правильном использовании подмножества языка программирования.

Окончание таблицы В.2

3 Прикладная программа представляет собой программу, разработанную для конкретного безопасного применения. Во многих случаях такая программа разрабатывается конечным пользователем либо подрядчиком, ориентированным на разработку прикладных программ. В тех случаях, когда ряд языков программирования поддерживает одни и те же рекомендации, разработчику следует выбрать тот, который повсеместно используется персоналом в конкретной промышленности или отрасли. Подмножество языка программирования следует выбирать с особым вниманием, чтобы исключить сложные структуры, которые могут привести к ошибкам реализации.

4 Если конкретный язык программирования не представлен в настоящей таблице, то это не означает, что он исключен. Такой конкретный язык программирования должен быть проверен на соответствие требованиям настоящего стандарта.

5 Существует ряд расширений языка PASCAL, включая свободно распространяемый PASCAL. Ссылки на PASCAL включают эти расширения.

6 Java имеет сборщик мусора времени выполнения. Подмножество Java может не иметь сборщика мусора. Некоторые реализации Java обеспечивают прогрессивную сборку мусора, которая восстанавливает свободную память в процессе выполнения программы и предотвращает выполнение остановки, когда исчерпана доступная память. Приложения жесткого реального времени не должны использовать любые средства сборки мусора.

7 Если применение языка Java требует использования интерпретатора времени выполнения для промежуточного кода Java, то такой интерпретатор следует рассматривать, как часть ПО, связанного с безопасностью, и с учетом требований ГОСТ 34332.4.

8 Информационные сведения о пунктах 15—24 см. в [43].

Подробное описание данных методов/средств приведено в [40], [49], [104], [134]—[142].

В.4.6 Автоматическая генерация программного обеспечения

Цель — автоматизация решения задач реализации ПО, наиболее подверженных ошибкам.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.2).

Описание. Проект системы описывают моделью (исполнимой спецификацией) на более высоком уровне абстракции, чем традиционный исполняемый код. Модель автоматически преобразуется генератором кода в исполнимую форму. Цель состоит в том, чтобы улучшить качество ПО посредством устранения подверженных ошибкам ручных задач кодирования. Дальнейшая возможная выгода состоит в том, что более сложные проекты могут быть выполнены на более высоком абстрактном уровне.

Подробное описание данного метода/средства приведено в [143], [144].

В.4.7 Управление тестированием и средства автоматизации

Цель — обеспечение систематического и всестороннего подхода к тестированию Э/Э/ПЭ СБЗС системы и ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.5).

Описание. Использование подходящих средств поддержки позволяет автоматизировать более трудоемкие и подверженные ошибкам задачи при разработке Э/Э/ПЭ СБЗС системы и дает возможность применения систематического подхода к управлению тестированием. Доступность поддержки обеспечивает более всесторонний подход и к обычному и регрессионному тестированию.

Подробное описание данного метода/средства приведено в [145].

В.5 Верификация и модификация

В.5.1 Вероятностное тестирование

Цель — получение количественных показателей надежности исследуемой программы.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблицы А.5, А.7 и приложение В, таблицы В.15, В.17).

Описание. Применение данного метода обеспечивает оценку статистической надежности ПО. Количественные показатели могут быть получены с учетом относительных уровней доверия и значимости и предоставлять:

- вероятность ошибки при запросе;
- вероятность ошибки в течение определенного периода времени;
- вероятность последствий ошибки.

Из этих показателей могут быть получены другие показатели, например:

- вероятность безошибочной работы;

- вероятность живучести;
- доступность;
- MTBF или частота отказов;
- вероятность безопасного исполнения.

Вероятностные суждения основывают либо на результатах статистических испытаний, либо на опыте эксплуатации. Обычно количество тестовых примеров или наблюдаемых практических примеров очень велико, и тестирование в режиме запросов занимает значительно меньше времени, чем в непрерывном режиме работы.

Для формирования входных данных тестирования и управления выходными данными тестирования обычно используют инструменты автоматического тестирования. Обширные тесты прогоняют на больших центральных компьютерах с имитацией соответствующей периферии. Тестируемые данные выбирают с учетом как систематических, так и случайных ошибок АС. Например, общее управление тестированием гарантирует профиль тестируемых данных, тогда как случайный выбор тестируемых данных может управлять отдельными тестовыми примерами более детально.

Как указано выше, индивидуальные средства для тестирования, выполнение тестирования и управление тестированием определяются детализированными целями тестирования. Другие важные условия задают математическими предпосылками, которые должны быть соблюдены, если оценка тестирования удовлетворяет заданным целям тестирования.

Из опыта эксплуатации также могут быть получены вероятностные характеристики поведения любого тестируемого объекта. Если соблюдаются одинаковые условия, то к оценкам результатов тестирования может быть применен одинаковый математический аппарат.

При использовании этих методов достаточно сложно продемонстрировать на практике сверхвысокие уровни надежности.

Подробное описание данного метода/средства приведено в [146]—[149].

В.5.2 Регистрация и анализ данных

Цель — документирование всех данных, решений и обоснований программных проектов для упрощения верификации, подтверждения соответствия, оценки и поддержки ПО.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблицы А.5 и А.8).

Описание. В процессе всего проектирования разрабатывают подробную документацию, в состав которой входят:

- результаты тестирования, выполняемого на каждом программном модуле;
- решения и их разумные обоснования;
- описание проблем и их решений.

В процессе и по завершении проекта эта документация может быть проанализирована на наличие широкого набора информации. В частности, такая информация, использовавшаяся в качестве обоснования при принятии конкретных решений в процессе разработки проекта и очень важная для обслуживания вычислительных систем, не всегда известна инженерам по эксплуатации.

Подробное описание данного метода/средства приведено в [150].

В.5.3 Тестирование интерфейса

Цель — обнаружение ошибок в интерфейсах подпрограмм.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.5).

Описание. Возможны несколько уровней детализации или полноты тестирования. К наиболее важным уровням относится тестирование:

- всех интерфейсных переменных с их предельными значениями;
- всех отдельных интерфейсных переменных с их предельными значениями с другими интерфейсными переменными с их нормальными значениями;
- всех значений предметной области каждой интерфейсной переменной с другими интерфейсными переменными с их нормальными значениями;
- всех значений всех переменных в разных комбинациях (возможно только для небольших интерфейсов);
- заданных условий испытаний, относящихся к каждому вызову каждой подпрограммы.

Эти тестирования особенно важны, если интерфейсы не имеют возможности обнаруживать неправильные значения параметров. Такие тестирования также важны при генерации новых конфигураций ранее существовавших подпрограмм.

В.5.4 Анализ граничных значений

Цель — обнаружение программных ошибок при предельных и граничных значениях параметров.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.2, Б.3 и Б.8).

Описание. Предметную входную область программы разделяют на множество входных классов в соответствии с отношениями эквивалентности (см. В.5.7). Тестирование должно охватывать границы и экстремальные значения классов. Данным тестированием проверяют совпадение границы предметной входной области в спецификации с границами, установленными программой. Использование нулевого значения как в непосредственных, так и в косвенных преобразованиях часто приводит к ошибкам. Особого внимания требуют:

- нулевой делитель;
- знаки пробела *ASC*II;
- пустой стек или элемент списка;
- заполненная матрица;
- ввод нулевой таблицы.

Обычно границы входных значений напрямую соотносят с границами выходных значений. Для установления выходных параметров в их предельные значения необходимо записывать специальные тестовые примеры. Следует также, по возможности, рассмотреть спецификацию такого тестового примера, который побуждает выходное значение превысить установленные спецификацией граничные значения.

Если выходные значения являются последовательностью данных (например, таблица), то особое внимание следует уделить первому и последнему элементам, а также спискам, содержащим либо ни одного, либо один, либо два элемента.

Подробное описание данного метода/средства приведено в [58].

В.5.5 Предположение ошибок

Цель — исключение ошибок программирования.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.2 и Б.8).

Описание. Опыт тестирования и интуиция в сочетании со сведениями и заинтересованностью относительно тестируемой системы могут побудить разработчика к добавлению некоторых неклассифицированных тестовых примеров к набору заданных тестовых примеров.

Специальные значения или комбинации значений могут быть подвержены ошибкам. Некоторые, вызывающие интерес тестовые примеры, могут быть получены из анализа контрольных списков. Следует также рассмотреть, является ли система достаточно устойчивой. Например, следует ли нажимать клавиши на передней панели слишком быстро или слишком часто. Что произойдет, если две клавиши нажать одновременно.

Подробное описание данного метода/средства приведено в [58].

В.5.6 Введение ошибок

Цель — подтверждение адекватности набора тестовых примеров.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.2 и Б.4).

Описание. В программу вводят (подсеивают) некоторое число известных типов ошибок, и программа выполняется с тестовыми примерами в режиме тестирования. При обнаружении только некоторых введенных ошибок тестовый пример становится неадекватным. Отношение числа найденных введенных ошибок к общему числу введенных ошибок оценивают как отношение числа найденных реальных ошибок к общему числу реальных ошибок. Это дает возможность оценить количество остаточных ошибок и тем самым остальную работу по тестированию.

$$\frac{\text{Обнаруженные введенные ошибки}}{\text{Общее число введенных ошибок}} = \frac{\text{Обнаруженные реальные ошибки}}{\text{Общее число реальных ошибок}}$$

Обнаружение всех введенных ошибок может указывать либо на адекватность тестового примера, либо на то, что введенные ошибки было слишком легко найти. Ограничениями данного метода являются: порядок получения любых полезных результатов, типы ошибок. Также необходимо, чтобы позиции введенных ошибок отражали статистическое распределение реальных ошибок.

Подробное описание данного метода/средства приведено в [151].

В.5.7 Разделение входных данных на классы эквивалентности

Цель — адекватное тестирование программных средств с использованием минимума тестируемых данных. Тестируемые данные получают путем выбора частей входных данных предметной области, необходимых для анализа ПО.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.2 и Б.3).

Описание. В основу данного метода тестирования положено соотношение эквивалентности входных данных, определяющее разбиение входных данных предметной области.

Тестовые примеры выбирают в целях охвата всех предварительно специфицированных разбиений. Из каждого класса эквивалентности выбирают по меньшей мере один тестовый пример.

Существуют следующие основные возможности разбиения входных данных:

- классы эквивалентности, образованные из спецификации, — интерпретация спецификации может быть ориентирована либо на входные значения (например, выбранные значения считаются одинаковыми), либо ориентирована на выходные значения (например, набор значений приводит к одному и тому же функциональному результату);

- классы эквивалентности, образованные в соответствии с внутренней структурой программы, — результаты класса эквивалентности определяют из статического анализа программ, например, набор значений обрабатывают одним и тем же способом.

Подробное описание данного метода/средства приведено в [58]—[60].

В.5.8 Структурное тестирование

Цель — применение тестов, анализирующих определенные подмножества структуры программы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.2).

Описание. На основе анализа программы выбирают набор входных данных так, чтобы мог быть проанализирован достаточно большой (часто с заранее заданным значением) процент программных кодов. Меры охвата программы, в зависимости от степени требуемой строгости, могут быть различными. В любом случае целью должно быть 100 % выбранной метрики охвата; если невозможно достигнуть 100 % охвата, то причины, почему 100 % охвата не может быть достигнуто, должны быть документально оформлены в отчете о тестировании (например, если возникает аппаратная проблема, желательно ввести только код защиты). В ГОСТ 34332.4—2021 (рекомендации к таблице Б.2) специально упомянуты широко поддерживаемые инструментами тестирования первые четыре метода, приведенные в перечислении ниже. Могут быть также применены следующие методы:

- охват точек входа (граф запросов) — гарантирует, что каждая подпрограмма (стандартная подпрограмма или функция) по крайней мере однажды должна быть вызвана (это наименее строгое структурное измерение охвата).

Примечание — В объектно-ориентированных языках может быть несколько подпрограмм с одним и тем же именем, которые применяются к различным вариантам полиморфизма (переопределяющие подпрограммы), которые могут вызываться динамической диспетчеризацией. В таких случаях должна быть протестирована каждая такая переопределенная подпрограмма;

- операторы — гарантирует, что все операторы в коде были выполнены по крайней мере однажды;

- условные переходы — должны быть проверены обе ветви каждого условного перехода. Это может оказаться нецелесообразным для некоторых типов кодов защиты;

- составные условия — анализируют каждое условие в составном условном переходе (связанное оператором И/ИЛИ). См. MCDC (охват условного модифицированного решения, документ DO178B);

- LCSAJ — последовательность линейного кода и переходов представляет собой любую линейную последовательность закодированных утверждений, включая условные утверждения, заканчивающиеся переходом. Многие потенциальные подпоследовательности могут оказаться невыполнимыми из-за ограничений, которые налагаются на входные данные в результате выполнения предыдущего кода;

- поток данных — выполняющиеся последовательности выбираются на основе используемых данных; например, последовательность, где одна и та же переменная и записывается, и считывается;

- базовая последовательность — одна из минимального набора конечных последовательностей от начала до конца, когда все дуги охвачены (перекрывающиеся комбинации последовательностей в этом базовом наборе могут сформировать любую последовательность через эту часть программы). Тесты всех базовых последовательностей показали свою эффективность при обнаружении ошибок.

Подробное описание данного метода/средства приведено в [58]—[60].

В.5.9 Анализ потока управления

Цель — обнаружение низкокачественных и потенциально некорректных структур программ.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.8).

Описание. Анализ потока управления представляет собой метод статического тестирования для нахождения подозреваемых областей программы, которые не соответствуют оправдавшей себя практике программирования. Программу анализируют, формируя направленный граф, который может быть проанализирован на наличие:

- недоступных фрагментов программы, например, безусловных переходов, которые делают фрагменты программы недостижимыми;

- запутанных кодов. Хорошо структурированный код имеет управляющий граф, допускающий сокращение путем последовательного сокращения графа до одного узла. В отличие от него плохо структурированный код может быть сокращен только до группы, состоящей из нескольких узлов.

Подробное описание данного метода/средства приведено в [60].

В.5.10 Анализ потока данных

Цель — обнаружение низкокачественных и потенциально некорректных структур программ.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.8).

Описание. Анализ потока данных представляет собой метод статического тестирования, объединяющий информацию, полученную из анализа потока управления, с информацией о том, какие переменные считываются или записываются в различных частях кода. Данный метод позволяет проверять:

- переменные, которые могут быть считаны до присвоения им значений. Такую ситуацию можно исключить, если всегда присваивать значение при объявлении новой переменной;
- переменные, записанные несколько раз, но несчитанные. Такая ситуация может указывать на пропущенный код;
- переменные, которые записаны, но никогда не считываются. Такая ситуация может указывать избыточный код.

Аномальный поток данных не всегда непосредственно соответствует программным ошибкам, но если аномалии исключены, то маловероятно, что код будет содержать ошибки.

Подробное описание данного метода/средства приведено в [60], [152].

В.5.11 Тестирование на символьном уровне

Цель — демонстрация соответствия между исходным кодом и спецификацией.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.8).

Описание. Переменные программы оценивают после замены во всех операторах присваивания левой его части на правую. Условные ветви и циклы преобразуют в булевские выражения. Окончательный результат представляет собой символьное выражение для каждой переменной программы. Это выражение является формулой для значения, которое программа вычислила бы, при задании реальных данных. Оно может быть проверено по отношению к предполагаемому выражению.

Такое использование символьного выполнения тестирования является систематическим способом генерации тестовых данных для ветвей логики программы. Символьное средство выполнения тестирования может быть включено в интегрированный комплекс инструментальных средств для выполнения тестирования и анализа кода элемента программного обеспечения.

Подробное описание данного метода/средства приведено в [153], [154].

В.5.12 Формальное доказательство (верификация)

Цель — доказательство правильности программы по отношению к некоторой абстрактной модели программы с использованием теоретических и математических моделей и правил.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблицы А.5 и А.9).

Описание. Тестирование — распространенный способ исследования правильности программы. Однако исчерпывающее тестирование обычно недостижимо ввиду сложности программ, имеющих практическое значение, поэтому таким способом может быть исследована только часть возможного поведения программы. Напротив, при формальной верификации применяют математические операции к математическому представлению программы, чтобы установить, что программа ведет себя, как определено для всех возможных входных данных.

Для формальной верификации системы требуется абстрактная модель программы и ее заданное поведение (т. е. спецификация), представленные на формальном языке. Спецификация может быть полной или может быть ограничена определенными свойствами программы:

- свойствами функциональной корректности (т. е. программа должна продемонстрировать определенную функциональность);
- свойствами безопасности (т. е. неправильное поведение никогда не будет происходить) и живучести (т. е. в конечном счете будет вести себя правильно);
- синхронизирующими свойствами (т. е. события, реализующие поведение, произойдут в определенное время).

Результатом формальной верификации является однозначный вывод о том, что абстрактная модель программы корректна по отношении к спецификации для всех возможных входных данных, то есть модель удовлетворяет заданным свойствам.

Однако правильность модели непосредственно не доказывает правильность фактической программы, поэтому далее необходим шаг, который должен показать, что модель — это точная абстракция фактической программы для моделируемых свойств. Некоторые свойства программы, представляющие практический интерес, не могут быть формально описаны (например, большинство проблем синхронизации и планирования или субъективные свойства, такие как «ясный и простой» пользовательский интерфейс, или любое свойство или цель проекта, которые не могут быть легко представлены на формальном языке). Поэтому формальная верификация не полностью

заменяет моделирование и тестирование, но зато она дополняет эти методы, обеспечивая их доказательством корректности работы программы для всех входных данных. Применение формальной верификации может гарантировать корректность абстрактной модели программы, а применение тестирования гарантирует, что фактическая программа ведет себя, как ожидалось.

Использование формальной верификации на стадии проектирования позволяет значительно сократить время разработки программы благодаря обнаружению существенных ошибок и упущению на ранних стадиях проектирования, таким образом сокращая время, необходимое на итерации между проектированием и тестированием.

Ряд формальных методов, применяющихся на практике, описаны в В.2.4: например, CCS, CSP, HOL, LOTOS, OBJ, временная логика, VDM и Z.

В.5.12.1 Проверка модели

Проверка модели — это метод для формальной верификации реактивных и конкурентных систем. Задавая конечное состояние структуры, которая описывает поведение системы, проверяют свойство, записанное в виде формулы во временной логике, соответствует ли оно структуре. Для автоматического и исчерпывающего прохода всех состояний структуры используют эффективные алгоритмы (например, SPIN, SMV и UPPAAL). Если свойство не удовлетворяется, то генерируется контрпример. Эта процедура показывает, как свойства нарушаются в структуре, и содержит очень полезную информацию для исследования системы. Применение метода проверки модели позволяет обнаружить глубоко скрытые (подводные) ошибки, которые могут быть не обнаружены при обычном контроле и тестировании.

Необходимо отметить, что метод проверки модели полезен при анализе нюансов сложной структуры. Он может быть полезен для некоторых приложений с низким УПБ. Однако необходимо быть очень внимательным, если в приложениях с высоким УПБ существуют нюансы сложной структуры.

Подробное описание данного метода/средства приведено в [155]—[158].

В.5.13 Метрики сложности программного обеспечения

Цель — прогнозирование характеристик программ, исходя из свойств самих программ или их разработки либо предысторий тестирования.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблица А.9 и приложение В, таблица В.19).

Описание. Данными методами оценивают некоторые структурные свойства ПО и их отношения к требуемым характеристикам, например, надежности или сложности. Для оценки большинства средств требуются программные инструменты. Некоторые применяющиеся метрики перечислены ниже:

- теоретическая сложность графа. Эта метрика может быть применена на раннем этапе жизненного цикла для оценки компромиссных решений и основана на величине сложности графа управления программы, представленной ее цикломатическим числом;
- число способов активизации некоторых программных модулей (доступность) — чем больше программных модулей может быть доступно, тем должна быть большая вероятность их отладки;
- теория метрик Холстеда. При помощи этих средств вычисляют длину программы путем подсчета количества операторов и операндов; данная метрика дает меру сложности и размеры, которые формируют основу для сравнений при оценке будущих разрабатываемых ресурсов;
- число входов и выходов на программный модуль. Сведение к минимуму числа точек входов/выходов является ключевой особенностью методов структурного проектирования и программирования.

Подробное описание данного метода/средства приведено в [159].

В.5.14 Формальные проверки

Цель — обнаружение ошибок в модуле ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.8).

Описание. Формальный контроль — это структурированный процесс проверки материалов о ПО, который выполняется коллегами разработчика этих материалов, для нахождения ошибок и оказания помощи разработчику улучшить материал. Разработчик не должен принимать участия в процессе контроля, кроме информирования проверяющих на этапе их ознакомления с материалами. Формальные проверки могут быть выполнены для конкретных элементов ПО, созданных на любой стадии ЖЦ разработки ПО.

Проверяющие должны ознакомиться с проверяемыми материалами до выполнения проверки. Роли проверяющих в процессе проверки должны быть ясно определены. Должна быть подготовлена программа проверки. Должны быть определены входные и выходные критерии, основанные на требуемых характеристиках элемента ПО. Входные критерии — это такие критерии или требования, которые должны быть удовлетворены до выполнения проверки. Выходные критерии — это такие критерии или требования, которые должны быть удовлетворены, чтобы считать процесс проверки завершенным успешно.

В процессе проверки ее результаты должны быть формально зафиксированы модератором, роль которого должна упростить проверку. По результатам проверки всеми проверяющими должно быть достигнуто согласие. Ошибки должны быть разделены на требующие исправления до принятия элемента ПО и требующие исправления

к заданному моменту времени или этапу. После завершения проверки выявленные ошибки должны быть переданы разработчику для последующего исправления. В зависимости от количества и контекста выявленных ошибок модератор может решить вопрос о необходимости повторной проверки материалов о ПО.

Подробное описание данного метода/средства приведено в [58]—[60], [160].

В.5.15 Сквозной контроль (программного обеспечения)

Цель — обнаружение несоответствий между спецификацией и реализацией.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.8).

Описание. Сквозной контроль является неформальным методом, выполняемым разработчиком элемента ПО в присутствии его коллег в целях обнаружения ошибок в элементе ПО. Он может быть выполнен для конкретных элементов ПО, созданных на любой стадии ЖЦ разработки ПО.

Чтобы гарантировать, что Э/Э/ПЭ СБЗС система соответствует требованиям, заданным в спецификации, необходимо исследовать и оценить заданные в спецификации функции безопасности системы. Любые сомнения, связанные с реализацией и использованием создаваемой системы, документально оформляют в целях их дальнейшего разрешения. В отличие от формальной проверки в процедуре сквозного контроля разработчик принимает активное участие.

Подробное описание данного метода/средства приведено в [58]—[60].

В.5.16 Анализ проекта

Цель — выявление дефектов в проекте ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение В, таблица В.8).

Описание. Под анализом проекта понимают формальное, документально оформленное, всестороннее и систематическое исследование проекта ПО в целях оценки требований к проекту и возможности проекта удовлетворить этим требованиям, а также для определения проблем и предложений по их решению.

Анализ проекта является средством оценки соответствия состояния проекта входным требованиям, а также средством идентификации возможностей для его дальнейшего усовершенствования. Даже если разработка проекта на этапах ЖЦ выполняется успешно, и основные конкретные проектные требования удовлетворены, то анализ проекта должен быть выполнен для того, чтобы исследовать все интерфейсные аспекты; гарантировать, что проект может быть верифицирован, удостовериться, что проект удовлетворяет проектным требованиям; гарантировать, что проект в наибольшей степени удовлетворяет требованиям безопасности. Такой анализ предназначен для проверки результатов работы разработчиков и должен рассматриваться как действия по подтверждению и совершенствованию этих результатов.

Чтобы обнаружить неправильное поведение ПО (такое как непредвиденные последовательности выполнения операторов или логики, непреднамеренные выходы, неправильная синхронизация, нежелательные действия), может быть использован строгий метод проверки, такой как «анализ паразитных цепей».

Подробное описание данного метода/средства приведено в [55], [58]—[60].

В.5.17 Макетирование/анимация

Цель — проверка возможности реализации системы при наличии заданных ограничений; увязка интерпретации разработчика спецификации системы с ее потребителем для исключения непонимания между ними.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.3 и Б.5).

Описание. Выделяются подмножество системных функций, ограничения и требования к рабочим параметрам. С помощью инструментов высокого уровня строят макет. На данном этапе не требуется рассматривать ограничения, например, используемый компьютер, язык реализации, объем программ, обслуживание, надежность и доступность. Макет оценивают по критериям потребителя, и в рамках этой оценки могут быть модифицированы системные требования.

Подробное описание данного метода/средства приведено в [56].

В.5.18 Моделирование процесса

Цель — тестирование функции программной системы вместе с ее интерфейсами во внешнем окружении, не допуская модификации реального окружения.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблица А.7; приложение Б, таблица Б.3 и приложение В, таблицы В.7 и В.13).

Описание. Создание системы только для целей тестирования, имитирующей поведение управляемого оборудования (УО).

Имитация может быть осуществлена только программными средствами либо сочетанием ПО и АС. Она должна:

- обеспечить входные данные, эквивалентные тем, которые реализуются на реальной установке УО;

- реагировать на выходные результаты тестирования ПО способом, точно отражающим объект управления;
- иметь возможность для оператора вводить входные данные, чтобы обеспечить любые возмущения, с которыми должна справиться тестируемая система.

Когда тестируется ПО, то моделируются заданные АС с их входными и выходными данными.

Подробное описание данного метода/средства приведено в [161].

В.5.19 Требования к реализации

Цель — установление демонстрируемых требований к функционированию системы ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.6).

Описание. Выполняют анализ как системы, так и спецификаций требований к ПО для спецификации всех общих и конкретных, явных и неявных требований к функционированию.

Каждое требование к функционированию анализируют по очереди для того, чтобы определить:

- критерии успешности результата, который следует получить;
- возможность получения меры критерия успешности;
- потенциальную точность таких результатов измерения;
- этапы проектирования, на которых эти результаты измерения могут быть оценены;
- этапы проектирования, на которых могут быть получены эти результаты измерений.

Затем анализируют целесообразность каждого требования к функционированию для получения списка требований к функционированию, критериев успешности результата и возможных результатов измерений. Основными целями являются:

- установление, что каждое требование к характеристикам связано хотя бы с одним измерением;
- выбор (где это возможно) точных и эффективных мер, которые могут быть использованы на самых ранних стадиях разработки;
- спецификация важных и факультативных требований к характеристикам и критериям успешности результата;
- использование (по возможности) применения одного измерения для нескольких требований к характеристикам.

Подробное описание данного метода/средства приведено в [56].

В.5.20 Моделирование реализации

Цель — гарантирование того, что рабочая производительность системы достаточна для удовлетворения специфицированных требований.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение Б, таблицы Б.2 и Б.5).

Описание. Спецификация требований включает в себя требования к пропускной способности и реакции конкретных функций, возможно, в сочетании с ограничениями на использование общих системных ресурсов. Предложенный проект системы сравнивают с установленными требованиями посредством:

- создания модели процессов системы и их взаимодействий;
- определения используемых каждым процессом ресурсов (время процессора, полоса пропускания канала связи, объем памяти и т. п.);
- определения распределения запросов к системе при средних и наихудших условиях;
- вычисления средних и наихудших значений величин пропускной способности и времени отклика для конкретных функций системы.

Для простых систем может оказаться достаточным аналитическое решение, тогда как для более сложных систем более подходящим для получения точных результатов является создание модели системы.

Перед детальным моделированием может быть использована более простая проверка «бюджета ресурсов», при которой суммируют требования к ресурсам для всех процессов. Если сумма этих требований к системе превышает возможности спроектированной системы, проект считают не реализуемым. Даже в случае, если проект проходит эту простую проверку, моделирование выполнения может показать, что имеются слишком большие задержки и времена откликов происходят из-за недостатка ресурсов. Для исключения такой ситуации инженеры часто проектируют системы, в которых используется только часть (например 50 %) общих ресурсов для уменьшения вероятности нехватки ресурсов.

Подробное описание данного метода/средства приведено в [56].

В.5.21 Проверка на критические нагрузки/стресс-тестирование

Цель — проверка тестируемого объекта при исключительно высокой нагрузке для демонстрации того, что тестируемый объект будет легко выдерживать нормальную рабочую нагрузку.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.6).

Описание. Существует множество тестов для проверки на критические нагрузки или стресс-тестирование, например:

- если работа объекта происходит в режиме упорядоченного опроса, то объект тестирования подвергается гораздо большим входным изменениям в единицу времени, чем при нормальных условиях;
- если работа объекта происходит по запросам, то число запросов в единицу времени для тестируемого объекта увеличивается по сравнению с числом запросов при нормальном режиме работы;
- если объем базы данных играет важную роль, то этот объем увеличивают относительно ее объема при нормальных условиях;
- имеющие решающее влияние устройства настраивают на свои максимальные скорости или самые малые скорости соответственно;
- для экстремальных тестов все факторы, имеющие решающее влияние, по возможности, вводят одновременно в граничные условия.

Для указанных выше тестов может быть оценено поведение во времени тестируемого объекта. Могут быть также исследованы изменения нагрузки и проверка размеров внутренних буферов или динамических переменных, стеков и т. п.

В.5.22 Ограничения на время ответа и объем памяти

Цель — обеспечение соответствия системы требованиям к параметрам времени и памяти.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение Б, таблица Б.6).

Описание. Спецификация требований к Э/Э/ПЭ СБЗС системе и ПО включает в себя требования к памяти и времени выполнения системой конкретных функций, возможно, совместно с ограничениями на использование общих системных ресурсов.

Данный метод применяют для определения распределения запросов при средних и наихудших условиях. Рассматриваемый метод требует оценки используемых ресурсов и затраченного времени каждой функцией системы. Такие оценки могут быть получены различными способами, например, сравнением с существующей системой или макетированием и дальнейшим сравнением времени реакции с критическими системами.

В.5.23 Анализ влияния

Цель — определение влияния, изменяющего или расширяющего программную систему, которому могут подвергаться также и другие программные модули в данной программной системе, а также другие системы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.8).

Описание. Перед выполнением модификации или расширением ПО следует определить влияние модификаций или расширений на ПО, а также определить, на какие программные системы и программные модули это повлияет.

Далее принимают решение о повторной верификации программной системы. Оно зависит от числа подвергнувшихся воздействию программных модулей, их критичности и характера изменений. Возможными решениями могут быть:

- повторная проверка только изменений программного модуля;
- повторная проверка всех подвергнувшихся воздействию программных модулей;
- повторная проверка всей системы.

Подробное описание данного метода/средства приведено в [162].

В.5.24 Управление конфигурацией программного обеспечения

Цель — обеспечение согласованности результатов работы групп, ответственных за ожидаемые результаты разработки, поскольку эти результаты меняются. В общем случае управление конфигурацией применимо к разработке как АС, так и ПО.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.8).

Описание. Управление конфигурацией ПО представляет собой метод, используемый в течение всей разработки (см. ГОСТ 34332.4—2021, пункт 6.2.3). При его применении требуется документальное оформление разработки каждой версии, каждого значимого результата каждой версии, каждой взаимосвязи между результатами различных версий. Полученная документация позволяет разработчику определять, как влияет на другие результаты изменение одного модуля (особенно одного из его элементов). В частности, системы или подсистемы могут быть надежно скомпонованы (сконфигурированы) из согласованных наборов версий модулей.

Подробное описание данного метода/средства приведено в [59], [60], [163].

В.5.25 Регрессионное подтверждение соответствия

Цель — гарантирование того, что обоснованные выводы сделаны из регрессионного тестирования.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.8).

Описание. Полное регрессионное тестирование большой или сложной системы обычно требует больших усилий и ресурсов. По возможности желательно ограничить регрессионное тестирование, охватив только системные аспекты, представляющие в данный момент основной интерес для разрабатываемой системы. В таком частичном регрессионном тестировании важно иметь четкое понимание области применения этого частичного тестирования и сделать строго обоснованные выводы о тестируемом состоянии системы.

Подробное описание данного метода/средства приведено в [145].

В.5.26 Анимация спецификации и проектирования

Цель — осуществление верификации ПО посредством систематической проверки спецификации.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.9).

Описание. Для определения поведения конечного исполнимого ПО рассматривают представление ПО на более высоком уровне абстракции, чем исполняемый код (например, спецификацию или описание проекта). Проверку автоматизируют каким-то образом (в зависимости от формы и возможностей представления абстракции более высокого уровня) для имитации поведения и результатов исполняемого ПО. Одним из применений этого подхода является генерирование тестов («оракулов»), которые могут быть впоследствии применены к исполняемому ПО для автоматизации в некоторой степени процесса тестирования. Другое применение — это анимация пользовательского интерфейса, для того, чтобы нетехнические конечные пользователи могли более детально понять смысл спецификации, с которой работают разработчики ПО. Это облегчает взаимодействие между разработчиками ПО и конечными пользователями.

Подробное описание данного метода/средства приведено в [164].

В.5.27 Тестирование, основанное на модели (генерация тестов)

Цель — обеспечение эффективной автоматической генерации тестовых примеров из моделей системы и создания наборов тестов с высокой воспризводимостью.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.5).

Описание. В методе тестирования, основанном на модели (MBT), использует подход «черного ящика», в котором общие задачи тестирования, такие как генерация тестовой комбинации (TCG) и оценка результатов тестирования, основаны на модели тестируемой (прикладной) системы (SUT). Как правило, но не всегда, данные о системе и поведение пользователя смоделированы с использованием методов конечных автоматов, марковских процессов, таблиц решений и т. п. Кроме того, тестирование, основанное на модели, может быть объединено с измерением тестового охвата на уровне исходного кода, а функциональные модели могут быть основаны на существующем исходном коде.

Тестирование, основанное на модели, обеспечивает автоматическую генерацию эффективных контрольных примеров/процедур, используя модели системных требований и заданной функциональности.

Так как тестирование — очень дорогой процесс, существует большой спрос на автоматические средства генерации тестов. Поэтому направление тестирования, основанное на модели, в настоящий момент активно исследуется, и создается большое число доступных средств генерации тестовых комбинаций. Эти средства обычно формируют тестовую комбинацию из модели поведения системы, гарантируя при этом, что будут удовлетворены требования охвата диагностикой.

Модель является абстрактным частичным представлением требуемого поведения тестируемой системы. Из этой модели формируют примеры тестов, создавая абстрактную тестовую комбинацию. Из этой абстрактной тестовой комбинации выделяют контрольные примеры и проверяют систему. Кроме того, эти контрольные примеры могут быть также использованы для проверки и модели системы. Метод тестирования, основанный на модели, с генерацией тестовой комбинации основан на использовании формальных методов и тесно связан с ними, поэтому рекомендации по его применению аналогичны рекомендациям по УПБ: для высоких уровней полноты безопасности (УПБ): для более высоких УПБ — «ОР» (особо рекомендованный), для низких УПБ — применение не требуется «- -».

В общем случае метод состоит из набора следующих действий:

- создание модели (из системных требований);
- генерация ожидаемых входов;
- генерация ожидаемых выходов;
- выполнение тестов;
- сравнение фактических результатов с ожидаемыми выходами;
- выбор дальнейших действий (изменение модели, генерация дальнейших тестов, оценка надежности/качества ПО).

Для получения тестов могут быть использованы различные методы и средства представления моделей поведения пользователя/системы, например:

- таблицы решений;
- конечные автоматы;

- формальные грамматики;
- цепи Маркова;
- диаграммы состояний;
- доказательство теорем;
- логическое программирование с ограничениями;
- модель проверки;
- моделирование на символьном уровне;
- использование модели потока событий;
- параллельные иерархические конечные автоматы для тестирования реактивных систем и т. д.

Тестирование, основанное на модели, с недавних пор целенаправленно используют в областях, критичных для безопасности. Оно позволяет на ранних стадиях выявить неоднозначности в спецификации и проекте, обеспечивает возможность автоматически генерировать много неповторяемых эффективных тестов, оценивать регрессионный набор тестов и оценивать надежность и качество ПО, а также облегчать обновление наборов тестов.

Подробное описание данного метода/средства приведено в [165]—[173].

В.6 Оценка функциональной безопасности

Примечание — Соответствующие методы и средства см. также в Б.6.

В.6.1 Таблицы решений (таблицы истинности)

Цель — обеспечение ясных и согласующихся спецификаций и анализа сложных логических комбинаций и их отношений.

Примечание — Ссылки на данный метод/средство приведены в ГОСТ 34332.4—2021 (приложение А, таблица А.10 и приложение Б, таблица Б.7).

Описание. В данном методе используют бинарные таблицы для точного описания логических отношений между булевыми переменными программы.

Использование таблиц и точность метода позволили применить его в качестве средства анализа сложных логических комбинаций, выраженных в двоичных кодах.

Рассматриваемый метод достаточно легко поддается автоматизации, поэтому его можно использовать в качестве средства спецификации систем.

В.6.2 Исследование опасности и работоспособности программного обеспечения (CHAZOP, FMEA)

Цель — определение угроз безопасности в предлагаемой или существующей системе, их возможных причин, последствий и рекомендуемых действий по минимизации вероятности их появления.

Описание. Группа специалистов в области создаваемой системы принимает участие в структурном анализе проекта системы в ходе ряда запланированных совещаний. Они рассматривают как реализацию функций проекта системы, так и способы работы системы на практике (включая действия персонала и процедуры эксплуатации системы). Руководитель группы специалистов инициирует ее участников распознавать потенциальные опасности и управляет этой процедурой, описывая каждую часть системы в сочетании с отдельными ключевыми словами: «отсутствует», «более», «менее», «часть целого», «больше чем» (или «так же, как и») и «иначе чем». Каждое применимое условие или режим отказа рассматривается с точки зрения реализуемости, причин возникновения, возможных последствий (появляется ли опасность), способа устранения и, в случае устранения, выбора наиболее целесообразного метода.

Исследование опасностей может выполняться на разных стадиях разработки проекта, однако наиболее эффективным такое исследование может быть на начальных стадиях с тем, чтобы как можно раньше повлиять на основные решения по проектированию и работоспособности.

Метод HAZOP создавался для производственных процессов и требует модификации при его применении к ПО. Были предложены различные производные методы (Computer HAZOPs — «CHAZOPs»), которые сопровождались новыми руководящими материалами и/или реализовывали способы систематического охвата системной и программной архитектур.

Подробное описание данного метода/средства приведено в [109], [174], [175].

В.6.3 Анализ отказов по общей причине

Цель — определение возможных отказов в нескольких системах или нескольких подсистемах, которые могут свести к нулю преимущества избыточности из-за одновременного появления одних и тех же отказов во многих частях системы.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.10).

Описание. В системах, ориентированных на безопасность объекта (СБЗС системах), часто используют избыточность АС и мажоритарный принцип голосования. Этот подход исключает случайные отказы в компонентах или подсистемах АС, которые могут помешать корректной обработке данных.

Однако некоторые отказы могут оказаться общими для нескольких компонентов или подсистем. Например, если система установлена в одном помещении, то недостатки вентиляции могут снизить преимущества избыточности. Это может оказаться верным и для других внешних влияний на систему (например, пожар, затопление, электромагнитные воздействия, трещины в панелях и землетрясение). Система может быть также подвержена воздействиям, относящимся к ее функционированию и эксплуатации. Поэтому важно, чтобы в рабочих инструкциях были предусмотрены адекватные и хорошо задокументированные процедуры по функционированию и эксплуатации системы, а обслуживающий персонал был хорошо обучен.

Внутренние причины также вносят большой вклад в общее число отказов. Их основой могут являться ошибки проектирования общих или идентичных компонентов и их интерфейсов, в том числе и устаревших компонентов. Анализ отказов по общей причине применяют также для отыскания дефектов в системе. К методам анализа отказов по общей причине относятся: общее управление качеством; анализ проектов; верификация и тестирование независимой группой; анализ реальных ситуаций, полученных из опыта работы аналогичных систем. Однако область применения такого анализа выходит за рамки только АС. Даже если разные программы используют в разных каналах избыточных систем, возможна некоторая общность в программных подходах, которая может привести к росту отказов по общей причине (например, ошибки в общей спецификации).

Если отказы по общей причине не появляются точно в одно и то же время, то должны быть предприняты меры предосторожности путем сравнения методов, применяемых в различных каналах. При этом применение каждого метода должно приводить к обнаружению отказа до того, как он окажется общим для всех каналов. При анализе отказов по общей причине следует использовать этот подход.

Подробное описание данного метода/средства приведено в [176].

B.6.4 Структурные схемы надежности

Цель — моделирование в форме диаграмм набора событий, которые должны происходить, и условий, которые должны быть удовлетворены для успешного выполнения операций системы или задач.

Примечание — Ссылка на данный метод/средство приведена в ГОСТ 34332.4—2021 (приложение А, таблица А.10).

Описание. Данный метод позволяет сформировать маршрутную карту, состоящую из блоков, линий и логических переходов. Такая маршрутная карта начинается от одной стороны диаграммы и проходит через блоки и логические переходы до другой стороны диаграммы. Блок представляет собой условие или событие. Маршрут проходит через него, если условие истинно или событие произошло. Когда маршрут подходит к логическому переходу, то он продолжается, если выполняется критерий логического перехода. Если маршрут достигает какой-либо вершины, то он может продолжаться по всем исходящим из нее путям. Если существует по меньшей мере один успешный маршрут через всю диаграмму, то цель анализа считается достигнутой.

Подробное описание данного метода/средства приведено в [75], [77].

Приложение Г
(справочное)

Методы и средства для проектирования специализированных интегральных схем

В настоящем приложении приведен обзор методов и средств, на которые даны ссылки в ГОСТ 34332.3. В нем также приведены ссылки на источники с более полным описанием этих методов и средств. Настоящее приложение не должно рассматриваться как полное или исчерпывающее.

Примечание — При проектировании Э/Э/ПЭ СБЗС специализированные интегральные схемы обычно не проектируют, за исключением уникальных случаев. Данное приложение включено в настоящий стандарт для того, чтобы предоставить проектировщику дополнительные сведения, необходимые ему для сознательного заказа разработки или выбора специализированных интегральных схем, пригодных для применения в конкретной проектируемой Э/Э/ПЭ СБЗС системе.

Г.1 Описание проекта на языке (V)HDL

Цель — функциональное описание технических средств на языке высокого уровня, например, VHDL или Verilog.

Описание. Абстрактное функциональное описание проекта АС осуществляют на языке высокого уровня, например, VHDL или Verilog. Используемый язык должен обеспечивать возможность функционального и/или прикладного описания АС и должен быть абстрагирован от более поздних деталей реализации. Потоки данных, условные переходы, арифметические и/или логические операции должны быть реализованы с использованием присваивания назначения и операторов языка описания АС без ручного преобразования в логические вентильные структуры прикладной библиотеки.

Примечание — Для простоты «функциональное описание технических средств на языке высокого уровня» далее будет обозначено как (V)HDL.

Подробное описание данного метода/средства приведено в [177].

Г.2 Ввод описаний схем

Цель — функциональное описание компонуемой схемы при ее представлении на уровне логических элементов и/или макросов библиотеки поставщика.

Описание. Описание функциональности компонуемой схемы формируется при вводе описаний схем и связей между ними. Функцию, которая должна быть реализована, компонуют (собирают) из элементарных логических схем, таких как «И», «ИЛИ», «НЕ», а также макросов, состоящих из сложных арифметических и логических функций, которые затем соединяют между собой. Сложные функциональные схемы должны быть иерархически декомпозированы и могут быть представлены на различных иерархически связанных рисунках. Межсоединения должны быть уникально определены и иметь конкретные имена сигналов по всей иерархии. Насколько это возможно, необходимо избегать использования глобальных сигналов (меток).

Г.3 Структурное описание

Цель — структурирование описания функциональности схемы выполняют таким способом, чтобы оно было легко воспринимаемо, то есть, чтобы функция схемы могла быть интуитивно понята из описания без привлечения моделирования.

Примечание — См. также В.2.7 приложения В.

Описание. Описание функциональности схемы формируют на языке (V)HDL или вводом описания схем. Рекомендуется легко узнаваемая модульная структура. Каждый модуль должен быть реализован одним и тем же способом и должен быть описан так, чтобы можно было легко понять все его подфункции. Рекомендуется строго разделять описание реализуемой функции и описание структуры на уровне межсоединений, то есть модуль, реализуемый из нескольких других подмодулей, должен содержать только описание межсоединений этих подмодулей и не должен содержать описание логики схемы.

Г.4 Средства, проверенные в эксплуатации

Цель — применение средств, проверенных на практике, для предотвращения систематических отказов при условии, что эти средства достаточно долго и успешно применялись в различных проектах.

Описание. Большинство используемых средств для разработки СИС и полевых программируемых вентильных матриц (ППВМ) включает в себя сложное программное обеспечение, которое не может считаться работающим без каких-либо ошибок, а также довольно часто ошибки могут возникнуть вследствие неправильной эксплуатации таких средств. Поэтому для разработки СИС и ППВМ необходимо использовать только средства с атрибутом «проверено на практике». Это подразумевает:

- применение средств, которые использовались (в сопоставимой версии ПО) в течение длительного периода времени или большим числом пользователей в различных проектах такой же сложности;

- наличие у каждого разработчика СИС/ППВМ опыта работы с этими средствами в течение продолжительного периода времени;

- применение общеиспользуемых средств (соответствующим числом пользователей) так, чтобы была доступна информация об известных отказах от всех пользователей (в процессе управления версиями со «списком ошибок»). Эта информация должна быть быстро интегрирована в процесс проектирования, чтобы помочь избежать систематических отказов;

- проверку целостности внутренних средств базы данных и проверку достоверности данных, что поможет избежать ошибок данных, получаемых из базы данных. Стандартными инструментами проверяют согласованность внутренней базы данных, например, непротиворечивость данных, полученных в результате работы средств синтеза, и полученных в результате размещения и трассировки (place-and-route-tool), для работы с уникальными данными.

Примечание — Проверка непротиворечивости — это обязательная функция используемого средства, и разработчик на нее влияет слабо. Поэтому, если существует возможность ручной проверки непротиворечивости, разработчик должен адекватно ее использовать.

Г.5 Моделирование на языке (V)HDL

Цель — функциональная проверка схемы, описанной на языке (V)HDL, средствами моделирования.

Примечание — См. также Г.6 приложения Г.

Описание. Проверку функции осуществляют с помощью моделирования всей схемы или каждого подмодуля. Модель на языке (V)HDL обнаруживает последовательность выходов, вызванную внутренним изменением состояний схемы в результате применения входных стимулов. Проверка обнаруженной выходной последовательности может быть выполнена либо путем предварительной последовательности выходных сигналов («форма волны»), либо специальной средой, известной как «испытаний стенд», который устанавливают в процессе проектирования. Для обеспечения получения правильных результатов и маскирования ошибочного временного поведения сигналов (в шинах и трассировках с тремя устойчивыми состояниями), которое может быть вызвано ошибкой моделирования, выбирают модель, классифицируемую как «проверенную в эксплуатации».

Г.6 Функциональное тестирование на уровне модуля

Цель — функциональное тестирование «снизу-вверх».

Примечание — См. также Г.5, Г.13 приложения Г.

Описание. Проверку реализуемой функции осуществляют, например, посредством моделирования на уровне модулей. Испытуемый модуль инсталлируют в типичной виртуальной тестовой среде, известной как «стенд» и стимулируют тестовой схемой, содержащейся в коде. Испытуемый модуль инсталлируют в типичную виртуальную тестовую среду, известную как «испытательный стенд», и моделируют тестовой схемой, содержащейся в коде. Требуется достаточно высокий охват указанной функции, включая все особые случаи, если они существуют. Автоматическая проверка выходной последовательности по коду «испытательного стенда» должна быть более приоритетной по сравнению с ручной проверкой.

Г.7 Функциональное тестирование на верхнем уровне

Цель — проверка всей специализированной интегральной схемы (СИС).

Примечание — См. также Г.8 приложения Г.

Описание. Задача этого теста — проверка всей СИС.

Г.8 Функциональное тестирование во внешней системной среде

Цель — проверка заданной функции, встроенной в системное окружение.

Примечание — См. также Г.7 приложения Г.

Описание. Этим тестом проверяют всю функциональность СИС в ее системной среде, например, со всеми другими компонентами, которые расположены на печатных платах или в других местах. Моделирование всех соответствующих компонентов на печатной плате и моделирование СИС, на основе ее модели, для проверки корректной функциональности рекомендуется проводить с учетом сигналов синхронизации. Полное функциональное тестирование включает также тестирование модулей, которые становятся активными только во время отказа.

Г.9 Ограниченное использование асинхронных структур

Цель — предотвращение типичных проблем синхронизации в процессе синтеза, предотвращение неоднозначности в процессе моделирования и синтеза, связанных с трудностями создаваемой модели, проектирование тестируемости.

Описание. Асинхронные структуры, формирующие такие сигналы, как УСТАНОВКА и СБРОС, построенные на комбинаторной логике, чувствительны к процедуре синтеза. В результате могут формироваться схемы, создающие импульсные помехи или срабатывающие от инверсной синхронизации, и поэтому их необходимо избегать. Такие проблемы при создании модели не могут быть интерпретированы должным образом средствами синтеза, что приводит к неоднозначным результатам при моделировании асинхронных структур. Так как асинхронные структуры являются плохо тестируемыми или нетестируемыми вовсе, то эффективность охвата тестами при заводских испытаниях или тестирования в режиме онлайн снижается. Поэтому рекомендуется реализовывать полностью синхронную систему с ограниченным числом синхросигналов. В системах с многофазной синхронизацией все синхросигналы должны быть сформированы из одного центрального генератора синхросигналов. Синхронизацию на входе последовательностной логики всегда следует осуществлять только синхросигналом, который не содержит комбинаторной логики. Входы последовательностных ячеек асинхронных структур УСТАНОВКА и СБРОС всегда должны быть обеспечены синхросигналами, которые не содержат комбинационной логики. Устройства, задающие сигналы УСТАНОВКА и СБРОС, следует синхронизировать с помощью двух триггеров.

Г.10 Синхронизация основных входов и управление метастабильностью

Цель — предотвращение неоднозначного поведения схемы в результате нарушения времен установки и промежуточного хранения.

Описание. Входные сигналы от внешних периферийных устройств являются обычно асинхронными и могут произвольно изменять свое состояние. Непосредственная обработка таких сигналов выполняется синхронными элементами последовательностных схем СИС/ППВМ. Например, использование триггеров ведет к нарушению времени установки и промежуточного хранения, что приводит к непредсказуемым нарушениям в синхронизации и функциональном поведении СИС/ППВМ. Это может привести к метастабильности элемента памяти. Поэтому, чтобы избежать функциональной неоднозначности, каждый асинхронный входной сигнал должен быть синхронизован системой синхронизации ASIC. Рекомендуются следующие меры:

- входные сигналы следует синхронизировать двумя последовательными элементами памяти (триггерами) или некоторой эквивалентной схемой, чтобы достигнуть предсказуемого функционального поведения;
- каждый асинхронный входной сигнал следует обязательно синхронизировать способом, определенным выше, то есть каждый асинхронный сигнал следует синхронизовать только от одной такой схемы синхронизации. В случае необходимости выход схемы синхронизации может быть использован для множественного доступа;
- такую схему синхронизации следует использовать для проверки устойчивости сигналов параллельной шины и управлять согласованностью данных в точке выборки.

Г.11 Проектирование тестируемости

Цель — предотвращение нетестируемых или плохо тестируемых структур, чтобы обеспечить высокий уровень тестового охвата при заводских испытаниях или тестирования в режиме онлайн.

Примечание — См. также Г.31 приложения Г.

Описание. Проектом по тестированию необходимо управлять путем избегания:

- асинхронных структур;
- защелок и трех состояний сигналов на микросхеме;
- логических «И», логических «ИЛИ» на проводных соединениях и избыточной логических элементов.

Комбинационная глубина вспомогательных цепей играет важную роль в процессе тестирования. Тестовая комбинация, необходимая для полного тестирования, экспоненциально возрастает с комбинационной глубиной цепи. Поэтому схемы с высокой комбинационной глубиной плохо тестируются адекватными средствами.

При подходе, ориентированном на проектирование тестируемости, гарантируется, что требуемый тестовый охват будет достигнут. Поскольку фактический тестовый охват может быть определен на очень поздней стадии процесса проектирования, недостаточное внимание к проблемам «проектирования тестируемости» может привести к существенному уменьшению достижимого тестового охвата и увеличению объема работ.

Примечание — Тестовый охват обычно определяется процентом обнаруженных константных отказов.

Г.12 Разбиение на модули

Цель — описание модулей, реализующих функции схемы.

Примечание — См. также В.2.8, В.2.9 приложения В и Г.3 приложения Г.

Описание. Применяют строгое разделение полной функциональности по различным модулям с ограниченными функциями. Таким способом обеспечивают прозрачность модулей с точно определенным интерфейсом. Каждую подсистему на всех уровнях проекта явно определяют и ограничивают по размеру (только несколько функций). Интерфейсы между подсистемами сохраняют настолько простыми, насколько это возможно, и пересечение модулей (то есть совместно используемые данные, обмен информацией) минимизируют. Также ограничивают сложность отдельных подсистем.

Г.13 Охват сценариями верификации (испытательные стенды)

Цель — количественная и качественная оценка применяемых сценариев верификации в процессе функционального тестирования.

Описание. Качество сценариев верификации определяют в процессе функционального тестирования, то есть применяемый тестовый набор для верификации конкретной функции, включая все особые случаи, если они существуют, должен быть документально оформлен в количественной или качественной форме. При количественном подходе должны быть документально оформлены достигнутый тестовый охват и глубина примененных функциональных тестов. Получающийся охват должен удовлетворять уровням, установленным для каждой из метрик охвата. Любое исключение должно быть документально оформлено. В случае качественного подхода должно быть оценено число проверенных строк кода, инструкций или путей («охват кода») проверяемого кода схемы.

Примечание — Особый метод анализа «охват кода» имеет ограниченное применение из-за высокого параллелизма описания АС и необходимости обоснования исчерпывающими проверками. «Охват кода» обычно служит для демонстрации неохваченного функционального кода.

Г.14 Соблюдение руководств по кодированию

Цель — строгое соблюдение стиля кодирования, приводящее к синтаксически и семантически корректному коду схемы.

Описание. Синтаксические правила кодирования помогают создать легко читаемый код и лучшую документацию, включая управление версиями. Обычно руководства содержат правила по организации и комментированию блоков или модулей схемы.

Применение семантических правил кодирования помогает предотвратить типичные проблемы реализации с помощью устранения структур, которые приводят к ошибке синтеза, — неоднозначной реализации функции схемы. Типичным правилом является, например, предотвращение асинхронных структур или структур, которые приводят к непредсказуемой последовательности синхросигналов. Обычно к таким неоднозначностям приводит использование защелок или взаимодействие данных с синхросигналами.

В руководствах по разработке рекомендуют избегать систематических проектных сбоев во время процесса разработки СИС. Соблюдение стиля кодирования в определенном смысле ограничивает эффективность проекта, однако, в свою очередь, дает преимущество в предотвращении сбоев во время процесса разработки СИС. Соблюдение стиля кодирования, в частности:

- предотвращает типичные недостатки кодирования или сбои;
- ограничивает использование проблемных структур, которые приводят к неоднозначным результатам синтеза;
- применяется для проектирования тестируемости;
- обеспечивает прозрачный и удобный код.

Пример стиля кодирования

1 Код должен содержать столько комментариев, сколько это необходимо для понимания деталей реализации и функции. Используемые соглашения должны быть определены перед началом проекта. В течение стадии проектирования должно быть проверено соответствие определенных соглашений.

1.1 В стандартные заголовки включают историю, перекрестные ссылки на спецификацию, информацию об ответственности и данные, сопровождающие проект, такие как номер версии, запросы на изменение и т. д.

1.2 Легко читаемые шаблоны: эквивалентные процессы должны быть описаны одинаковой процедурой, то есть использование предопределенных шаблонов для повторяющихся процессов («если-то-иначе», «для» и т. д.).

1.3 Точное и читаемое соглашение о присвоении имен, например, прописная/строчная буква, префикс и постфикс, точная дифференциация между именем порта, внутренними сигналами, константами, переменными, низкий активный уровень (xxx_n) и т. д.

1.4 Должны быть введены ограничения на размер модуля и число портов на модуле для увеличения читаемости кода.

1.5 Структурированная и защищенная разработка кода, например информация о состоянии, должна быть инкапсулирована в FSM (сокрытие информации), чтобы обеспечить легкое изменение кода.

1.6 Должны быть реализованы проверки достоверности, такие как проверка диапазона и т. д.

1.7 Предотвращение следующих структур/команд:

- использование просмотра диапазона по индексу в порядке возрастания ключа (от x к y) для сигналов шины;

- команды «Отключить» в Verilog (соответствует команде goto);

- многомерных массивов (>2), записей;

- сочетания типов данных без знака и со знаком.

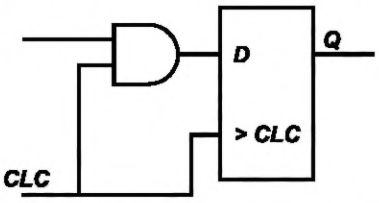
2 Завершенный проект синхронизации (допускается получение тактовых импульсов только от центральных тактовых генераторов).

2.1 Выходы модуля должны быть синхронизированы, что также обеспечит тестируемость и статический анализ временных диаграмм.

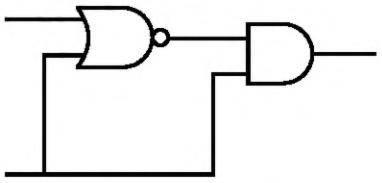
2.2 Управляющие импульсы должны быть обработаны с особыми предосторожностями.

3 Предотвращение связи сигналов данных с синхросигналами повышает тестируемость, воспроизводимость между данными до и после компоновки и уровень соответствия поведения на уровне межрегистровых пересылок (RTL).

4 Избыточная логика не является тестируемой, и ее следует избегать:



Связь сигналов данных с синхросигналами



Избыточная логика

5 Необходимо избегать обратных связей в комбинационной логике, потому что это ведет к нестабильности проекта, и он не будет тестируемым.

6 Рекомендуется, чтобы проект был полностью просматриваемым (прогоняемым при моделировании).

7 Предотвращение защелок увеличивает тестируемость и сокращает ограничения синхронизации в процессе синтеза.

8 Сигнал основного сброса и все асинхронные входные сигналы следует синхронизировать двумя последовательными элементами памяти (триггерами) или эквивалентной(ыми) схемой(ами) (метастабильность).

9 Рекомендуется избегать асинхронных сигналов установки/сброса, за исключением сигнала основного сброса.

10 Сигналы на уровне порта модуля должны быть типа std_logic или std_logic_vector.

Г.15 Применение средств проверки кода

Цель — автоматическая проверка правил кодирования («стиль кодирования») инструментальными средствами проверки кода.

Описание. Применение средств проверки кода помогает в значительной степени автоматически соблюдать стиль кодирования и генерировать документацию в режиме онлайн. Однако автоматическое средство проверки кода позволяет проверить синтаксис и семантику кода в целом. Поэтому применение таких инструментов следует сопровождать расширением общих правил кодирования («конкретный инструмент») с помощью проектирования специальных правил кодирования, которые разработчик должен реализовывать и оценивать отдельно.

Г.16 Программирование с защитой

Примечание — См. В.2.5 приложения В.

Г.17 Документирование результатов моделирования

Цель — документальное оформление всех данных, необходимых для успешного моделирования для проверки заданной функции схемы.

Описание. Все данные, необходимые для функционального моделирования на уровне модуля, микросхемы или системы должны быть правильно документально оформлены и заархивированы для того, чтобы:

- повторить моделирование на любой более поздней стадии посредством изменения параметров модели;
- продемонстрировать правильность и полноту всех определенных функций.

Должна быть заархивирована база данных, хранящая:

- средства моделирования, включающие полное ПО используемых инструментов, например, средств моделирования, синтеза с указанием версий и необходимой библиотеки моделирования;
- файл с журналом моделирования, который включает в себя полную информацию о времени моделирования, примененных инструментов с указанием версий и полный текст отчета о всей работе, если это необходимо;
- все соответствующие результаты моделирования, включая последовательности сигналов, особенно в случае ручной проверки, и документацию полученных результатов.

Г.18 Проверка кода

Цель — анализ описания схемы.

Примечание — См. В.5.14 приложения В.

Описание. Анализ описания схемы должен быть выполнен посредством:
- контроля стиля кодирования;
- проверки соответствия со спецификацией описанной функциональности;
- контроля кодирования с защитой, обработки ошибок и обработки исключений.

Примечание — Если моделирование на языке (V)HDL не выполнено, у полноты проверки кода и достигнутых результатов должно быть «эквивалентное качество», т. е качество, которое могло бы быть достигнуто при моделировании на языке (V)HDL.

Г.19 Сквозной контроль

Цель — проверка описания схемы с помощью сквозного контроля.

Описание. Сквозной контроль выполняют следующим образом: группа сквозного контроля выбирает небольшой набор тестовых примеров — представительные наборы входных и соответствующих ожидаемых выходных данных для программы. Затем вручную прослеживает прохождение тестовых данных через логику программы.

Примечание — Как самостоятельное средство его следует применять только к схемам с очень низкой сложностью. В случае сбоя при моделировании на языке (V)HDL у полноты сквозного контроля и качества достигнутых результатов должно быть эквивалентное качество, которое могло бы быть достигнуто посредством моделирования на языке (V)HDL.

Подробное описание данного метода/средства приведено в [55].

Г.20 Применение проверенных гибких проводников

Цель — предотвращение отказа при работе гибких проводников посредством применения проверенных гибких проводников.

Описание. Если поставщик проверяет гибкие проводники, должны быть выполнены следующие требования:
- проверку гибкого проводника следует выполнять для работы СБС, имеющей, по меньшей мере, эквивалентный или более высокий УПБ, чем планируемая к разработке система;
- должны быть выполнены все допущения и ограничения, необходимые для проверки гибкого проводника;
- должны быть легко доступны все необходимые документы для проверки гибкого проводника, см. также Г.17 «Документирование результатов моделирования»;
- должна быть строго соблюдена каждая спецификация поставщика, а доказательства соответствия должны быть задокументированы.

Г.21 Проверка гибкого проводника

Цель — предотвращение отказа гибкого проводника во время работы посредством проверки гибкого проводника в течение ЖЦ объекта.

Примечание — См. также Г.6 приложения Г.

Описание. Если гибкий проводник не был разработан специально для работы в СБС, должен быть проверен сгенерированный код в тех же условиях, которые применяются для проверки любого исходного кода. Это означает, что должны быть определены и реализованы все возможные тестовые примеры. Затем функциональная проверка должна быть выполнена посредством моделирования.

Г.22 Моделирование логической схемы на основе списка соединений для проверки ограничений синхронизации

Цель — независимая проверка достигаемого ограничения синхронизации во время синтеза.

Описание. Моделирование логической схемы на основе списка соединений, созданного на этапе синтеза, с учетом реальной длины соединений при расчете времени задержки в соединительных линиях и задержек в логических элементах. Должны быть сформированы такие входные сигналы, с помощью которых будет охвачен высокий процент ограничений синхронизации и будут включены все наихудшие случаи для синхронизации. Вообще входные сигналы должны быть выбраны такими, чтобы выполнялось функциональное тестирование на уровне модуля (Г.6) или функциональное тестирование на верхнем уровне (Г.7), подходящие критерии для выбора которого обеспечены при условии, что во время функционального теста может требоваться достаточный тестовый охват. Схема должна быть протестирована при наилучших и наихудших условиях при указанной максимальной тактовой частоте.

Проверка синхронизации может быть выполнена с помощью автоматической проверки времени установки и времени промежуточного хранения для элементов памяти (триггеров) целевой библиотеки, а также с помощью функциональной проверки схемы. Функциональная проверка в основном должна осуществляться с помощью анализа выходных сигналов микросхемы. Она может быть выполнена автоматически, посредством сравнения выходных сигналов схемы с соответствующей эталонной моделью или с исходным кодом схемы на языке (V)HDL. Этот тест известен как «регрессионный тест» и его выполнение должно быть более предпочтительным по сравнению с ручной проверкой выходных сигналов.

Примечание — Данный метод позволяет проверить работу системы синхронизации только для тех путей списка соединений логических элементов, которые фактически активны во время моделирования, и поэтому такой специально формируемый метод не может обеспечить полный временной анализ схемы в общем случае.

Г.23 Статический временной анализ задержки распространения сигнала

Цель — независимая проверка ограничений синхронизации, осуществляемая во время синтеза.

Описание. При статическом временном анализе СВА осуществляют анализ всех путей списка соединений (схемы), полученного средствами синтеза, с учетом реальной длины соединений при расчете времени задержки в линиях связи и задержек логических элементов без применения реального моделирования. Это позволяет в общем случае выполнить полный анализ ограничений синхронизации для всей схемы. Тестируемая схема должна быть проанализирована в наилучших и наихудших условиях эксплуатации, на максимально задаваемой тактовой частоте, с учетом возможной неустойчивости синхронизации и расфазировки ее рабочего цикла. Число путей, не соответствующих синхронизации, может быть ограничено определенным минимумом, в зависимости от используемого метода проектирования. Перед началом проектирования рекомендуется исследовать, проанализировать и определить применяемый метод, который должен обеспечить легко читаемые результаты.

Примечание — Можно предположить, что СВА явно охватывает все существующие пути синхронизации, если:

- ограничения синхронизации должным образом определены;
- тестируемая схема содержит только такие пути синхронизации, которые могут быть проанализированы средствами СВА, то есть в общем случае полностью синхронные схемы.

Г.24 Проверочное сравнение списка соединений логических элементов с эталонной моделью средствами моделирования

Цель — проверка функциональной эквивалентности синтезированного списка соединений логических элементов.

Описание. Моделирование списка соединений логических элементов, сгенерированного средствами синтеза. Используемые входные сигналы для проверки микросхемы моделированием точно соответствуют входным сигналам, примененным в процессе выполнения функционального тестирования на уровне модуля (Г.6) и функционального тестирования на верхнем уровне (Г.7) для функциональной проверки на уровне модуля и на верхнем уровне, соответственно. Функциональную проверку в основном следует осуществлять посредством анализа выходных сигналов микросхемы. Она может быть выполнена автоматически, посредством сравнения выходных сигналов микросхемы с соответствующей эталонной моделью или с исходным кодом схемы на языке (V)HDL. Этот тест известен как «регрессионный тест» и его выполнение более предпочтительно по сравнению с ручной проверкой выходных сигналов.

Примечание — Данный метод позволяет проверить функциональное поведение только для тех путей списка соединений логических элементов, которые фактически активны во время моделирования. Поэтому тестовый охват может быть только таким же, как и во время исходного функционального тестирования на уровне модуля или на верхнем уровне, соответственно. Можно дополнить этот метод формальным тестированием на эквивалентность. Во всех случаях функциональную проверку исходного кода на языке (V)HDL следует выполнять с окончательным списком соединений, сгенерированным средствами синтеза.

Г.25 Сравнение списка соединений логических элементов с эталонной моделью (формальный тест на эквивалентность)

Цель — независимая от моделирования проверка функциональной эквивалентности.

Описание. Осуществляют сравнение функциональности схемы, описанной в исходных кодах языка (V)HDL с функциональностью схемы, описанной списком соединений логических элементов, сгенерированным средствами синтеза. Средствами, в основе которых лежит принцип формальной эквивалентности, можно проверять функциональную эквивалентность схемы, описанной различными формами ее представления, например, на языке (V HDL или в виде ее списка соединений. В случае применения этого метода нет необходимости в функциональном моделировании, но независимая функциональная проверка возможна. Успешное применение этого метода может быть гарантировано, если полную эквивалентность можно доказывать только с используемым инструментом, а все сообщения о несоответствиях оцениваются автоматически или проверяют вручную.

Примечание — Данный метод выгодно объединить с методом Г.24 «Проверочное сравнение списка соединений логических элементов с эталонной моделью средствами моделирования». В любом случае функциональную проверку исходного кода на языке (V)HDL следует выполнять с окончательным списком соединений, сгенерированным средствами синтеза.

Г.26 Проверка требований и ограничений поставщика СИС

Цель — предотвращение отказов в процессе разработки посредством проверки требований поставщика.

Описание. Осуществление тщательной проверки требований и ограничений поставщика [например, минимальное и максимальное объединение и разветвление на входе и выходе, максимальная длина соединения (задержка в соединительной линии), максимальная крутизна фронта сигналов, расфазировка тактовых сигналов и т. д.] средствами синтеза улучшает надежность изделия. Требования поставщика к процессу разработки очень важны, их нарушение оказывает существенное влияние на обоснованность применяемых моделей, использующихся для моделирования. Поэтому любое нарушение требований и ограничений поставщика ведет к некорректным результатам моделирования, приводящим к нежелательной функциональности.

Г.27 Документальное оформление ограничений, результатов и средств синтеза

Цель — документальное оформление всех сформированных ограничений, которые необходимы для оптимального синтеза при генерации окончательного списка соединений логических элементов.

Описание. Документальное оформление всех ограничений и результатов синтеза необходимо:

- для воспроизводства синтеза на любой более поздней стадии;
- независимого генерирования результатов синтеза для проверки.

Важными документами являются:

- описание настроек синтеза, включая применяемые инструментальные средства и ПО синтеза с указанием фактических версий, используемые библиотеки синтеза, заданные ограничения и сценарии;
- журнал выполнения синтеза (лог-файл) с указанием времени, используемого средства с указанием версии и полная документация для синтеза;
- сгенерированный список соединений с оценкой времени задержки (файл в формате стандартного времени задержки — СВЗ-файл).

Г.28 Применение проверенных в эксплуатации средств синтеза

Цель — применение средства, выполняющего преобразование описания схемы на языке (V)HDL в списке соединений логических элементов.

Описание. Применяют средство, отображающее функциональность схемы, описанной на исходном коде языка (V)HDL, в соединения соответствующих логических элементов и примитивы схем из целевой библиотеки СИС. Выбор реализации из множества возможных реализаций, выполняющей требуемую функциональность, зависит от самого оптимального результата, который получен для ограничений синтеза, таких как синхронизация (тактовая частота) и площадь кристалла.

Г.29 Применение проверенной в эксплуатации целевой библиотеки

Примечание — См. также Г.4 приложения Г.

Цель — предотвращение систематических отказов, вызванных ошибками в целевой библиотеке.

Описание. Целевые библиотеки синтеза и моделирования для разработки СИС формируют из общей базы данных и поэтому они не являются независимыми. Типичными примерами систематических отказов являются:

- неоднозначность между реальным и промоделированным поведением элементов схемы;
- некорректное моделирование, например, времени установки и времени хранения.

Поэтому при проектировании СИС, выполняющих функции безопасности, следует использовать только «проверенные в эксплуатации» технологии и целевые библиотеки. Это означает:

- применение целевых библиотек, которые в течение продолжительного времени использовались в проектах с сопоставимой сложностью и тактовой частотой;
- наличие доступности технологии и соответствующей целевой библиотеки в течение достаточно длительного периода времени; при этом можно считать, что библиотека обеспечивает достаточную точность моделирования.

Г.30 Процедуры, основанные на сценарии

Цель — обеспечение воспроизводимости результатов контроля и автоматизации циклов синтеза.

Описание. Это процедуры автоматического и основанного на сценарии управления циклами синтеза, включая определение применяемых ограничений. Помимо получения точной документации на все ограничения синтеза данное управление циклами помогает воспроизвести список соединений после изменения исходного кода на языке (V)HDL при идентичных условиях.

Г.31 Реализация тестовых структур

Цель — проектирование тестируемых СИС, гарантирующих заключительное испытание.

Описание. Это проектирование тестируемости с помощью создания различных тестовых структур, обеспечивающее создание микросхем, которые легко тестируются, например:

- сканирование пути. В этом методе либо все триггеры (полное сканирование проекта) или их часть (частичное сканирование проекта) соединены в единую цепь или несколько цепей, создающих цепочку сдвиговых регистров. Метод сканирования пути автоматически генерирует тестовый пример для всей логики схемы. Инструментальное средство, генерирующее тестовый пример, называют «автоматическим генератором тестовых при-

меров» — АГТП. Реализация метода сканирования пути существенно улучшает тестируемость схемы и позволяет получить более 98 % тестового охвата при разумных усилиях. Поэтому рекомендуется реализовывать полное сканирование, если это возможно;

- НЕ-И-дерево. В НЕ-И-дереве все основные входы схемы соединены каскадно и создают цепочку. Применяя подходящий тестовый пример («блуждающий бит»), можно протестировать поведение схемы при переключении (синхронизацию и уровень запуска). НЕ-И-дерево является средством, непосредственно характеризующим первичные входы. Его рекомендуется применять, если поведение схемы при переключении не может быть протестировано иначе;

- встроенное самотестирование ВСТ. Самотестирование схемы и в особенности самотестирование встроенного модуля памяти может оказаться очень эффективным, если на кристалле реализовать АГТП. АГТП выполняет автоматическую проверку структуры схемы с применением псевдослучайных тестовых примеров и оцениванием сигнатуры реализованной структуры схемы. АГТП рекомендуется как дополнительная мера, особенно для тестирования модуля памяти. Тестирование «сканирование пути» может быть заменено на АГТП;

- тестирование статического тока потребления СТП интегральной микросхемы (СТП-тестирование). Статическая КМОП-микросхема потребляет ток, главным образом, во время переключения. Поэтому абсолютно исправная микросхема потребляет очень небольшое значение тока (потребляемый ток менее 1 мкА) до тех пор, пока не запускается тестовый пример. СТП-тестирование очень эффективно и обеспечивает тестовый охват более 50 % сразу же после применения нескольких тестов. СТП-тестирование может быть применено на функциональных тестовых примерах, так же, как и на синтезированных тестовых примерах, сгенерированных средствами АГТП. Этот метод тестирования на практике оказался самым полезным и позволяющим обнаруживать отказы, которые редко обнаруживались или вовсе не обнаруживались при использовании других текстов. Поэтому данный метод следует применять при регулярных производственных испытаниях;

- граничное сканирование. При этом методе тестовая архитектура для проверки соединений компонентов на печатной плате реализуется в соответствии со стандартом JTAG [178]. Тот же самый подход может быть применен для проверки соединений модулей на уровне кристалла. Граничное сканирование в первую очередь рекомендуется для улучшения тестируемости печатных плат.

Г.32 Оценка тестового охвата посредством моделирования

Цель — определение достигаемого тестового охвата, реализованного с применением архитектуры встроенного тестирования во время испытаний изделия.

Описание. Тестовый охват, достигаемый с применением встроенного тестирования ВСТ, функционального тестового примера или любого иного метода, может быть определен посредством моделирования отказа. Во время моделирования отказа тестовый пример применяют к цепи, в которую введен отказ. Реакция цепи на отказ при запуске тестового примера соответствует введенному отказу и, таким образом, вносит вклад в тестовый охват. Моделирование отказа позволяет обнаружить константные отказы, «константную 1» и «константный 0», а достигаемый тестовый охват отражает качество примененного тестового примера. Моделирование отказа может быть использовано для очень эффективного обнаружения отказов, связанных с логическими элементами, которые не охватываются тестом «сканирование пути», например, в случае частичного сканирования.

Г.33 Оценка тестового охвата средствами автоматической генерации тестовых примеров АГТП

Цель — определение ожидаемого тестового охвата для синтезируемых тестовых примеров («сканирование пути», ВСТ) в процессе производственных испытаний.

Описание. В настоящее время существуют разные процедуры, в которых генерируют псевдослучайные или алгоритмические тестовые примеры для схемы, реализованные в методе «Сканирование пути». Средства синтеза, такие как АГТП, создают в процессе синтеза каталог невыявленных отказов. Таким способом можно оценить тестовый охват и определить нижний предел достигаемого тестового охвата для применяемого тестового примера. Следует учитывать, что тестовый охват ограничен логическими элементами схемы, которая охвачена методом «сканирование пути». Модули, такие как устройство памяти, ВСТ или часть схемы, которые оказались не охваченными методом «сканирование пути», не рассматривают при оценке тестового охвата.

Г.34 Обоснование проверенных в эксплуатации жестких соединений

Цель — предотвращение систематических отказов жестких соединений в СИС.

Описание. Жесткое соединение обычно рассматривают как «черный ящик», который обеспечивает выполнение требуемой функции, содержит базу данных компоновки соединения в целевой технологии и который поддерживает требуемый компонент схемы. Возможный функциональный отказ может трактоваться по аналогии с отказами дискретных компонентов, таких как стандартные микропроцессоры, модуль памяти и т. д. Эксплуатация таких жестких соединений без проверки правильности функционирования возможна, если можно считать доказанным, что для применяемой заданной технологии используемое соединение является проверенным в эксплуатации компонентом. При этом остальная часть схемы должна быть тщательно проверена.

Г.35 Применение проверенных жестких соединений

Цель — предотвращение систематических отказов при применении жестких соединений.

Примечание — См. также Г.6 приложения Г.

Описание. Проверка соединения на соответствие должна осуществляться производителем (поставщиком) из-за сложного характера соединений и предполагаемых ограничений на этапе проектирования на основе исходного кода (V)HDL. Проверка может быть оправдана только для конкретной конфигурации и целевой технологии применяемого компонента.

Г.36 Тестирование жестких соединений в сети

Цель — предотвращение систематических отказов жестких соединений.

Примечание — См. также Г.13 приложения Г.

Описание. Осуществляют проверку правильности функции и реализации использованных жестких соединений посредством применения сетевых тестов. При применении данной меры должны быть документально оформлены описание основной концепции тестирования и результаты оценки основной концепции.

Г.37 Проверка правил проектирования (DRC)

Цель — проверка правил проектирования схемы производителя (поставщика) изделия.

Описание. Проверка сгенерированного макета в части соблюдения правил проектирования производителя (поставщика), например, минимальная длина проводника, максимальная длина проводника и несколько правил размещения структур макета. Полное и правильное выполнение правил проектирования схемы должно быть подробно документально оформлено.

Г.38 Проверка соответствия топологии макета схеме

Цель — независимая проверка соответствия топологии макета его схеме (LVS).

Описание. При проверке соответствия топологии макета его схеме функциональность схемы формируется из базы данных компоновки элементов, и осуществляется сравнение выделенных элементов схемы, включая соединения, с входным списком соединений. Этот процесс гарантирует достоверность проверки эквивалентности размещения элементов схемы со списком соединений, определяющим функциональность схемы. Полное и корректное выполнение проверки соответствия топологии макета его схеме должно быть подробно документально оформлено.

Г.39 Дополнительный запас (более 20 %) для технологических процессов, применяемых менее трех лет

Цель — обеспечение надежности реализованной функции схемы даже при сильном колебании процесса и параметра.

Описание. Фактическое поведение схемы определяется количеством перекрывающихся физических эффектов, особенно для небольших структур (например, менее 0,5 мкм). На самом деле, из-за отсутствия подробных знаний и необходимых упрощений не может быть получена точная модель элементов схемы. С уменьшением геометрических размеров задержка сигналов в линиях передачи играет все более доминирующую роль. Задержка сигнала вдоль провода и перекрестные связи между проводами растут нелинейно. Задержка сигнала уже не являются незначительными по сравнению с задержкой в электронном ключе. Оценочные задержки в линиях показывают увеличение риска с уменьшением геометрических размеров.

Поэтому рекомендуется планировать достаточный запас (более 20 %) в отношении минимальных и максимальных временных ограничений для схем, разработанных с использованием процессов, применяемых менее трех лет, чтобы гарантировать правильное выполнение функций схемы при наличии сильно изменяющихся параметров в период производства или из-за отсутствия средств точного моделирования.

Г.40 Тестирование на выжигание

Цель — обеспечение надежности производимого кристалла микросхемы; отбраковка ранних дефектов. Чистые кристаллы не проверяют на термопрочность, например, с помощью стрессовых методов на уровне пластины.

Описание. Испытание на выжигание проводят при максимально допустимой рабочей температуре (как правило, 125 °С). Продолжительность испытания зависит от целевого УПБ или от конкретных рекомендаций по выжиганию, например, производителя СИС.

Выжигание дефектов может быть использовано:

- для отбраковки дефектов на ранних стадиях (уменьшение интенсивности отказов в начале корытообразной кривой распределения отказов);
- доказательства, что отказы на ранних стадиях уже устранены в процессе производства и тестирования (то есть что устройства, вышедшие из производства, уже находятся в области постоянной интенсивности отказов корытообразной кривой).

Г.41 Применение серийных устройств, проверенных в эксплуатации

Цель — обеспечение безотказности изготовленных кристаллов.

Описание. У разработчика и производителя систем безопасности («промышленного производства») должен быть достаточный опыт применения технологии программируемых устройств, а также средств разработки.

Г.42 Процесс производства, проверенный в эксплуатации

Цель — обеспечение безотказности изготовленных кристаллов.

Описание. Проверенный в эксплуатации процесс производства характеризуется достаточно высоким качеством серийного производства.

Г.43 Контроль качества производственного процесса

Меры по управлению качеством и механизмы управления процессом производства устройства гарантируют непрерывное управление процессом. Например, оптическое или электрическое управление тестовыми структурами, температурные испытания с изменением параметров влажности или циклический температурный тест.

Подробное описание данного метода/средства приведено в [74], [179].

Г.44 Передача качественно изготовленного устройства

Качество устройства обеспечивается выполнением выбранной группы стресс-тестов, например, температурными испытаниями с изменением параметров влажности или испытаниями с изменением температуры. Производитель устройства должен предоставлять такие доказательства.

Подробное описание данного метода/средства приведено в [74], [179].

Г.45 Передача качественно функционирующего устройства

Все устройства тестируют на функциональность. Производитель устройства должен предоставлять доказательства этому.

Г.46 Стандарты качества

Производитель СИС должен предусмотреть достаточный уровень управления качеством, например, иметь оформленное «Руководство по качеству и надежности», в котором документально подтверждено выполнение сертификации по ГОСТ ISO 9000 или стандартной оценки качества поставщика (SSQA).

Приложение Д
(справочное)

Методы и средства для разработки связанного с безопасностью объектно-ориентированного программного обеспечения

Все рекомендации настоящего стандарта по проектированию ПО применяются к объектно-ориентированному ПО. Поскольку при объектно-ориентированном подходе информация представляется иначе, чем при процедурном или функциональном подходе, требуется особое рассмотрение следующих рекомендаций:

- понимание разработчиками иерархий классов и идентификации функции(ий) ПО, которые будут выполняться при вызове заданного метода (включая существующую библиотеку классов, если она используется);
- применение структурного тестирования [см. ГОСТ 34332.4—2021 (таблица Б.2) и приложение В настоящего стандарта).

Таблицы Д.1 и Д.2 содержат справочные руководящие указания по использованию объектно-ориентированного ПО, которые дополняют более общие нормативные руководящие указания.

Таблица Д.1 — Архитектура объектно-ориентированного ПО

Рекомендации	Подробности	УПБ 1	УПБ 2	УПБ 3	УПБ 4
1 Прослеживаемость понятия прикладной области с классами архитектуры	Примечание 1	Р	ОР	ОР	ОР
2 Использование подходящих фреймов, общеиспользуемых комбинаций классов и шаблонов проектирования. Примечание — При использовании существующих фреймов и шаблонов проектирования к ним применяются требования, как и к предварительно разработанному ПО	Примечание 2	Р	ОР	ОР	ОР

Примечания

1 Прослеживаемость прикладной области с архитектурой класса менее важна.

Примеры

1 Для некоторой части, предназначенной для проекта, связанного с безопасностью, может существовать фрейм из проекта, не связанного с безопасностью, который успешно решает подобную задачу, и это хорошо известно участникам проекта. В этом случае рекомендуется использовать такой фрейм.

2 Могут понадобиться различные алгоритмы для решения тесно связанных подзадач проекта, связанного с безопасностью. Поэтому может быть выбран шаблон стратегии доступа к алгоритмам.

3 Часть проекта, связанного с безопасностью, может состоять из выдачи надлежащих предупреждений внутренним и внешним заинтересованным сторонам. Для организации этих предупреждений может быть выбран шаблон наблюдателя. Это требование не применяется для библиотек.

2 Как правило, абстрактный базовый класс обеспечивает доступ к производным конкретным классам.

Таблица Д.2 — Объектно-ориентированный рабочий проект

Рекомендации	УПБ 1	УПБ 2	УПБ 3	УПБ 4
1 Классы должны иметь только одну цель	Р	Р	ОР	ОР
2 Использование наследования при условии, что производный класс является уточнением своего основного класса	ОР	ОР	ОР	ОР
3 Ограничение глубины наследования стандартами кодирования	Р	Р	ОР	ОР
4 Переопределение операций (методов) строго контролируется	Р	ОР	ОР	ОР
5 Использование множественного наследования только для интерфейсных классов	ОР	ОР	ОР	ОР
6 Наследование от неизвестных классов	- -	- -	НР	НР

Окончание таблицы Д.2

Рекомендации	УПБ 1	УПБ 2	УПБ 3	УПБ 4
7 Подтверждение того, что повторно используемые объектно-ориентированные библиотеки отвечают рекомендациям данной таблицы	ОР	ОР	ОР	ОР
Примечания 1 Один класс характеризуется наличием одной ответственности — необходимо быть внимательным при описании тесно связанных данных и операций над этими данными. 2 Необходимо внимательно следить за тем, чтобы не допустить циклических зависимостей между объектами.				

В таблице Д.3 приведены неформальные определения использованных выше терминов.

Таблица Д.3 — Некоторые объектно-ориентированные термины

Термин	Определение
Основной класс	Класс, у которого есть производные классы. Примечание — Основной класс иногда называют надклассом, или родительским классом
Переопределение	Замена операции (метода, подпрограммы) другой операцией (методом, подпрограммой) с той же самой сигнатурой и иерархией наследования во время выполнения; свойство объектно-ориентированных языков или программ, реализующее полиморфизм
Производный класс	Класс (совокупность атрибутов и операций), который наследовал атрибуты и/или операции от другого класса (основного класса). Производный класс иногда называют подклассом, или дочерним классом
Сигнатура операции (подпрограммы), (метода)	Имя операции (подпрограммы), (метода) вместе с ее параметрами (аргументами) и их типы, иногда также их возвращаемые типы. Примечание — Две сигнатуры равны, если у них одни и те же имена, число и типы параметров; в некоторых языках должны быть одинаковыми также возвращаемые типы
Фрейм	Структура программы, обычно предварительно разработанная для заполнения данными конкретного применения

Приложение Е
(справочное)

Вероятностный подход к определению полноты безопасности предварительно разработанного программного обеспечения

Е.1 Общие положения

Настоящее приложение содержит исходные руководящие материалы по использованию вероятностного подхода к определению полноты безопасности программных средств для предварительно разработанных программ на основе опыта их эксплуатации. Вероятностный подход является наиболее подходящим для оценки операционных систем, компонентов библиотек, компиляторов и других программных систем. Настоящее приложение содержит краткое описание возможностей вероятностного подхода, а также ссылки на источники с более подробным их описанием. Настоящее приложение не следует рассматривать как полное или исчерпывающее. Его следует использовать только тем специалистам, кто компетентен в статистическом анализе.

Предложенные в настоящем приложении методы могут быть также использованы для демонстрации роста уровня полноты безопасности ПО, которое некоторое время успешно эксплуатировалось. Например, ПО, созданное в соответствии с требованиями ГОСТ 34332.4 для УПБ 1, после соответствующего периода успешной работы в большом числе применений может быть пригодным для демонстрации соответствия уровню полноты безопасности УПБ 2.

Число запросов без отказов при испытании или число часов, необходимое для работы без отказов, для определения конкретного уровня полноты безопасности представлено в таблице Е.1. В таблице Е.1 также обобщены результаты, приведенные в Е.2.1 и Е.2.3.

Опыт эксплуатации может быть выражен математически, как показано в Е.2, для дополнения или замены статистического тестирования, а опыт эксплуатации, полученный из нескольких мест эксплуатации, может быть объединен (путем добавления конкретного числа обработанных запросов или часов работы в период эксплуатации), но только в случае, если:

- программная версия, подлежащая использованию в Э/Э/ПЭ СБЗС системе будет идентична версии, для которой предъявлен результат опыта ее эксплуатации;
- эксплуатационный профиль входного пространства аналогичен;
- существует эффективная система уведомлений и документирования отказов;
- справедливы принятые в Е.2 предположения.

Таблица Е.1 — Необходимая предыстория для определения уровня полноты безопасности

УПБ	Режим работы с низкой интенсивностью запросов (вероятность отказа при выполнении планируемых функций по запросу)	Число реальных запросов		Режим с высокой интенсивностью запросов или с непрерывным запросом (вероятность опасного отказа в час)	Общее число часов эксплуатации	
		$1 - \alpha = 0{,}99$	$1 - \alpha = 0{,}95$		$1 - \alpha = 0{,}99$	$1 - \alpha = 0{,}95$
УПБ 4	От 10^{-5} до 10^{-4}	$4{,}6 \cdot 10^5$	$3 \cdot 10^5$	От 10^{-9} до 10^{-8}	$4{,}6 \cdot 10^9$	$3 \cdot 10^9$
УПБ 3	От 10^{-4} до 10^{-3}	$4{,}6 \cdot 10^4$	$3 \cdot 10^4$	От 10^{-7} до 10^{-6}	$4{,}6 \cdot 10^8$	$3 \cdot 10^8$
УПБ 2	От 10^{-3} до 10^{-2}	$4{,}6 \cdot 10^3$	$3 \cdot 10^3$	От 10^{-6} до 10^{-5}	$4{,}6 \cdot 10^7$	$3 \cdot 10^7$
УПБ 1	От 10^{-2} до 10^{-1}	$4{,}6 \cdot 10^2$	$3 \cdot 10^2$	От 10^{-5} до 10^{-4}	$4{,}6 \cdot 10^6$	$3 \cdot 10^6$

Примечания

1 Величина $1 - \alpha$ представляет собой уровень доверия.

2 Предпосылки и описание процедур получения числовых значений в настоящей таблице см. в Е.2.1 и Е.2.3.

Подробное описание данного метода/средства приведено в [75], [84].

Е.2 Формулы статистического тестирования и примеры их использования

Е.2.1 Простой статистический тест для режима работы с низкой интенсивностью запросов

Е.2.1.1 Исходные предпосылки:

- распределение тестовых данных равно распределению запросов при выполнении операций в режиме онлайн;
- прохождения тестов статистически не зависят друг от друга в отношении причины отказа;
- для обнаружения любых отказов, которые могут появиться, существует адекватный механизм;
- число тестовых запросов n >100;
- во время прогона n тестовых запросов отказы отсутствуют.

Е.2.1.2 Результаты

Вероятность отказа p (на один запрос) при уровне доверия $1 - \alpha$ определяется из выражения:

$$p \le 1 - \sqrt[n]{\alpha} \text{ или } n \ge -\frac{\ln \alpha}{p}.$$

Е.2.1.3 ***Пример***

Таблица Е.2 — Вероятности отказа режима работы с низкой интенсивностью запросов

1 – α	*p*
0,95	*3/n*
0,99	*4,6/n*

Для вероятности отказа при запросе для УПБ 3 при уровне доверия 95 % применение указанной формулы дает 30000 тестовых примеров при выполнении условий принятых предпосылок. Результаты для каждого УПБ объединены в таблице Е.1.

Е.2.2 Тестирование входного массива (предметной области) для режима работы с низкой интенсивностью запросов

Е.2.2.1 Исходные предпосылки

Единственная исходная предпосылка состоит в том, что тестируемые данные выбирают так, чтобы обеспечить случайное унифицированное распределение по входному массиву (предметной области).

Е.2.2.2 Результаты

Необходимо определить число тестов n, которые требуются, исходя из порога точности δ входов для тестируемой функции с низкой интенсивностью запросов (например, безопасное отключение).

Таблица Е.3 — Средние расстояния между двумя точками тестирования

Размер предметного пространства	Среднее расстояние между двумя точками тестирования в произвольном направлении
1	$\delta = 1/n$
2	$\delta = \sqrt[2]{1/n}$
3	$\delta = \sqrt[3]{1/n}$
k	$\delta = \sqrt[k]{1/n}$
Примечание — k может быть любым положительным целым числом. Значения 1, 2 и 3 приведены только в качестве примеров.	

Е.2.2.3 ***Пример***

Рассмотрим безопасное отключение, которое зависит только от двух переменных A и B. Если проверкой было установлено, что пороговые значения, которые разделяют входную пару переменных A и B, определены с точностью до 1 % от диапазона измерения A или B, то число равномерно распределенных тестовых запросов, требуемых в области A и B, будет равно $n = 1/\delta^2 = 10^4$.

Е.2.3 Простой статистический тест для режима с высокой интенсивностью запросов или в режиме с непрерывным запросом

Е.2.3.1 Исходные предпосылки:

- распределение данных такое же, как и распределение при выполнении операций в режиме онлайн;
- относительное уменьшение вероятности отсутствия отказа пропорционально продолжительности рассматриваемого интервала времени и постоянно в противном случае;
- для обнаружения любых отказов, которые могут появиться, существует адекватный механизм;
- тест выполняется в течение времени тестирования t;
- во время тестирования t никаких отказов не происходит.

Е.2.3.2 Результаты

Соотношение между интенсивностью отказов λ, уровнем доверия $1 - \alpha$ и временем тестирования t имеет вид:

$$\lambda = \frac{\ln \alpha}{t}.$$

Интенсивность отказов обратно пропорциональна среднему времени наработки на отказ (MTBF):

$$\lambda = \frac{1}{\text{MTBF}}.$$

Примечание — В настоящем стандарте не делается различий между интенсивностью отказов в час и частотой отказов в час. Вероятность отказа F связана с частотой отказов f выражением

$$F = 1 - e^{-tt},$$

однако область применения настоящего стандарта охватывает частоту отказов менее 10^{-5} 1/ч, а для небольших значений частоты справедливо $F \cong f \cdot t$.

Е.2.3.3 ***Пример***

Таблица Е.4 — Вероятности отказа для режима с высокой интенсивностью запросов или для режима с непрерывным запросом

$1 - \alpha$	λ
0,95	***3/t***
0,99	***4,6/t***

Для подтверждения того, что среднее время наработки на отказ составляет по меньшей мере 10^8 ч с уровнем доверия 95 %, требуется время тестирования $3 \cdot 10^8$ ч и должны быть соблюдены исходные предпосылки. Число тестов, необходимое для каждого УПБ, — в соответствии с таблицей Г.1.

Е.2.4 Полное тестирование

Программу можно рассматривать как урну, содержащую N шаров. Каждый шар представляет собой конкретное свойство программы. Шары извлекают случайно и заменяют после проверки. Полное тестирование достигается, если все шары извлечены.

Е.2.4.1 Исходные предпосылки:

- распределение тестируемых данных таково, что каждое из N свойств программы тестируется с равной вероятностью;
- тесты проводят независимо друг от друга;
- каждый появляющийся отказ обнаруживается;
- число тестовых примеров n много больше N;
- во время прогона n тестовых запросов отказы не появляются;
- каждым прогоном теста контролируется одно свойство программы (свойство программы — это то, что может быть протестировано во время одного прогона теста).

Е.2.4.2 Результаты

Вероятность тестирования всех свойств программы p определяется выражением:

$$p = \sum_{j=0}^{N-1} (-1)^j \left(\frac{N}{j}\right)\left(\frac{N-j}{N}\right)^n$$

или

$$p = 1 + \sum_{j}^{N} (-1)^j C_{j,N} \left(\frac{N-j}{N}\right)^n,$$

где $C_{j,N} = (N\,(N-1)\ldots\,(N-j))/j!$

При оценке этого выражения обычно только первые его члены имеют значение, поскольку в реальных условиях выполняется соотношение n много больше N, что делает все члены этого выражения при большом j несущественными. Это видно из таблицы Г.5.

Пример

Рассмотрим программу, которая имела несколько инсталляций в течение нескольких лет. За это время она выполнялась по меньшей мере $7,5 \cdot 10^6$ раз. Предположим, что каждое сотое выполнение программы соответствует перечисленным выше исходным предпосылкам (см. Е.2.4.1). Поэтому для статистической оценки могут быть приняты $7,5 \cdot 10^4$ выполнений программы. Если предположить, что 4000 тестовых прохождений программы могут выполнить исчерпывающее тестирование, считая такую оценку консервативной, то в соответствии с таблицей Г.5, вероятность того, что не все будет протестировано, составляет $2,87 \cdot 10^{-5}$.

При N = 4000 значения первых членов в зависимости от n представлены в таблице Е.5.

Таблица Е.5 — Вероятность тестирования всех свойств программы

n	*p*
$5 \cdot 10$	$1 - 1{,}9 \cdot 10^{-2} + 1{,}10 \cdot 10^{-4} - \ldots$
$7{,}5 \cdot 10$	$1 - 2{,}87 \cdot 10^{-5} + 4 \cdot 10^{-10} - \ldots$
$1 \cdot 10$	$1 - 5{,}54 \cdot 10^{-8} + 1{,}52 \cdot 10^{-15} - \ldots$
$2 \cdot 10$	$1 - 7{,}67 \cdot 10^{-19} + 2{,}9 \cdot 10^{-37} - \ldots$

На практике такие оценки должны быть консервативными.

Подробное описание данного метода/средства приведено в [179]—[182].

Приложение Ж
(справочное)

Определение свойств стадий жизненного цикла программного обеспечения

Таблица Ж.1 — Спецификация требований к ПО Э/Э/ПЭ СБЗС системы [см. ГОСТ 34332.4—2021 (подраздел 7.2 и приложение В, таблица В.1)]

Свойство	Определение
1 Полнота охвата потребностей безопасности программным обеспечением	Спецификация требований к ПО Э/Э/ПЭ СБЗС системы охватывает все потребности и ограничения системы, появившиеся на более ранних стадиях ЖЦ системы и определенные для ПО. Потребности и ограничения Э/Э/ПЭ СБЗС системы, как правило, устанавливают перед началом разработки спецификации требований к ПО Э/Э/ПЭ СБЗС системы. Они могут включать в себя спецификацию того, что не должно быть выполнено ПО или чего следует избегать
2 Корректность охвата потребностей безопасности программным обеспечением	Спецификация требований к ПО Э/Э/ПЭ СБЗС системы адекватно отвечает потребностям и ограничениям системы, которые были определены для ПО. Целью является обеспечение уверенности в том, что все, что определено в спецификации, действительно гарантирует безопасность для всех необходимых условий
3 Отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности	Внутренняя полнота и согласованность спецификации требований к ПО Э/Э/ПЭ СБЗС системы: предоставление всей необходимой информации для всех функций и ситуаций, которая должна быть получена из спецификации; отсутствие в ней противоречивых или несогласованных положений. Противоречие полноте и согласованности потребностям Э/Э/ПЭ СБЗС системы, внутренней полноте и согласованности может быть оценено только на основе спецификации требований к ПО Э/Э/ПЭ СБЗС системы
4 Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным ПО системы безопасности	Спецификация требований к ПО Э/Э/ПЭ СБЗС системы не содержит требований, которые не нужны для обеспечения безопасности УО. Цель состоит в том, чтобы избежать ненужной сложности при разработке и внедрении ПО, чтобы уменьшить риск отказов и избежать функций, не важных для безопасности, но мешающих или угрожающих функциям, которые важны для безопасности
5 Способность обеспечения проведения оценки и подтверждения соответствия	Спецификация требований к ПО Э/Э/ПЭ СБЗС системы является первоисточником необходимой информации для выполнения тестов и исследований, которые в результате их выполнения формируют объективное подтверждение того, что ПО удовлетворяет спецификации требований к ПО Э/Э/ПЭ СБЗС системы

Таблица Ж.2 — Проектирование и разработка ПО — проектирование архитектуры программ [см. ГОСТ 34332.4—2021 (пункт 7.4.3 и приложение В, таблица В.2)]

Свойство	Определение
1 Полнота спецификации требований к ПО системы безопасности	Проект архитектуры ПО охватывает все потребности и ограничения Э/Э/ПЭ СБЗС системы, появившиеся в спецификации требований к ПО Э/Э/ПЭ СБЗС системы
2 Корректность спецификации требований к ПО системы безопасности	Проект архитектуры ПО адекватно соответствует спецификации требований к ПО Э/Э/ПЭ СБЗС системы
3 Отсутствие собственных ошибок проекта	Проект архитектуры ПО и проектная документация не имеют ошибок, которые могут быть идентифицированы, независимо от любого указанного требования к ПО Э/Э/ПЭ СБЗС системы. Примеры ошибок: зависания, доступ к несанкционированным ресурсам, утечки ресурсов, внутренняя неполнота (то есть ошибки при обращении ко всем ситуациям, которые непосредственно установлены в проекте)

Окончание таблицы Ж.2

Свойство	Определение
4 Простота и понятность. Предсказуемость поведения	Проект архитектуры ПО, обеспечивающий корректный и точный прогноз функционирования ПО для всех указанных ситуаций. В частности, эти ситуации включают в себя ситуации с ошибками и с отказами. Предсказуемость подразумевает, в частности, что функционирование не зависит от элементов, не контролируемых разработчиками или пользователями
5 Верифицируемость и тестируемость проекта	Проект архитектуры ПО и проектная документация обеспечивают и облегчают формирование убедительного доказательства, что все заданные требования к ПО Э/Э/ПЭ СБЗС системы правильно учтены в проекте и что проект лишен внутренних ошибок. Верифицируемость может подразумевать получение таких свойств, как простота, модульность, ясность, тестируемость, доказуемость и т. д., в зависимости от используемых методов верификации
6 Отказоустойчивость	Проект архитектуры ПО дает уверенность в безопасности при наличии ошибок (внутренних ошибок, ошибок операторов или внешних систем). Можно спроектировать активную или пассивную защиту. Проекты активной защиты могут включать в себя такие функции, как обнаружение, создание сообщений и локализацию ошибок, постепенный вывод из эксплуатации и освобождение от любых нежелательных побочных эффектов до возобновления нормального функционирования. Проекты пассивной защиты включают в себя функции, которые гарантируют непроникновение определенных типов ошибок или определенных условий (лавинообразных потоков на входах, особенно дат и времени) без привлечения ПО
7 Защита от отказов по общей причине, вызванной внешним событием	Проект архитектуры ПО облегчает идентификацию видов отказов по общей причине и эффективных средств предостережения от отказов

Таблица Ж.3 — Проектирование и разработка ПО: инструментальные средства поддержки и языки программирования [см. ГОСТ 34332.4—2021 (пункт 7.4.4 и приложение В, таблица В.3)]

Свойство	Определение
1 Поддержка разработки ПО с требуемыми свойствами ПО	Средства, обеспечивающие обнаружение ошибок или устранение склонных к ошибкам структур
2 Четкость работы и функциональность инструментальных средств	Обеспечение всестороннего охвата и ответной реакции для всех аспектов работы инструментальных средств
3 Корректность и воспроизводимость результата	Непротиворечивость и точность результата работы инструментального средства для любого входного задания

Таблица Ж.4 — Проектирование и разработка ПО — детальное проектирование [см. ГОСТ 34332.4—2021 (пункт 7.4.5 и приложение В, таблица В.4)]

Свойство	Определение
1 Полнота спецификации требований к ПО системы безопасности	Приняты методы детального проектирования и разработки ПО, которые гарантируют, что полученное ПО охватывает все потребности и ограничения системы безопасности, сформированные для ПО
2 Корректность спецификации требований к ПО системы безопасности	Существует конкретное доказательство, позволяющее утверждать, что требования к Э/Э/ПЭ СБЗС системе, сформированные для ПО, выполняются разработанным ПО
3 Отсутствие собственных ошибок проекта	Разработанное ПО лишено внутренних отказов. Примеры отказов: зависание, доступ к несанкционированным ресурсам, утечки ресурсов
4 Простота и понятность. Предсказуемость поведения	Поведение разрабатываемого ПО предсказуемо объективным и убедительным тестированием и анализом

Окончание таблицы Ж.4

Свойство	Определение
5 Верифицируемость и тестируемость проекта	Разработанное ПО является поддающимся проверке и тестируемым
6 Отказоустойчивость/Обнаружение неисправностей	Методы и разработки обеспечивают гарантию того, что разработанное ПО будет вести себя безопасно при наличии ошибок
7 Отсутствие отказов по общей причине	Методы и проекты обеспечивают определение видов отказов по общей причине и обеспечивают эффективные средства предостережения от отказов ПО

Таблица Ж.5 — Проектирование и разработка ПО: тестирование и интеграция программных модулей [см. ГОСТ 34332.4—2021 (пункты 7.4.7, 7.4.8 и приложение В, таблица В.5)]

Свойство	Определение
1 Полнота тестирования и интеграции в соответствии со спецификациями проекта ПО	Тестирование ПО обеспечивает исследование поведения ПО с достаточной полнотой, чтобы гарантировать, что все требования спецификации проектирования ПО были удовлетворены
2 Корректность тестирования и интеграции в соответствии со спецификациями проекта ПО (успешное выполнение)	Если задача тестирования модуля завершена, то существует конкретное доказательство, позволяющее утверждать, что требования к Э/Э/ПЭ СБЗС системе были удовлетворены
3 Воспроизводимость	К согласованным результатам приводит повторение отдельных оценок, выполняемых как часть тестирования и интеграции модуля
4 Точно определенная тестируемая конфигурация	Для надлежащих версий модулей и ПО выполняют тестирование и интеграцию модуля, полученный требуемый результат связывают с конкретной конфигурацией «как построено» ПО

Таблица Ж.6 — Интеграция программируемых электронных устройств (ПО и АС) [см. ГОСТ 34332.4—2021 (подраздел 7.5 и приложение В, таблица В.6)]

Свойство	Определение
1 Полнота интеграции в соответствии со спецификациями проекта	Интеграция обеспечивает соответствующую глубину и охват элементов системы, чтобы продемонстрировать, что система может выполнить функции, для которых она предназначена, и не выполняет непредусмотренные функции при всех возможных условиях эксплуатации и при отказе системы. Интеграция включает в себя принципы, используемые для проверки, целевые уровни проекта и аспекты интеграции (например, проверку полноты взаимодействия между модулями)
2 Корректность интеграции в соответствии со спецификациями проекта (успешное выполнение)	Интеграция основана на корректных предположениях. Например, правильность ожидаемых результатов, рассматриваемых условий использования, а также репрезентативность тестовых сред. Если задача интеграции завершена, то существует конкретное доказательство, позволяющее утверждать, что требования к безопасности были удовлетворены
3 Воспроизводимость	К согласованным результатам приводит повторение отдельных оценок, выполняемых, как часть интеграции модуля
4 Точно определенная конфигурация интеграции	Интеграция дает соответствующую гарантию, что она была эффективно применена в соответствии с документами к правильной версии элементов и ПО, а полученный требуемый результат связывают с конкретной конфигурацией «как построено» ПО

Таблица Ж.7 — Подтверждение соответствия для аспектов ПО Э/Э/ПЭ СБЗС системы [см. ГОСТ 34332.4—2021 (подраздел 7.7 и приложение В, таблица В.7)]

Свойство	Определение
1 Полнота подтверждения соответствия в соответствии со спецификацией проекта ПО	Подтверждение соответствия ПО охватывает все требования спецификации проектирования ПО
2 Корректность подтверждения соответствия в соответствии со спецификацией проекта ПО (успешное выполнение)	Если задача подтверждения соответствия ПО выполнена, то существует конкретное доказательство, позволяющее утверждать, что требования к безопасности были удовлетворены
3 Воспроизводимость	К согласованным результатам приводит повторение отдельных оценок, выполняемых как часть подтверждения соответствия ПО
4 Подтверждение соответствия точно определенной конфигурации	Четкое и краткое определение: - системы, - требований, - окружающей среды

Таблица Ж.8 — Модификация ПО [см. ГОСТ 34332.4—2021 (подраздел 7.8 и приложение В, таблица В.8)]

Свойство	Определение
1 Полнота модификации в соответствии с требованиями к модификации	Модификация была должным образом одобрена авторизованным персоналом, с соответствующим пониманием ее функционала, последствий для системы безопасности, а также технических и эксплуатационных последствий
2 Корректность модификации в соответствии с требованиями к модификации	Модификация достигает своих заданных целей
3 Отсутствие собственных ошибок проекта	Модификация не вносит новые систематические ошибки. Примеры ошибок: деление на ноль, выход индексов или указателей за границы своих значений, использование неинициализированных переменных
4 Предотвращение нежелательного поведения	Модификация не вносит какое-либо поведение, которое, согласно ограничениям, установленным в спецификации требований к ПО Э/Э/ПЭ СБЗС системы, должно быть предотвращено
5 Верифицируемость и тестируемость проекта	Проект ПО является таким, что влияние модификации полностью и всесторонне оценивается
6 Регрессионное тестирование и охват проверкой	Проект ПО является таким, что эффективное и полное регрессионное тестирование позволяет продемонстрировать, что ПО после модификации продолжает удовлетворять спецификации требований к ПО Э/Э/ПЭ СБЗС системы

Таблица Ж.9 — Верификация ПО Э/Э/ПЭ СБЗС системы [см. ГОСТ 34332.4—2021 (подраздел 7.9 и приложение В, таблица В.9)]

Свойство	Определение
1 Полнота верификации в соответствии с предыдущей стадией	Верификация способна установить, что ПО удовлетворяет всем соответствующим требованиям спецификации требований к ПО Э/Э/ПЭ СБЗС системы
2 Корректность верификации в соответствии с предыдущей стадией (успешное выполнение)	Если задача верификации ПО завершена, то существует конкретное доказательство, позволяющее утверждать, что требования к Э/Э/ПЭ СБЗС системе были удовлетворены
3 Воспроизводимость	К согласованным результатам приводит повторение отдельных оценок, выполняемых как часть верификации
4 Верификация точно определенной конфигурации	Для надлежащих версий элементов и ПО выполняют верификацию, полученный требуемый результат связывают с конкретной конфигурацией «как построено» ПО

Таблица Ж.10 — Оценка функциональной безопасности [см. ГОСТ 34332.4—2021 (раздел 8 и таблица В.10)]

Свойство	Определение
1 Полнота оценки функциональной безопасности в соответствии с настоящим стандартом	Оценка функциональной безопасности ПО формирует ясное утверждение о степени найденного соответствия, сделанных обоснованиях, мерах по устранению недостатков с рекомендуемыми сроками их устранения, полученные выводы и рекомендации по их принятию, квалифицированному принятию, или отклонению с указанием любых временных ограничений для этих рекомендаций
2 Корректность оценки функциональной безопасности в соответствии с проектными спецификациями (успешное выполнение)	Задача оценки функциональной безопасности ПО завершена, и существует конкретное доказательство, позволяющее утверждать, что требования к Э/Э/ПЭ СБЗС системе были удовлетворены
3 Доступное для анализа решение всех выявленных проблем	Существует ясное утверждение о том, в каком объеме были рассмотрены проблемы, возникающие во время оценки функциональной безопасности ПО
4 Возможность модификации оценки функциональной безопасности после изменения проекта без необходимости проведения серьезной переработки оценки	Результаты оценки функциональной безопасности ПО могут быть перерассчитаны, причем при повторной оценке функциональной безопасности частей ПО после их изменения не выполняется повторная оценка функциональной безопасности всего ПО, она лишь корректируется с учетом измененных частей
5 Воспроизводимость	Оценка функциональной безопасности выполняется как согласованный, запланированный и открытый процесс с конкретными персоналом и документами. Данный процесс реализует рассмотрение основания для выполнения оценок и решения, которые выполняют все те, на которых влияют эти решения, включая поставщиков системы, пользователей, специалистов по обслуживанию и руководство. Оценка функциональной безопасности позволяет независимому компетентному персоналу повторять отдельные оценки, выполненные как часть всей оценки
6 Своевременность	Оценка функциональной безопасности выполняется с соответствующей частотой, связанной со стадиями жизненного цикла ПО Э/Э/ПЭ СБЗС системы и, по крайней мере, до определения существующих опасностей. Также она обеспечивает своевременное создание отчетов о несоответствиях. Результаты тестов, проверок, исследований и т. д. фактически доступны, если они требуются в качестве входной информации для формирования решения об оценке
7 Точно определенная конфигурация	Результаты оценки функциональной безопасности ПО связывают с конкретной конфигурацией Э/Э/ПЭ СБЗС системы, для которой результаты оценки функциональной безопасности должны быть обоснованы

Библиография

[1] Технический регламент Таможенного союза ТР ТС 002/2011 О безопасности высокоскоростного железнодорожного транспорта

[2] Технический регламент Таможенного союза ТР ТС 003/2011 О безопасности инфраструктуры железнодорожного транспорта

[3] Технический регламент Таможенного союза ТР ТС 014/2011 Безопасность автомобильных дорог

[4] Ларионов А.М., Майоров С.А., Новиков Г.И. Вычислительные комплексы, системы и сети. Учебник для вузов. — Л.: Энергоатомиздат. Ленингр. отделение, 1987. — 288 с., ил.

[5] Денисенко В. В. Компьютерное управление технологическим процессом, экспериментом, оборудованием. М.: Горячая линия-Телеком, 2009. — 608 с., ил. ISBN: 978-5-9912-0060-8

[6] Жуков В.В., Лабковский М.Д. Регулировка электромеханических и радиотехнических приборов и систем: Учеб. пособие для сред., проф.-техн. училищ. М.: Высш. шк., 1984, 200 с., ил.

[7] Платунов А., Постников Н., Чистяков А. Механизм граничного сканирования в неоднородных микропроцессорных системах

[8] Грушвицкий Р., Ильин И., Михайлов М. Граничное сканирование высокоскоростных соединений. «Компоненты и технологии». № 11. 2006

[9] Каневский И.Н., Сальникова Е.Н. Неразрушающие методы контроля: Учебное пособие. — Владивосток: Изд-во ДВГТУ, 2007. — 243 с.

[10] Харченко В.С., Юрченко Ю.Б. Анализ структур отказоустойчивых бортовых комплексов при использовании электронных компонентов Industry.//Технология и конструирование в электронной аппаратуре. — 2003. — № 2. — С. 3—10

[11] Алгоритмы автоматического восстановления ИС. Лекция 17. Методы и алгоритмы автоматического восстановления ИС. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[12] Системы противоаварийной защиты. Интеллектуальный подход. Emerson/ gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[13] М. Вернер. Основы кодирования. Учебник для ВУЗов. Москва: Техносфера, 2004. — 288 с. ISBN 5-94836-019-9

[14] Корректирующие коды. gost.e-ecolog.ru (доступ 12.04.2021)

[15] Коды исправляющие ошибки. Питерсон У., Уэлдон Э. М. «Мир». 1976

[16] Блох Э. Л., Зяблов В. В. Линейные каскады коды. М.: Наука, 1982

[17] Модули памяти. Портал Мир цифровой техники. gost.e-ecolog.ru (доступ 12.04.2021)

[18] Калядин А. Отладчики микроконтроллеров и их применение в разработке микроконтроллерных приложений. Мир компьютерной электроники.//МКА — gost.e-ecolog.ru (доступ 12.04.2021)

[19] Попарная запись-считывание ячеек с помощью «бегущего» кода для тестирования памяти. gost.e-ecolog.ru search.ask.com/web?q=RAM+test+%22Abraam%22&chn=&doi=&geo=ru_RU&guid=&o=APN11910&p2=%5EET%5Ech00ru%5E&prt=&ver=&tpr=2&ts=1504602302196 (доступ 12.04.2021)

[20] R. Nair, S.M. Thatte, J.A. Abraham Эффективные алгоритмы тестирования полупроводниковых запоминающих устройств с произвольным доступом, IEEE Trans. Comput. C-27 (6), 572—576, 1978. gost.e-ecolog.ru citation.cfm?id=1310824 (доступ 12.04.2021)

[21] Харкевич А. А. Борьба с помехами. М.: Гос. изд-во физ.-мат. лит., 1983. 277 с.

[22] Р.В. Хемминг. Теория кодирования и теория информации. М.: Радио и связь, 1983, 176 с.

[23] Морелос-Сарагоса Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. М.: Техносфера, 2005

[24] Коды Хемминга. Ознакомление с принципами построения кодов Хэмминга. Новосибирский государственный технический университет. Портал Вунивере. gost.e-ecolog.ru (доступ 12.04.2021)

[25] Системы управления двигателем. Wiki Electro Schneider Electric. gost.e-ecolog.ru D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B4%D0%B2%D0%B8%D0%B3%D0%B0%D1%82%D0%B5%D0%BB%D0%B5%D0%BC (доступ 12.04.2021)

[26] Компания Backhoff. «Желтые» модули противоаварийной защиты работают по промышленной шине//Автоматизация в промышленности. Январь 2005, с. 36—38

[27] Мясникова Н.В., Панов А.П., Цыпин Б.В. Система для исследования характеристик датчиков динамического давления.//Измерение. Мониторинг. Управление. Контроль. — 2013. — № 4 (6). — С. 32—36

[28] Кирпичев А., Симчук А., Тищенко Ю. Датчики динамического давления: продукция компании «ГлобалТест».// Электроника: Наука — Технология — Бизнес. № 1/2008. С. 89—91

[29] Схемы реле. Преобразование полярности. gost.e-ecolog.ru (доступ 12.04.2021)

[30] Руководство по безопасной автоматизации химических процессов 2-е издание, Декабрь 2016, 648 с. ISBN: 978-1-118-94949-8. gost.e-ecolog.ru (доступ 12.04.2021)

[31] ISO 21500:2021, Project, programme and portfolio management — Context and concepts (Управление проектами, программами и портфелем ценных бумаг. Контекст и концепции)

[32] ISO/IEC 33001:2015, Information technology — Process assessment — Concepts and terminology (Информационные технологии. Оценка процесса. Понятия и терминология)

[33] ISO/IEC 33002:2015, Information technology — Process assessment — Requirements for performing process assessment (Информационная технология. Оценка процесса. Требования к проведению оценки процесса)

[34] ISO/IEC TS 33030:2017, Information technology — Process assessment — An exemplar documented assessment process (Информационная технология. Оценка процесса. Пример документированного процесса оценки)

[35] ISO/IEC TR 33015:2019, Information technology — Process assessment — Guidance for process risk determination (Информационная технология. Оценка процесса. Определение риска процесса)

[36] ISO/IEC 15504-5:2012, Information technology — Process assessment — Part 5: An exemplar software life cycle process assessment model (Информационные технологии. Оценка процессов. Часть 5. Образец модели оценки процессов жизненного цикла программного обеспечения)

[37] ISO/IEC 33004:2015, Information technology — Process assessment — Requirements for process reference, process assessment and maturity models (Информационные технологии. Оценка процесса. Требования к эталонным моделям процесса, моделям оценки процесса и моделям зрелости)

[38] IEC 61506:1997, Industrial-process measurement and control — Documentation of application software (Измерение и управление производственными процессами. Документация прикладного программного обеспечения)

[39] Формальная спецификация: Методы и применения. N. Nissanke, Springer-Verlag Telos, 1999, ISBN-10: 1852330023. gost.e-ecolog.ru (доступ 12.04.2021)

[40] IEC 61131-3:2013, Programmable controllers — Part 3: Programming languages (Контроллеры программируемые. Часть 3. Языки программирования)

[41] Лабораторная работа № 1. Методология объектно-ориентированного моделирования. Запорожская государственная инженерная академия. gost.e-ecolog.ru (доступ 12.04.2021)

[42] Харель Д. Статехартс: Визуальный формализм для сложных систем. Наука о компьютерном программировании. Изд-во Elsevier Science B.V. 1987. gost.e-ecolog.ru statecharts.pdf (доступ 12.04.2021)

[43] Джеффри, Джон Э. Хопкрофт, Раджив Мотвани. Введение в теорию автоматов, языков и вычислений. https://gost.e-ecolog.ru 12.04.2021)

[44] Трутнев Д.Р. Архитектуры информационных систем. Основы проектирования: Учебное пособие. — СПб.: НИУ ИТМО, 2012. — 66 с.

[45] ISO/IEC 15909-1:2019, Systems and software engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation (Системная и программная инженерия. Сети Петри высокого уровня. Часть 1. Понятия, определения и графические обозначения)

[46] ISO/IEC 15909-2:2011, Systems and software engineering — High-level Petri nets — Part 2: Transfer format (Разработка программного обеспечения и систем. Сети Петри высокого уровня. Часть 2. Формат передачи)

[47] Анализ требований к системе. Джеффри О. Грэди, Академическая пресса, 2006, ISBN 012088514X, 9780120885145. gost.e-ecolog.ru (доступ 12.04.2021)

[48] Требования к программному обеспечению: практические методы сбора и управления требованиями в течение всего цикла разработки изделия. Карл Юджин Вигерс, Microsoft Press, ISBN-13: 978-0735679665, ISBN-10: 0735679665. gost.e-ecolog.ru (доступ 12.04.2021)

[49] IEC 60880:2006, Nuclear power plants — Instrumentation and control systems important to safety — Software aspects for computer-based systems performing category A functions (Атомные электростанции. Системы контроля и управления, важные для безопасности. Программное обеспечение компьютерных систем, выполняющих функции категории А)

[50] Требования к программному обеспечению (3-е изд.) (рекомендации разработчика). ISBN-13: 978-0735679665, ISBN-10:0735679665. gost.e-ecolog.ru dp/0735679665 (доступ 12.04.2021)

[51] Обеспечение качества программного обеспечения: от теории до реализации. Даниэль Галин, Pearson Education, 2004, ISBN 0201709457, 9780201709452. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[52] IEC 81346-1:2009, Industrial systems, installations and equipment and industrial products — Structuring principles and reference designations — Part 1: Basic rules (Промышленные системы, установки, оборудование и промышленная продукция. Принципы структурирования и коды. Часть 1. Основные правила)

[53] IEC 81346-2:2019, Industrial systems, installations and equipment and industrial products — Structuring principles and reference designations — Part 2: Classification of objects and codes for classes (Промышленные системы, установки, оборудование и промышленная продукция. Принципы структурирования и кодированные обозначения. Часть 2. Классификация объектов и коды классов)

[54] ISO/TS 81346-10:2015, Industrial systems, installations and equipment and industrial products — Structuring principles and reference designation — Part 10: Power plants (Промышленные системы, установки, оборудование и промышленная продукция. Принципы структурирования и кодовые обозначения. Часть 10. Энергетические установки)

[55] IEC 61160:2005, Design review (Анализ проекта)

[56] Программное обеспечение Проектирование систем реального времени. J.E. Cooling, Pearson Education, 2003, ISBN 0201596202, 9780201596205. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[57] IEC 61163-1:2006, Reliability stress screening — Part 1: Repairable assemblies manufactured in lots (Сплошная проверка аппаратных элементов на надежность в напряженном состоянии. Часть 1. Подлежащие ремонту аппаратные элементы, изготовливаемые партиями)

[58] Искусство тестирования программ. Гленфорд Майерс, Том Баджетт, Кори Сандлер. Изд-во «Вильямс», 2012. 272 с. ISBN 978-5-8459-1796-6, 978-1-118-03196-4

[59] Разработка программного обеспечения: Обновление. Ян Соммервилл, Эддисон-Уэсли Лонгман, Амстердам; 8-е изд., 2006, ISBN 0321313798, 9780321313799. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[60] Разработка программного обеспечения (10-е изд.). Йен Соммервилл, Пирсон Студиум, 8. Ауфлаге, 2007, ISBN-13: 978-0133943030, ISBN-10: 0133943038. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[61] Практическое тестирование программного обеспечения: технологический подход (Springer Professional Computing) 2003 г. Ред. ISBN-13: 978-0387951317, ISBN-10: 0387951318. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021).

[62] Основные сведения о программном обеспечении. Франк Ф. Цуй, Орландо Карам. Джонс и Бартлетт, 2006. ISBN 076373537X, 9780763735371. gost.e-ecolog.ru dp/076373537X (доступ 12.04.2021)

[63] Систематическое тестирование программного обеспечения. Рик Д. Крейг, Стефан П. Яскиль. Artech House, 2002. ISBN 1580535089, 9781580535083. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[64] Практическое проектирование надежных систем. П.О'Коннор, Д. Ньютон, Р. Бромли, Джон Уайли и сыновья, 2002, ISBN 0470844639, 9780470844632. gost.e-ecolog.ru gost.e-ecolog.ru X (доступ 12.04.2021)

[65] IEC 60300-3-2:2004, Dependability management — Part 3-2: Application guide — Collection of dependability data from the field (Управление общей надежностью. Часть 3-2. Руководство по применению. Полевой сбор данных по общей надежности)

[66] IEC 60068-1:2013, Environmental testing — Part 1: General and guidance (Испытание на воздействие внешних факторов. Часть 1. Общие положения и руководство)

[67] Статический анализ и обеспечение программного обеспечения. Дэвид Вагнер U.C. Berkeley. gost.e-ecolog.ru eecs.berkeley.edu/~daw/talks/sas01.ppt (доступ 12.04.2021)

[68] Билл Грэм, Пол Н. Леру. Использование статического и динамического анализа для повышения качества продукции и эффективности разработки. SWD Software Ltd., 2015. gost.e-ecolog.ru (доступ 12.04.2021)

[69] IEC 60812:2018, Failure modes and effects analysis (FMEA and FMECA) [Анализ видов и последствий отказов (FMEA и FMECA)]

[70] IEC 62502:2010, Analysis techniques for dependability — Event tree analysis (ETA) (Методы анализа надежности. Анализ дерева событий)

[71] Анализ вида, последствий и критичности отказов (FMECA). gost.e-ecolog.ru 12.04.2021)

[72] ARP926A-2011 Процедура анализа неисправностей/отказов. gost.e-ecolog.ru (доступ 12.04.2021)

[73] Мариса А. Санчес и Мигель А. Фельдер. Систематический подход к созданию тестовых примеров на основе неисправностей. gost.e-ecolog.ru (доступ 12.04.2021)

[74] IEC 61165:2006, Application of Markov techniques (Применение марковских методов)

[75] IEC 61078:2016, Reliability block diagrams (Блок-схемы расчета надежности)

[76] ISO/IEC Guide 98-3:2008, Uncertainty of measurement — Part 3: Guide to the expression of uncertainty in measurement (GUM:1995) — Supplement 1: Propagation of distributions using a Monte Carlo method (Неопределенность измерения. Часть 3. Руководство по выражению неопределенности измерения. Дополнение 1. Трансформирование распределений с использованием метода Монте-Карло)

[77] IEC 61025:2006, Fault tree analysis (FTA) (Анализ диагностического дерева неисправностей)

[78] Бинарная схема принятия решений (BDD). Куала-Лумпур 2009. Международная конференция по будущему компьютеру и коммуникации. Печать ISBN: 978-0-7695-3591-3, Номер присоединения: 10804572, DOI: 10.1109/ICFCC.2009.31. gost.e-ecolog.ru (доступ 12.04.2021)

[79] IEC 62551:2012, Analysis techniques for dependability — Petri net techniques (Способы анализа надежности. Метод Petri net)

[80] Чернышова Н.Н. Имитационное моделирование бизнес-процессов. (Учебно-методическое пособие). Нижний Новгород: Изд-во Нижегородского государственного университета, 2010. — 28 с.

[81] Рябухин С.И. Применение сетей Петри для моделирования событийно-процессных цепей и построения структур базы данных//Вестник Новосиб. гос. ун-та. Серия: Информационные технологии. 2013. Т. 11, вып. 4. С. 92—101. ISSN 1818-7900

[82] Болотковский Ю.И. OrCAD Моделирование. Поваренная книга. gost.e-ecolog.ru bolotovsky_orcad_091-173.pdf (доступ 12.04.2021)

[83] Полянский А. Лекции. gost.e-ecolog.ru (доступ 12.04.2021)

[84] IEC 61069-5:2016, Industrial-process measurement, control and automation — Evaluation of system properties for the purpose of system assessment — Part 5: Assessment of system dependability (Измерение и управление промышленным процессом. Определение свойств системы с целью ее оценки. Часть 5. Оценка надежности системы)

[85] Требования к проектированию. Э. Халл, К. Джексон Springer, 2005, ISBN 1852338792, 9781852338794. http:// gost.e-ecolog.ru (доступ 12.04.2021)

[86] Бахтизин В.В., Глухова Л.А. Технологии разработки программного обеспечения. Учебное пособие. — Минск: БГУИР, 2010. — 267 с.: ил. ISBN 978-985-488-512-4.

[87] Анализ и проектирование систем. Д. Йейтс, А. Уэйкфилд. Pearson Education, 2003, ISBN 0273655361, 9780273655367. gost.e-ecolog.ru (доступ 12.04.2021)

[88] Разработка систем реального времени. Р.Уильямс. Butterworth-Heinemann, 2006, электронная книга ISBN: 9780080456409, Paperback ISBN: 9780750664714. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[89] ISO/IEC 8631:1989, Information technology — Program constructs and conventions for their representation (Информационные технологии. Программные структуры и условные обозначения для их представления)

[90] Проектирование и разработка программного обеспечения. Г. Ланкастер. Паскаль Пресс, 2001, ISBN 1741251753, 9781741251753. gost.e-ecolog.ru
gost.e-ecolog.ru (доступ 12.04.2021)

[91] Практика формальных методов в критически важных для безопасности системах. С. Лю, В. Ставриду, Б. Дутертре, Ж. Системс. Программное обеспечение 28, 77 — 87, Elsevier, 1995. gost.e-ecolog.ru bitstream/10993/15849/1/INFSOF-D-13-00339R2.pdf (доступ 12.04.2021)

[92] Формальные методы: использование и актуальность для разработки критически важных для безопасности систем. Л.М. Баррока, Дж.А. Макдермид, The Computer Journal 35 (6), 579 — 599, 1992. gost.e-ecolog.ru comjnl/35.6.579 (доступ 12.04.2021)

[93] Как создать правильное программное обеспечение. Введение в формальную спецификацию и разработку программы путем преобразований. E.A. Boiten et al., The Computer Journal 35 (6), 547—554, 1992. gost.e-ecolog.ru academia.edu/288606/High-Integrity_System_ Specification_ and_Design (доступ 12.04.2021)

[94] Связь и параллелизм. Р. Мильнер, Прентис-Холл, 1989, ISBN 9780131150072. gost.e-ecolog.ru co.uk/9780131150072/Communication-Concurrency-Prentice-Hall-International-0131150073/plp (доступ 12.04.2021)

[95] Коммуникация последовательных процессов: Первые 25 лет. А. Абдаллах, С. Джонс, Дж.Сандерс (Эдс.). Springer, 2004, ISBN 3540258132, 9783540258131. gost.e-ecolog.ru (доступ 12.04.2021)

[96] Вычислительная логика высшего порядка. Дж.Ллойд. В вычислительной логике: логическое программирование и Beyond, лекции по информатике, Springer Berlin/Heidelberg, 2002, ISBN 978-3-540-43959-2. gost.e-ecolog.ru springer.com/chapter/10.1007/3-540-45628-7_6 (доступ 12.04.2021)

[97] ISO 8807:1989, Information processing systems; open systems interconnection; LOTOS; a formal description technique based on the temporal ordering of observational behaviour (Системы обработки информации. Взаимосвязь открытых систем. LOTOS. Техника формального описания, основанная на временном упорядочении наблюдаемого поведения)

[98] Проектирование программного обеспечения с OBJ. Алгебраическая спецификация в действии. Дж.Гоген, Г. Малкольм. Springer, 2000, ISBN 0792377575, 9780792377573. gost.e-ecolog.ru (доступ 12.04.2021)

[99] Математическая логика для компьютерных наук. М. Бен-Ари. Springer, 2001, ISBN 1852333197, 9781852333195. gost.e-ecolog.ru esc=y (доступ 12.04.2021)

[100] ISO/IEC 13817-1:1996, Information technology — Programming languages, their environments and system software interfaces — Vienna development method — Specification language — Part 1: Base language (Информационные технологии. Языки программирования, их среды и интерфейсы системного программного обеспечения. Венский метод разработки. Язык спецификации. Часть 1. Базовый язык)

[101] Систематическая разработка программного обеспечения с использованием VDM. С. В.Джонс. Прентис-Холл. 2-е изд., 1990. gost.e-ecolog.ru (доступ 12.04.2021)

[102] Формальная спецификация с использованием языка Z, 2-е изд. Д. Лайтфут. Palgrave Macmillan, 2000, ISBN 9780333763278. gost.e-ecolog.ru (доступ 12.04.2021)

[103] Метод В. С. Шнайдер. Palgrave Macmillan, 2001, ISBN 9780333792841. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[104] Концепции в языках программирования. Дж. Митчелл. Изд-во Кембриджского университета, 2003, ISBN-13: 978-0521780988, ISBN-10: 0521780985. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[105] Основания языков программирования. Дж. Митчелл. Изд-во Регулярная и хаотическая динамика. 2010. 720 с. ISBN 978-5-93972-757-0

[106] Основные концепции языков программирования. Роберт У. Себеста. — Изд-во «Вильямс», 2001. 672 с. ISBN 5-8459-0192-8, 0-201-75295-6

[107] Введение в разработку программного обеспечения на основе компонентов К.-К. Лау и С. ди Кола, Всемирный научный 2017. ISBN 978-981-3221-87-1, 978-981-3221-89-5. gost.e-ecolog.ru worldscibooks/10.1142/10486 (доступ 12.04.2021)

[108] Реинжиниринг: почему и как реконструировать программное обеспечение. Рене Р. Клош. Подготовка к Калифорнийскому симпозиуму по программному обеспечению (CSS 96). Университет Южной Калифорнии, с. 92—99, апрель 1996 года. gost.e-ecolog.ru =pdf (доступ 12.04.2021)

[109] Анализ критичности программного обеспечения COTS/SOUP. П. Бишоп, Т. Клеман, С. Герра. В проектировании надежности и безопасности систем, том 81, выпуск 3, сентябрь 2003 г., Elsevier Ltd., 2003 gost.e-ecolog.ru ist.psu.edu/viewdoc/download?doi=10.1.1.6.7663&rep=rep1&type=pdf (доступ 12.04.2021)

[110] Введение в теорию автоматов, языки и вычисления (3-е изд.). J. Hopcroft, R. Motwani, J. Ullman, Addison-Wesley Longman Publishing Co, 2006, ISBN: 0321462254. gost.e-ecolog.ru (доступ 12.04.2021)

[111] Руководство по научным вычислениям. P.R. Turner. CRC Press, 2001, ISBN 0849312426, 9780849312427. gost.e-ecolog.ru (доступ 12.04.2021)

[112] Диаграммы последовательности сообщений, Д. Харель, П. Тиагараджан. В UML для реального: проектирование встраиваемых систем реального времени. ред. Л. Лаваньо. Springer, 2003, ISBN 1402075014, 9781402075018. gost.e-ecolog.ru (доступ 12.04.2021)

[113] ISO/IEC 19501:2005, Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2 [Информационные технологии. Открытая распределительная обработка. Унифицированный язык моделирования (UML). Версия 1.4.2]

[114] Основы кодов исправления ошибок, В. Хаффман, В. Плесс. Изд-во Кембриджского университета, 2003, ISBN 0521782805, 9780521782807. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[115] Использование трассировок в анализе программ. А. Грос, Р. Джоши. Лекции по информатике, том 3920, Спрингер Берлин/Гейдельберг, 2006, ISBN 978-3-540-33056-1. gost.e-ecolog.ru pdf/10.1007%2F11691372_25.pdf (доступ 12.04.2021)

[116] Моделирование разнообразия при проектировании программного обеспечения. Обзор, Б. Литтлвуд, П. Попов, Л. Стриджини. ACM Computing Survey, том 33, № 2, 2001. gost.e-ecolog.ru (доступ 12.04.2021)

[117] N-версионный подход к отказоустойчивому программному обеспечению, А. Авизиенис, IEEE Transactions on Software Engineering, том SE-11, № 12, с. 1491—1501, 1985. gost.e-ecolog.ru (доступ 12.04.2021)

[118] Экспериментальная оценка принятия независимости в многоверсионном программировании, Дж. Рыцарь, Н. Г. Левесон. IEEE Transactions on Software Engineering, том SE-12, № 1, 1986. gost.e-ecolog.ru gost.e-ecolog.ru)ступ 12.04.2021)

[119] В поисках эффективного разнообразия. Исследование шести языков отказоустойчивого программного обеспечения управления полетом. А. Авизиенис, М. Р. Лю и В. Шютц. 18-й симпозиум по отказоустойчивым вычислениям, Токио, Япония, 27—30 июня 1988 года, IEEE Computer Society Press, 1988, ISBN 0-8186-0867-6. gost.e-ecolog.ru (доступ 12.04.2021)

[120] Изучение компенсируемых транзакций. Цзин Ли, Хуэйбяо Чжу, Гэгуан Пу, Цзифэн Хэ. На семинаре по программному обеспечению, 2007. SEW 2007. IEEE, 2007, ISBN 978-0-7695-2862-5. gost.e-ecolog.ru document/4402774/ (доступ 12.04.2021)

[121] Отказоустойчивость программного обеспечения. Крис Инасио, Университет Карнеги-Меллона, 18-849b, надежные встраиваемые системы, 1998. gost.e-ecolog.ru (доступ 12.04.2021)

[122] Надежные компьютерные системы: проектирование и оценка, Д. П. Сивиорек и Р. С. Шварц, А. К. Петерс Лтд., 1998, ISBN 156881092X gost.e-ecolog.ru dp/156881092X (доступ 12.04.2021)

[123] Обеспечение живучести критически важной системы. Через архитектуры программного обеспечения, E.A. Strunk. Springer Berlin/Heidelberg, 2004, ISBN 978-3-540-23168-4. gost.e-ecolog.ru =ADA446905 (доступ 12.04.2021)

[124] Отказоустойчивость, принцип и практика. Т. Андерсон и П. А. Ли, том 3, надежные вычислительные и отказоустойчивые системы, Springer Verlag, 1987, ISBN 3-211-82077-9. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[125] Диагностика неисправностей: модели, искусственный интеллект, приложения. Дж. Корбич, Дж. Косельни, З. Ковальчук, В. Чолева. Springer, 2004, ISBN 3540407677, 9783540407676. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[126] Программное обеспечение. Проектирование самоадаптирующихся систем. Cheng, B.H.C., de Lemos, R., Inverardi, P., Magee, J. (Eds.). ISBN 978-3-642-02161-9. gost.e-ecolog.ru (доступ 12.04.2021)

[127] B.H.C. Cheng et al. (Eds), Самоадаптивные системы, LNCS 5525, с. 1—26, 2009. gost.e-ecolog.ru book/9783642021602 (доступ 12.04.2021)

[128] Динамическая реконфигурация архитектур программного обеспечения архитектур через аспекты. С. Коста и др. Лекции по информатике, том 4758/2007, Springer Berlin/Heidelberg, 2007, ISBN 978-3-540-75131-1. https://link.springer.com/chapter/10.1007/978-3-540-75132-8_24?no-access=true (доступ 12.04.2021)

[129] Архитектура системы, управляемой по времени. H. Kopetz TU Wien Январь 2000. TU Wien. gost.e-ecolog.ru dependability.org/wg10.4/timedepend/05-Kopet.pdf (доступ 12.04.2021)

[130] На пути к интеграции на основе моделей инструментов и методов для проектирования, проверки и внедрения встраиваемых систем управления. Йозеф Портер, Gábor Karsai, Péter Völgyesi, Harmon Nine, Peter Humke, Graham Hemingway, Ryan Thibodeaux, János Sztipanovits. Международная конференция по типовым инженерным языкам и системам. 2008: Модели в программном обеспечении Engineeringpp 20-34. gost.e-ecolog.ru springer.com/chapter/10.1007/978-3-642-01648-6_3 (доступ 12.04.2021).

[131] Анализ формальной верификации для архитектуры системы, управляемой по времени. Джон Рашби. Документ с приглашением, представленный на FTRTFT 02, Ольденбург, Германия, сентябрь 2002 года. Springer-Verlag LNCS том 2469, с. 83—105. gost.e-ecolog.ru (доступ 12.04.2021)

[132] Комплексный тест по оценке соответствия Ада (АКАТС), версия 2.5, Орган по оценке соответствия Ада, апрель 2002 года. gost.e-ecolog.ru (доступ 12.04.2021)

[133] Демонстрация эквивалентности исходного кода и содержимого PROM. Ди Джей Пэви и Л. А. Уинсборроу. Компьютерный журнал. т. 36, № 7, 1993

[134] ISO/IEC 1539-1:2018, Information technology — Programming languages — Fortran — Part 1: Base language (Информационная технология. Языки программирования. Фортран. Часть 1. Основной язык)

[135] ISO/IEC 7185:1990, Information technology — Programming languages — Pascal (Информационные технологии. Языки программирования. Паскаль)

[136] ISO/IEC 8652:2012, Information technology — Programming languages — Ada (Информационная технология. Языки программирования. Ada)

[137] ISO/IEC 9899:2018, Information technology — Programming languages — C (Информационные технологии. Языки программирования. Си)

[138] ISO/IEC 10206:1991, Information technology — Progamming languages — Extended Pascal (Информационные технологии. Языки программирования. Расширенный Паскаль)

[139] ISO/IEC 10514-1:1996, Information Technology — Programming Languages — Part 1: Modula-2, Base Language (Информационные технологии. Языки программирования. Часть 1. Базовый язык Modula-2)

[140] ISO/IEC 10514-3:1998, Information technology — Programming languages — Part 3: Object Oriented Modula-2 (Информационные технологии. Языки программирования. Часть 3. Modula-2, объектно-ориентированный)

[141] ISO/IEC 14882:2017, Programming languages — C++ (Языки программирования. C++)

[142] IEC TR 15942:2000, Information technology — Programming languages — Guide for the use of the Ada programming language in high integrity systems (Информационные технологии. Языки программирования. Руководство по применению языка программирования Ada в системах высокой целостности)

[143] Создание встроенного программного обеспечения на основе языков проектирования на уровне системы, Р. Домер, Д. Гайски. В «ASP-DAC 2004: Материалы конференции ASP-Dac 2004 «Азиатско-южнотихоокеанская конференция по автоматизации проектирования, 2004», IEEE Circuits and Systems Society. IEEE, 2004, ISBN 0780381750, 9780780381759 gost.e-ecolog.ru (доступ 12.04.2021)

[144] Преобразование моделей алгебры процесса в конечные автоматы UML. Преодолен ли семантический разрыв? М.Ф. ван Амстел и др., «Первая международная конференция по теории и практике модельных преобразований, ICMT». Ред. А. Вальесильо. Springer, 2008, ISBN 3540699260, 9783540699262. http://gost.e-ecolog.ru gost.e-ecolog.ru -2-2008-proceedings/oclc/272298867 (доступ 12.04.2021)

[145] Управление процессом тестирования. Практические инструменты и методы управления тестированием оборудования и программного обеспечения. Р. Блэк, Джон Уайли и сыновья, 2002, ISBN 0471223980, 9780471223986. gost.e-ecolog.ru - Tools-0471223980/plp (доступ 12.04.2021)

[146] Обсуждение статистического тестирования для применения, связанного с безопасностью. С. Кубалл, Ж.Х.Р. Май, Proc IMechE. Том 221. Часть O: J. Риск и надежность, Институт инженеров-механиков, 2007 gost.e-ecolog.ru (доступ 12.04.2021)

[147] Оценка вероятности отказа при тестировании выявляет отсутствие отказов, W.K. Miller, L.J. Morell и др. IEEE Transactions on Software Engineering, том 18, No. 1, стр. 33—43, январь 1992 года. gost.e-ecolog.ru chapter/10.1007/3-540-45416-0_16 (доступ 12.04.2021)

[148] Оценка надежности по результатам соответствующих испытаний программного обеспечения для защиты растений, J. May, G. Hughes, A.D. Lunn. Журнал по разработке программного обеспечения IEE. Том 10. № 6. стр. 206—218, ноябрь 1995 года (ISSN: 0268-6961). gost.e-ecolog.ru 12.04.2021)

[149] Валидация сверхвысокой надежности для систем на базе программного обеспечения, B. Littlewood и L. Strigini. Комм. ACM 36 (11), 69—80, 1993. gost.e-ecolog.ru (доступ 12.04.2021)

[150] Надежность критически важных компьютерных систем 2. Ф. Дж. Редмилл, Elsevier Applied Science, 1989, ISBN 1851663819, 9781851663811. gost.e-ecolog.ru tems/F-J-Redmill/9781851663811 (доступ 12.04.2021)

[151] Внесение ошибок в программное обеспечение. Защита программ от ошибок. Дж. Воас, Г. Макгро. Wiley Computer Pub., 1998, ISBN 0471183814, 9780471183815. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[152] А. Ахо, Р. Сети, Дж. Ульман. «Компиляторы: принципы, технологии и инструменты», М.: «Вильямс», 2003. 768 с. gost.e-ecolog.ru (доступ 12.04.2021)

[153] Использование символьного выполнения для верификации систем, критически важных для безопасности. А. Коэн-Поризини, Г. Денаро, С. Гецци, М. Материалы 8-й Европейской конференции по разработке программного обеспечения и 9-го международного симпозиума ACM SIGSOFT по основам разработки программного обеспечения. ACM, 2001, ISBN:1-58113-390-1 gost.e-ecolog.ru gost.e-ecolog.ru 4.2021)

[154] Использование символьного выполнения для управления созданием текста. G. Lee, J. Morris, K. Parker, G. Bundell, P. Lam. In Software Testing, Verification and Reliability, vol. 15, No. 1, 2005. John Wiley & Sons, Ltd. gost.e-ecolog.ru gost.e-ecolog.ru 04.2021)

[155] Является ли доказательство более экономичным, чем тестирование? С. Кинг, Р. Чепмен, Дж. Хаммонд, А. Прайор. IEEE Transactions on Software Engineering, vol. 26, No. 8, August 2000. gost.e-ecolog.ru cfm?id=631243 (доступ 12.04.2021)

[156] Проверка модели. Э. М. Кларк, О. А. Пелед. MIT Press, 1999, ISBN 0262032708, 9780262032704. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[157] Проверка систем и программного обеспечения. Методы и инструменты проверки моделей. Б. Берард, М. Бидойт, А. Финкель, Ф. Ларуссини, А. Пети, Л. Петруччи, Ф. Шноебелен, и П. Маккензи, Springer, 2001, ISBN 3-540-41523-8. gost.e-ecolog.ru (доступ 12.04.2021)

[158] Логика в информатике: моделирование и рассуждения о системах. М. Хут и М. Райан. Изд-во Кембриджского университета, 2000 год, ISBN 0521652006, 0521656028. gost.e-ecolog.ru gost.e-ecolog.ru (доступ 12.04.2021)

[159] Метрики и модели при проектировании программного обеспечения. S.H. Kan. Addison-Wesley, 2003, ISBN 0201729156, 9780201729153. gost.e-ecolog.ru Quality_E.html?id=EaefcL3pWJYC&redir_esc=y (доступ 12.04.2021)

[160] Fagan, M. Проверки проекта и кода для уменьшения ошибок при разработке программы. IBM Systems Journal 15, 3 (1976): 182—211 gost.e-ecolog.ru (доступ 12.04.2021)

[161] EmStar: среда для разработки программного обеспечения беспроводных встраиваемых систем. J. Elson et al. gost.e-ecolog.ru , gost.e-ecolog.ru (доступ 12.04.2021)

[162] Технология разработки программного обеспечения, учебное пособие для студентов, Минск, БНТУ, 2020

[163] Интеграция возможностей модели зрелости: руководство по интеграции процессов и улучшению продукции, Мэри Бет Криссис, Майк Конрад, Сэнди Шрум, Аддисон-Уэсли, 2003, ISBN 0321154967, 9780321154965. gost.e-ecolog.ru (доступ 12.04.2021)

[164] Анимация B модели для внешней верификации. H. Waeselynck, S. Behnia, In Proceedings of the Second International Conference on Formal Engineering Methods, 1998. Компьютерное общество IEEE, 1998, ISBN 0-8186-9198-0. gost.e-ecolog.ru (доступ 12.04.2021)

[165] Экард Брингманн, Андреас Крамер; Тестирование автомобильных систем на основе моделей: ICST, с. 485—493, 2008 Международная конференция по тестированию, проверке и валидации программного обеспечения, 2008. gost.e-ecolog.ru (доступ 12.04.2021).

[166] Брой М., Проблемы в области разработки автомобильного программного обеспечения, Международная конференция по разработке программного обеспечения (ICSE '06), Шанхай, Китай, 2006. gost.e-ecolog.ru citation.cfm?id=1134292, gost.e-ecolog.ru (доступ 12.04.2021)

[167] Хаймдаль, М.П. Е.: Тестирование на основе моделей: впереди проблемы, Конференция по компьютерному программному обеспечению и приложениям (COMPSAC 2005), 25—28 июля 2005 года, Эдинбург, Шотландия, Великобритания, 2005. gost.e-ecolog.ru challenges_ahead (доступ 12.04.2021)

[168] Джонатан Джеки, Маргус Веанес, Колин Кэмпбелл и Вольфрам Шульте, тестирование и анализ программного обеспечения на основе моделей с C #, ISBN 978-0-521-68761-4, Cambridge University Press, 2008. https://books.google.ru/books/about/Model_Based_Software_Testing_and_Analysi.html?id=3LW7Y8y2F4oC&redir_esc=y (доступ 12.04.2021)

[169] С.Дж. Проуэлл, Применение моделей, использующих цепи Маркова, для тестирования сложных систем, HICSS '05:38 Ежегодные Гавайи, Международная конференция по системным наукам, 2005. gost.e-ecolog.ru researchgate.net/publication/221184160_Using_Markov_Chain_Usage_Models_to_Test_Complex_Systems (доступ 12.04.2021)

[170] Марк Уттинг и Бруно Легард, Практическое тестирование на основе моделей. Подход к инструментам, ISBN 978-0-12-372501-1, Morgan-Kaufmann, 2007. gost.e-ecolog.ru Model-Based_Testing_A_Tools_Approach (доступ 12.04.2021)

[171] Ян Бэйли и Хун Чжу. Формальная спецификация вариантов и поведенческих особенностей шаблонов проекта. Препринт представлен в Journal of Systems and Software 6 июля 2009 года. gost.e-ecolog.ru HongZhu/Publications/compsac08JSI-v8.pdf (доступ 12.04.2021)

[172] Модельное тестирование реактивных систем: серия лекций, LNCS 3472, Springer-Verlag, 2005, ISBN 978-3-540-26278-7. gost.e-ecolog.ru (доступ 12.04.2021)

[173] Тестирование на основе модели. Подключение спецификаций и тестирование. 9/26/2007. gost.e-ecolog.ru cs.utexas.edu/users/browne/uvvf2007/ModelBasedTesting.pdf (доступ 12.04.2021)

[174] OF-FMEA подход к анализу безопасности систем, преимущественно использующих объектно-ориентированное ПО, T. Cichocki, J. Gorski. В искусственном интеллекте и безопасности в вычислительных системах: 9-я Международная конференция, ACS'2002. Эд. Дж. Сольдек. Springer, 2003, ISBN 1402073968, 9781402073960. gost.e-ecolog.ru (доступ 12.04.2021)

[175] Майрон Хехт, Эмили Димпфль: Автоматизированная генерация режимов отказов и анализ результатов из моделей SysML. DOI: 10.13140/2.1.4578.9446. gost.e-ecolog.ru Generation_of_Failure_Modes_and_Effects_Analysis_from_SysML_Models (доступ 12.04.2021).

[176] Анализ надежности иерархических компьютерных систем, подверженных отказам по общей причине Л. Син, Л. Мешкат, С. Донохью. Проектирование надежности и безопасность систем, том 92, выпуск 3, март 2007. gost.e-ecolog.ru (доступ 12.04.2021)

[177] IEEE VHDL. Руководство по проектированию на Verilog + Standard VHDL. gost.e-ecolog.ru help/VHDL/p1076/P1076_Chap_00.pdf (доступ 12.04.2021)

[178] IEEE 1149.1-2001 — IEEE Standard Test Access Port and Boundary Scan Architecture (ИИЭР Стандартный порт тестового доступа и архитектура сканирования границ)

[179] IEC 61164:2004, Reliability growth — Statistical test and estimation methods (Повышение надежности. Статистические критерии и методы оценки)

[180] Верификация и валидация программного обеспечения в реальном времени. Глава 5. У. Дж. Квирк (ред.). Springer Verlag, 1985, ISBN 3-540-15102-8. gost.e-ecolog.ru (доступ 12.04.2021)

[181] Объединение усилий по вероятностной и детерминированной верификации. W.D. Ehrenberger, SAFECOMP 92, Пергамский пресс, ISBN 0-08-041893-7. gost.e-ecolog.ru B9780080418933500537 (доступ 12.04.2021)

[182] IEEE 352-2016 — IEEE Guide for General Principles of Reliability Analysis of Nuclear Power Generating Station Systems and Other Nuclear Facilities (ИИЭР Руководство по общим принципам анализа надежности систем атомной электростанции и других ядерных установок)

УДК 621.5:814.8:006.354

МКС 13.200;
13.220;
13.310;
13.320;
91.120.99

NEQ

Ключевые слова: системы, связанные с безопасностью зданий и сооружений; меры по снижению риска; методы оценки; методы и средства для контроля случайных отказов; методы и средства по предотвращению систематических отказов

Редактор *Л.В. Коретникова*
Технический редактор *И.Е. Черепкова*
Корректор *М.В. Бучная*
Компьютерная верстка *Л.А. Круговой*

Сдано в набор 31.05.2021. Подписано в печать 22.06.2021. Формат 60×84⅛. Гарнитура Ариал.
Усл. печ. л. 12,09. Уч.-изд. л. 10,94.

Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

Создано в единичном исполнении во ФГУП «СТАНДАРТИНФОРМ»
для комплектования Федерального информационного фонда стандартов,
117418 Москва, Нахимовский пр-т, д. 31, к. 2.
gost.e-ecolog.ru gost.e-ecolog.ru

Поправка к ГОСТ 34332.5—2021 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 5. Меры по снижению риска, методы оценки

В каком месте	Напечатано	Должно быть		
Предисловие. Таблица согласования	—	Казахстан	KZ	Госстандарт Республики Казахстан

(ИУС № 4 2022 г.)

Поправка к ГОСТ 34332.5—2021 Безопасность функциональная систем, связанных с безопасностью зданий и сооружений. Часть 5. Меры по снижению риска, методы оценки

В каком месте	Напечатано	Должно быть
Предисловие. Таблица согласования	—	Азербайджан | AZ | Азстандарт

(ИУС № 8 2023 г.)

gost.e-ecolog.ru