ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 26262-6— 2014

ДОРОЖНЫЕ ТРАНСПОРТНЫЕ СРЕДСТВА ФУНКЦИОНАЛЬНАЯ БЕЗОПАСНОСТЬ

Часть 6

Разработка программного обеспечения изделия

ISO 26262-6:2011
Road vehicles – Functional safety – Part 6: Product development at the software level (IDT)

Издание официальное

Москва
Стандартинформ
2015

Предисловие

1 ПОДГОТОВЛЕН Обществом с ограниченной ответственностью «Корпоративные электронные системы» и Федеральным государственным учреждением «Консультационно-внедренческая фирма в области международной стандартизации и сертификации - «Фирма «ИНТЕРСТАНДАРТ» на основе собственного аутентичного перевода на русский язык международного документа, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 58 «Функциональная безопасность»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии России от 10 июня 2014 г. № 524-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 26262-6:2011 «Дорожные транспортные средства. Функциональная безопасность. Часть 6. Разработка программного обеспечения изделия» (ISO 26262-6:2011 «Road vehicles – Functional safety – Part 6: Product development at the software level»)

Наименование настоящего стандарта изменено относительно наименования указанного международного стандарта для приведения в соответствие с ГОСТ Р 1.5 (подраздел 3.5)

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов и документов соответствующие им национальные стандарты, сведения о которых приведены в дополнительном приложении ДА

5 ВВЕДЕН ВПЕРВЫЕ

Правила применения настоящего стандарта установлены в ГОСТ Р 1.0—2012 (раздел 8). Информация об изменениях к настоящему стандарту публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок – в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске информационного указателя «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования – на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (gost.ru)

Содержание

Введение

Комплекс стандартов ИСО 26262 является адаптацией комплекса стандартов МЭК 61508 и предназначен для применения электрических и/или электронных (Э/Э) систем в дорожно-транспортных средствах.

Это адаптация распространяется на все виды деятельности в процессе жизненного цикла систем, связанных с безопасностью, включающих электрические, электронные и программные компоненты.

Безопасность является одним из важнейших вопросов в автомобилестроении. Создание новых функциональных возможностей не только в таких системах, как содействие водителю, силовые установки, управление динамикой автомобиля, но и в активных и пассивных системах безопасности тесно связано с деятельностью по проектированию систем безопасности. Разработка и интеграция этих функциональных возможностей повышает необходимость использования процессов разработки систем безопасности и обеспечения доказательств того, что все обоснованные цели системы безопасности выполнены.

С ростом сложности технологий, программного обеспечения и мехатронных устройств увеличиваются риски, связанные с систематическими отказами и случайными отказами оборудования. Чтобы предотвратить эти риски, комплекс стандартов ИСО 26262 включает соответствующие требования и процессы.

Безопасность системы достигается за счет ряда мер безопасности, которые реализуются с применением различных технологий (например, механических, гидравлических, пневматических, электрических, электронных, программируемых электронных) и применяются на различных уровнях процесса разработки. Несмотря на то, что настоящий стандарт касается функциональной безопасности Э/Э систем, подход, рассматриваемый в настоящем стандарте, может быть использован для разработки связанных с безопасностью систем, основанных на других технологиях. Настоящий стандарт:

a) обеспечивает жизненный цикл систем безопасности автомобиля (менеджмент, разработку, производство, эксплуатацию, обслуживание, вывод из эксплуатации) и поддерживает адаптацию необходимых действий для выполнения этих стадий жизненного цикла;

b) обеспечивает разработанный специально для автотранспорта основанный на риске подход для определения уровней полноты безопасности [уровни полноты безопасности автомобиля (УПБА)];

c) использует значения УПБА при спецификации соответствующих требований, чтобы предотвратить неоправданный остаточный риск;

d) устанавливает требования к мерам проверки соответствия и подтверждения, которые обеспечивают достижение достаточного и приемлемого уровня безопасности;

e) устанавливает требования к взаимодействию с поставщиками.

На функциональную безопасность влияют процессы разработки (в том числе спецификация требований, реализация, внедрение, интеграция, верификация, подтверждение соответствия и управление конфигурацией), процессы производства и обслуживания, а также процессы управления.

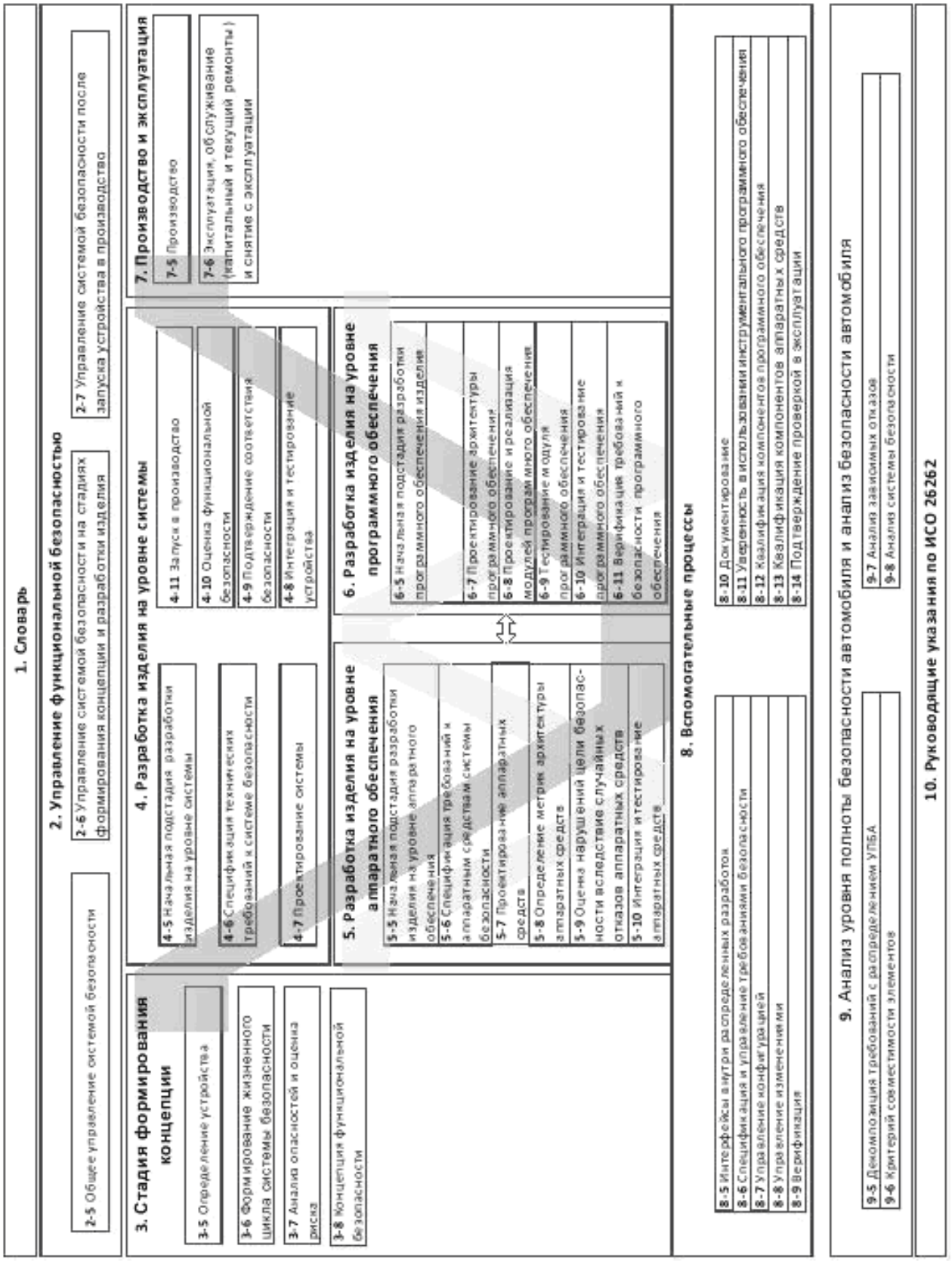
Вопросы безопасности тесно связаны с любыми опытно-конструкторскими работами, реализующими функционал и обеспечивающими качество создаваемых изделий, а также с результатами таких работ. Настоящий стандарт рассматривает связанные с безопасностью проблемы,касающиеся опытно-конструкторских работ и их результатов.

На рисунке 1 показана общая структура комплекса ИСО 26262. В нем для различных стадий разработки изделия используется эталонная V-модель процесса. На рисунке 1:

- залитая область в виде символа «V» представляет взаимосвязь между ИСО 26262-3, ИСО 26262-4, ИСО 26262-5, ИСО 26262-6 и ИСО 26262-7;

- ссылки на конкретную информацию даны в виде: «m-n», где «m» представляет собой номер части настоящего стандарта, а «n» указывает на номер раздела этой части.

Пример – 2-6 ссылается на пункт 6 ИСО 26262-2.



НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ДОРОЖНЫЕ ТРАНСПОРТНЫЕ СРЕДСТВА. ФУНКЦИОНАЛЬНАЯ БЕЗОПАСНОСТЬ

Часть 6

Разработка программного обеспечения изделия

Road vehicles – Functional safety – Part 6: Product development at the software level

Дата введения — 2015—05—01

1 Область применения

Настоящий стандарт применяется к связанным с безопасностью системам, включающим в себя одну или несколько электрических и/или электронных (Э/Э) систем, которые установлены в серийно производимых легковых автомобилях с максимальной массой (брутто) транспортного средства до 3500 кг. Настоящий стандарт не применяется для уникальных Э/Э систем в транспортных средствах специального назначения, таких как транспортные средства, предназначенные для водителей с ограниченными возможностями.

Системы и их компоненты, находящиеся в производстве или на стадии разработки до даты публикации настоящего стандарта, не входят в его область применения. Если разрабатываемые автомобили или их модификации используют системы и их компоненты, выпущенные до публикации настоящего стандарта, то только модификации этих систем должны быть разработаны в соответствии с настоящим стандартом.

Настоящий стандарт рассматривает возможные опасности, вызванные некорректным поведением Э/Э связанных с безопасностью систем, а также некорректным взаимодействием этих систем. Настоящий стандарт не рассматривает опасности, связанные с поражением электрическим током, возгоранием, задымлением, перегревом, излучением, токсичностью, воспламеняемостью, химической активностью, коррозией и подобные опасности, если они непосредственно не вызваны некорректным поведением Э/Э связанных с безопасностью систем.

Настоящий стандарт не рассматривает номинальные рабочие характеристики Э/Э систем, даже если для таких систем существуют стандарты, посвященные их функциональным рабочим характеристикам (например, активные и пассивные системы безопасности, тормозные системы, адаптивный круиз-контроль).

Настоящий стандарт устанавливает требования к разработке программного обеспечения изделия для автомобильной промышленности, в том числе:

- требования к инициализации разработки программного обеспечения изделия;
- спецификацию требований к безопасности программного обеспечения;
- требования к проектированию архитектуры программного обеспечения;
- требования к проектированию и реализации модуля программного обеспечения;
- требования к тестированию модуля программного обеспечения;
- требования к интеграции и тестированию программного обеспечения;
- требования к верификации требований к безопасности программного обеспечения.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты:

ИСО 26262-1:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 1. Термины и определения (ISO 26262-2:2011, Road vehicles – Functional safety – Part 1: Vocabulary)

ИСО 26262-2:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 2. Менеджмент функциональной безопасности (ISO 26262-2:2011, Road vehicles – Functional safety – Part 2: Management of functional safety)

ИСО 26262-3:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 3. Стадия формирования концепции (ISO 26262-3:2011, Road vehicles – Functional safety – Part 3: Concept phase)

ИСО 26262-4:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 4.

Издание официальное

Разработка изделия на уровне системы (ISO 26262-4:2011, Road vehicles – Functional safety – Part 4: Product development at the system level)

ИСО 26262-5:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 5. Разработка аппаратных средств изделия (ISO 26262-5:2011, Road vehicles – Functional safety – Part 5: Product development at the hardware level)

ИСО 26262-7:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 7. Производство и эксплуатация (ISO 26262-7:2011, Road vehicles – Functional safety – Part 7: Production and operation)

ИСО 26262-8:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 8. Вспомогательные процессы (ISO 26262-8:2011, Road vehicles – Functional safety – Part 8: Supporting processes)

ИСО 26262-9:2011 Дорожно-транспортные средства. Функциональная безопасность. Часть 9. Анализ уровня полноты безопасности автомобиля и анализ безопасности автомобиля (ISO 26262-9:2011, Road vehicles – Functional safety – Part 9: Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses)

Примечание – При пользовании настоящим стандартом целесообразно проверить действие ссылочных стандартов в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет или по ежегодному информационному указателю «Национальные стандарты», который опубликован по состоянию на 1 января текущего года, и по выпускам ежемесячного информационного указателя «Национальные стандарты» за текущий год Если заменен ссылочный стандарт, на который дана недатированная ссылка, то рекомендуется использовать действующую версию этого стандарта с учетом всех внесенных в данную версию изменений. Если заменен ссылочный стандарт, на который дана датированная ссылка, то рекомендуется использовать версию этого стандарта с указанным выше годом утверждения (принятия). Если после утверждения настоящего стандарта в ссылочный стандарт, на который дана датированная ссылка, внесено изменение, затрагивающее положение, на которое дана ссылка, то это положение рекомендуется применять без учета данного изменения. Если ссылочный стандарт отменен без замены, то положение, в котором дана ссылка на него, рекомендуется применять в части, не затрагивающей эту ссылку.

3 Термины, определения и сокращения

В настоящем стандарте применимы термины, определения и сокращения по ИСО 26262-1:2011.

4 Требования соответствия настоящему стандарту

4.1 Общие требования

Для соответствия настоящему стандарту должно быть выполнено каждое его требование, если для этого требования не выполняется одно из следующих условий:

a) в соответствии с настоящим стандартом предусмотрена настройка действий по обеспечению безопасности, поэтому данное требование не применяется или

b) существует обоснование того, что несоблюдение данного требования допустимо, а также показано соответствие этого обоснования настоящему стандарту.

Информация, обозначенная как «примечание» или «пример», должна использоваться только для понимания или для уточнения соответствующего требования и не должна толковаться как самостоятельное требование или быть для него полной или исчерпывающей.

Результаты действий по обеспечению безопасности представлены как результаты работы. В пунктах «Предварительные требования» перечисляется информация, которая должна быть доступна как результат работы предыдущей стадии. Так как некоторые требования разделов настоящего стандарта зависят от УПБА или могут быть адаптированы, то некоторые результаты работы в качестве предварительных условий могут не понадобиться.

В пунктах «Дополнительная информация» содержится информация, которую можно учитывать, но для которой в некоторых случаях настоящий стандарт не требует, чтобы она была результатом работы предыдущей стадии. Такая информация может быть доступна из внешних источников, от лиц или организаций, которые не несут ответственность за деятельность по обеспечению функциональной безопасности.

4.2 Интерпретация таблиц

В настоящем стандарте используются нормативные или справочные таблицы в зависимости от их контекста. Перечисленные в таблице различные методы вносят вклад в уровень уверенности в достижении соответствия с рассматриваемым требованием. Каждый метод в таблице включен либо в

а) последовательный список методов (он обозначен порядковым номером в левой колонке, например, 1, 2, 3) или

б) альтернативный список методов (он обозначен номером с последующей буквой в левом столбце, например, 2а, 2б, 2в).

В случае последовательного списка должны применяться все методы согласно рекомендациям для соответствующего значения УПБА. Если будут применяться другие методы, отличные от перечисленных, то должно быть дано обоснование, что они удовлетворяют соответствующим требованиям.

В случае альтернативного списка должна применяться подходящая комбинация методов в соответствии с указанным значением УПБА независимо от того, перечислены в таблице эти комбинации или нет. Если перечисленные методы имеют разные степени рекомендуемости их применения для некоторого значения УПБА, то следует отдать предпочтение методам с более высокой степенью рекомендуемости. Должно быть дано обоснование, что выбранная комбинация методов выполняет соответствующее требование.

Примечание – Обоснование, основанное на методах, перечисленных в таблице, является достаточным. Но это не означает, что существует какое-то предубеждение за или против применения методов, не перечисленных в таблице.

Для каждого метода степень рекомендуемости его применения зависит от значения УПБА и классифицируется следующим образом:

- "+ +" означает, что метод очень рекомендуется для определенного значения УПБА;
- "+" означает, что метод рекомендуется для определенного значения УПБА;
- "О" означает, что метод не имеет рекомендации за или против его применения для определенного значения УПБА.

4.3 Требования и рекомендации, зависимые от значения УПБА

Требования или рекомендации каждого подраздела должны соблюдаться для значений УПБА A, B, C и D, если не указано иное. Эти требования и рекомендации связаны со значениями УПБА цели безопасности. Если в соответствии с требованиями раздела 5 ИСО 26262-9 декомпозиция УПБА была выполнена на более ранней стадии разработки, то значения УПБА, полученные в результате декомпозиции, должны соблюдаться.

Если в настоящем стандарте значение УПБА дается в круглых скобках, то соответствующий подпункт должен рассматриваться как рекомендация, а не требование для этого значения УПБА. Это не относится к круглым скобкам в нотации, связанной с декомпозицией УПБА.

5 Начальная подстадия разработки программного обеспечения изделия

5.1 Цель

Цель данной подстадии заключается в планировании и инициировании действий по обеспечению функциональной безопасности подстадии разработки программного обеспечения.

5.2 Общие положения

Инициирование разработки программного обеспечения заключается в планировании деятельности по определению и формированию планов для подстадий разработки программного обеспечения и их вспомогательных процессов (см. ИСО 26262-8 и ИСО 26262-9) в зависимости от объема и сложности разработки устройства. Подстадии разработки программного обеспечения и вспомогательные процессы инициируются определением соответствующих методов, обеспечивающих выполнение требований и соответствующих значений УПБА. Методы поддерживаются руководящими указаниями и инструментальными средствами, которые определяются и планируются для каждой подстадии и вспомогательного процесса.

Примечание – Инструментальные средства, используемые для разработки программного обеспечения, могут включать в себя инструментальные средства, отличные от программных инструментальных

средств.

Пример – Инструментальные средства, используемые для стадий тестирования.

Планирование разработки программного обеспечения должно координироваться с разработкой изделия на уровне системы (см. ИСО 26262-4) и разработкой аппаратных средств изделия (см. ИСО 26262-5).

5.3 Входная информация

5.3.1 Предварительные требования

Необходима следующая информация:

- план проекта (уточненный) в соответствии с 5.5.1 ИСО 26262-4;
- план по обеспечению безопасности (уточненный) в соответствии с 5.5.2 ИСО 26262-4;
- техническая концепция системы безопасности в соответствии с 7.5.1 ИСО 26262-4;
- спецификация проекта системы в соответствии с 7.5.2 ИСО 26262-4;
- план интеграции и тестирования устройства (уточненный) в соответствии с 8.5.1 ИСО 26262-4.

5.3.2 Дополнительная информация

Следующая информация может быть учтена:

- доступность квалифицированных программных средств (см. раздел 11 ИСО 26262-8);
- доступность квалифицированных компонентов программного обеспечения (см. раздел 12 ИСО 26262-8);
- руководства по проектированию и кодированию для языков моделирования и программирования (из внешнего источника);
- руководства по применению методов (из внешнего источника); и
- руководящие указания по применению инструментальных средств (из внешнего источника).

5.4 Требования и рекомендации

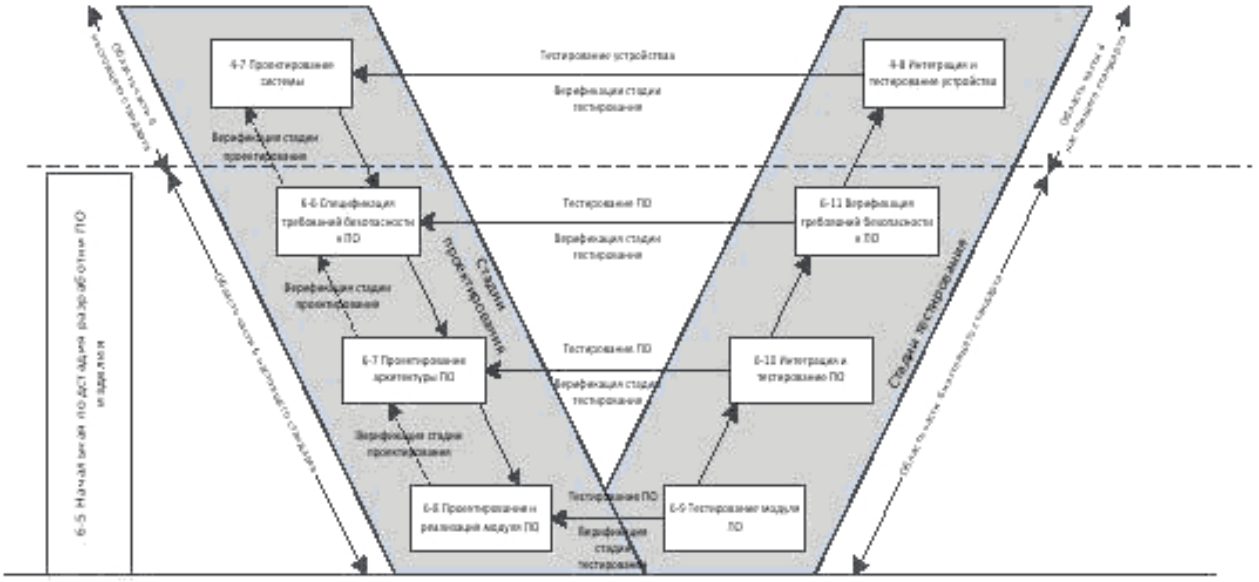
5.4.1 Должны быть спланированы действия и определены соответствующие методы разработки программного обеспечения изделия.

5.4.2 Должна быть выполнена настройка жизненного цикла разработки программного обеспечения изделия в соответствии с требованиями 6.4.5 ИСО 26262-2, на основе базовой модели стадии, приведенной на рисунке 2.

5.4.3 В случае создания конфигурируемого программного обеспечения применяется приложение С.

5.4.4 Процесс разработки программного обеспечения устройства, включая стадии жизненного цикла, методы, языки и инструментальные средства, должен быть согласованным для всех подстадий жизненного цикла программного обеспечения и быть совместимым со стадиями разработки на уровне системы и аппаратных средств так, чтобы необходимые данные могли быть преобразованы корректно.

Примечание – Последовательность стадий, задач и действий, включая итерации, для программного обеспечения устройства является гарантией обеспечения согласованности соответствующих результатов разработки аппаратных средств изделия (см. ИСО 26262-5) и результатов разработки на уровне системы (см. ИСО 26262-4).



Примечание – На рисунке конкретные разделы каждой части настоящего стандарта указаны следующим образом: «m-n», где «m» представляет собой номер части настоящего стандарта, а «n» указывает на номер ее раздела, например, 4-7 представляет раздел 7 ИСО 26262-4.

Рисунок 2 – Базовая модель стадии разработки программного обеспечения (ПО)

5.4.5 Для каждой подстадии разработки программного обеспечения должны быть выбраны, включая руководство по их применению,

a) методы и

b) соответствующие инструментальные средства.

5.4.6 Критериями выбора подходящего языка моделирования или программирования являются:

a) однозначность определения;

Пример – Синтаксиса и семантик языка.

b) поддержка встроенного программного обеспечения систем реального времени и обработка ошибок во время выполнения;

c) поддержка модульности, абстракции и структурированных конструкций.

Критерии, которые не получили достаточного развития в самом языке, должны быть поддержаны соответствующими рекомендациями или средой разработки.

Примечания

1 Выбранный язык программирования (например, ADA, C, C + +, Java, Ассемблер или графический язык моделирования) поддерживает свойства, приведенные в 5.4.7. Для выполнения этих свойств могут использоваться соответствующие рекомендации по программированию или моделированию.

2 Для тех частей программного обеспечения, где использование языков высокого уровня программирования не подходит, таких как низкоуровневое программное обеспечение интерфейса с техническими средствами, обработчики прерываний или критические по времени алгоритмы, могут быть использованы языки Ассемблера.

5.4.7 Для поддержки корректности разработки и реализации руководства по разработке и кодированию для моделирования или языков программирования должны рассматривать вопросы, перечисленные в таблице 1.

Примечания

1 Руководства по кодированию, как правило, различны для различных языков программирования.

2 Руководства по кодированию могут отличаться для разработки на основе модели.

3 Существующие руководства по кодированию могут быть модифицированы для разработки конкретного устройства.

Пример – MISRA C [3] и MISRA AC AGC [4] являются руководствами по кодированию для языка программирования C.

Таблица 1 – Вопросы, которые должны быть рассмотрены в руководствах по моделированию и кодированию

Свойства		УПБА			
		A	B	C	D
1a	Принудительное применение низкой сложности [a]	++	++	++	++
1b	Использование подмножеств языка [b]	++	++	++	++
1c	Принудительное применение строгой типизации	++	++	++	++
1d	Использование защитных методов реализации	o	+	++	++
1e	Использования установленных принципов проектирования	+	+	+	++
1f	Использование однозначного графического представления	+	++	++	++
1g	Использование руководства по стилю оформления программной документации	+	++	++	++
1h	Использованию соглашений об именовании	++	++	++	++

[a] Может потребоваться приемлемый компромисс данного подхода с другими методами в настоящем стандарте.

[b] Целями метода 1b являются:

– исключение неоднозначно определенных конструкций языка, которые могут быть по-разному интерпретированы разработчиками моделей, программистами, генераторами кодов или компиляторами;

– исключение конструкций языка, которые, как показывает опыт, легко приводят к ошибкам, например, присвоения в условиях или идентичность имен локальных и глобальных переменных;

– исключение конструкций языка, которые могут привести к необрабатываемым ошибкам во время выполнения.

[c] Цель метода 1c состоит в наложении принципов строгой типизации там, где они не присущи языку.

5.5 Результаты работы

5.5.1 План по обеспечению безопасности (уточненный)

В результате выполнения требований 5.4.1 – 5.4.7.

5.5.2 План верификации программного обеспечения

В результате выполнения требований 5.4.1 – 5.4.5 и 5.4.7.

5.5.3 Руководства по разработке и кодированию для языков моделирования и программирования

В результате выполнения требований 5.4.6 и 5.4.7.

5.5.4 Руководства по применению инструментальных средств

В результате выполнения требований 5.4.5 и 5.4.6.

6 Спецификация требований безопасности к программному обеспечению

6.1 Цели

Первой целью данной подстадии является формирование спецификации требований безопасности к программному обеспечению. Они выводятся из технической концепции системы безопасности и спецификации проекта системы.

Второй целью настоящего раздела является формирование подробных требований к программно-аппаратному интерфейсу, спроектированного в соответствии с требованиями раздела 7 ИСО 26262-4.

Третьей целью является проверка согласованности требований безопасности к программному обеспечению и требований к программно-аппаратному интерфейсу с технической концепцией системы безопасности и спецификацией проекта системы.

6.2 Общие положения

Технические требования к системе безопасности распределяются для аппаратных средств и программного обеспечения на стадии проектирования системы, представленной в разделе 7 ИСО 26262-4. Спецификация требований безопасности к программному обеспечению рассматривает ограничения аппаратных средств и влияние этих ограничений на программное обеспечение. Данная подстадия включает в себя спецификацию требований безопасности к программному обеспечению для поддержки последующих стадий проектирования.

6.3 Входная информация

6.3.1 Предварительные требования

Необходима следующая информация:

- техническая концепция системы безопасности в соответствии с 7.5.1 ИСО 26262-4;
- спецификация проекта системы в соответствии с 7.5.2 ИСО 26262-4;
- спецификация программно-аппаратного интерфейса в соответствии 7.5.3 ИСО 26262-4;
- план по обеспечению безопасности (уточненный) в соответствии с 5.5.1;
- план верификации программного обеспечения в соответствии с 5.5.2.

6.3.2 Дополнительная информация

Следующая информация может быть учтена:

- спецификация проекта аппаратных средств (см. 7.5.1 ИСО 26262-5);
- руководства по применению методов (из внешнего источника).

6.4 Требования и рекомендации

6.4.1 Требования безопасности к программному обеспечению рассматриваются для каждой функции, реализуемой программным обеспечением, отказ которой может привести к нарушению технических требований к системе безопасности, распределяемых программному обеспечению.

Пример – Функции, отказ которых может привести к нарушению требования безопасности, могут быть:

- ***функциями, которые обеспечивают системе достижение или поддержание безопасного состояния;***
- ***функциями, связанными с обнаружением, индикацией и обработкой неисправностей, связанных с безопасностью элементов аппаратного обеспечения;***
- ***функциями, связанными с обнаружением, оповещением и смягчением неисправности в самой программе.***

Примечание – К ним относятся как самоконтроль программного обеспечения операционной системы и специализированный самоконтроль программного обеспечения, обнаруживающего, оповещающего и обрабатывающего систематические ошибки в прикладном программном обеспечении;

- ***функциями, связанными с внутренними и внешними испытаниями автомобиля.***

Примечания

1 Внутренние тесты могут быть выполнены самой системой или с помощью других систем в сети автомобиля во время работы, а также перед и после стадии запуска транспортного средства.

2 К внешним испытаниям относятся испытания связанных с безопасностью функций или свойств в процессе производства или обслуживания;

- ***функциями, которые позволяют модификации программного обеспечения во время производства и обслуживания;***
- ***Функциями, связанными с рабочими характеристиками или критичными по времени операциями.***

6.4.2 Спецификация требований безопасности к программному обеспечению должна быть сформирована из технической концепции системы безопасности и проекта системы в соответствии с требованиями 7.4.1 и 7.4.5 ИСО 26262-4 и должна содержать:

a) спецификацию и менеджмент требований безопасности в соответствии с разделом 6 ИСО 26262-8;

b) специфицированные конфигурации системы и аппаратных средств.

Пример – Параметры конфигурации могут включать в себя регулятор усиления, полосу пропускания и предварительный делитель частоты тактового сигнала;

c) спецификацию программно-аппаратного интерфейса;

d) соответствующую требованиям спецификацию проекта аппаратных средств;

e) временные ограничения.

Пример – Время выполнения или реакции, определяемое требуемым временем отклика на уровне системы;

f) внешние интерфейсы.

Пример – Коммуникационные и пользовательские интерфейсы;

g) описание каждого режима работы транспортного средства, системы или аппаратных средств, оказывающих воздействие на программное обеспечение.

Пример – Режимами работы аппаратных средств устройств могут быть работа по умолчанию, инициализация, тестирование и расширенные режимы.

6.4.3 Если для требований безопасности к программному обеспечению применяется распределение УПБА, то должны соблюдаться требования раздела 5 ИСО 26262-9.

6.4.4 Спецификация программно-аппаратного интерфейса, формируемая в соответствии с требованиями раздела 7 ИСО 26262-4, должна быть подробно описана до уровня, позволяющего обеспечить корректное управление и использование аппаратных средств, и должна содержать описание каждой связанной с безопасностью зависимостью между аппаратными средствами и программным обеспечением.

6.4.5 Если кроме функций, для которых требования к безопасности определены в 6.4.1, с помощью встроенного программного обеспечения выполняются другие функции, то эти функции должны быть определены, либо на их спецификацию должны быть ссылки.

6.4.6 В соответствии с требованиями раздела 9 ИСО 26262-8 должна планироваться верификация требований безопасности к программному обеспечению и уточненной спецификации программно-аппаратного интерфейса.

6.4.7 Уточненная спецификация программно-аппаратного интерфейса должна быть проверена совместно с лицами, ответственными за разработку системы, аппаратных средств и программного обеспечения.

6.4.8 Требования безопасности к программному обеспечению и требования к уточненной спецификации программно-аппаратного интерфейса должны быть верифицированы в соответствии с требованиями разделов 6 и 9 ИСО 26262-8, чтобы показать свое:

a) соответствие и согласованность с техническими требованиями системы безопасности;

b) соответствие с проектом системы;

c) согласованность с аппаратно-программным интерфейсом.

6.5 Результаты работы

6.5.1 Спецификация требований безопасности к программному обеспечению

В результате выполнения требований 6.4.1 – 6.4.3 и 6.4.5.

6.5.2 Спецификация программно-аппаратного интерфейса (уточненная)

В результате выполнения требований 6.4.4.

Примечание – Данный результат работы совпадает с результатом работы 6.5.2 ИСО 26262-5.

6.5.3 План верификации программного обеспечения (уточненный)

В результате выполнения требований 6.4.6.

6.5.4 Отчет о верификации программного обеспечения системы безопасности

В результате выполнения требований 6.4.7 и 6.4.8.

7 Проектирование архитектуры программного обеспечения

7.1 Цели

Первой целью данной подстадии является разработка проекта архитектуры программного обеспечения, реализующая требования безопасности к программному обеспечению.

Второй целью данной подстадии является верификация проекта архитектуры программного обеспечения.

7.2 Общие положения

Проект архитектуры программного обеспечения описывает все компоненты программного обеспечения и их взаимодействия в иерархической структуре. В нем описываются статические аспекты, такие как интерфейсы и информационные каналы между всеми компонентами программного обеспечения, а также динамические аспекты, такие как последовательность процессов и временные характеристики.

Примечание – Проект архитектуры программного обеспечения не обязательно ограничивается одним микроконтроллером или ECU и связан с технической концепцией системы безопасности и проектом системы. В данном разделе также рассматривается архитектура программного обеспечения для каждого микроконтроллера.

Для разработки проекта архитектуры программного обеспечения реализуются как требования к программному обеспечению, связанные с безопасностью, так и все требования к программному обеспечению, не связанные с безопасностью. Следовательно, на данной подстадии связанные с безопасностью и не связанные с безопасностью требования рассматриваются в рамках единого процесса разработки.

Проект архитектуры программного обеспечения предусматривает средства для реализации требований безопасности к программному обеспечению и для управления сложностью разработки программного обеспечения.

7.3 Входная информация

7.3.1 Предварительные требования

Необходима следующая информация:

- план обеспечения безопасности (уточненный) в соответствии с 5.5.1;
- руководства по проектированию и кодированию для языков моделирования и программирования в соответствии с 5.5.3;
- спецификация программно-аппаратного интерфейса в соответствии с требованиями 7.5.3 ИСО 26262-4;
- спецификация требований безопасности к программному обеспечению в соответствии с 6.5.1;
- план верификации программного обеспечения (уточненный) в соответствии с 6.5.3;
- отчет о верификации программного обеспечения в соответствии с 6.5.4.

7.3.2 Дополнительная информация

Следующая информация может быть учтена:

- техническая концепция системы безопасности (см. с 7.5.1 ИСО 26262-4);
- спецификация проекта системы (см. 7.5.2 ИСО 26262-4);
- о доступных квалифицированных компонентах программного обеспечения (см. раздел 12 ИСО 26262-8);
- руководства по применению инструментальных средств в соответствии с 5.5.4;
- руководства по применению методов (из внешнего источника).

7.4 Требования и рекомендации

7.4.1 Для того, чтобы проект архитектуры программного обеспечения получал информацию, необходимую для обеспечения корректного и эффективного выполнения последующих действий по разработке, проект архитектуры программного обеспечения должен быть описан на соответствующих уровнях абстрагирования, используя представления для проекта архитектуры программного обеспечения, перечисленные в таблице 2.

Таблица 2 – Представления для проекта архитектуры программного обеспечения

Методы		УПБА			
		A	B	C	D
1	Неформальные представления	++	++	+	+
2	Полуформальные представления	+	++	++	++
3	Формальные представления	+	+	+	+

7.4.2 При разработке проекта архитектуры программного обеспечения необходимо рассмотреть следующее:

a) верефицируемость проекта архитектуры программного обеспечения.

Примечание – Это подразумевает двунаправленную прослеживаемость между проектом архитектуры программного обеспечения и требованиями к безопасности программного обеспечения;

b) пригодность программного обеспечения к конфигурируемости;

c) возможность разработки и реализации программных модулей;

d) тестируемость архитектуры программного обеспечения во время тестирования интеграции программного обеспечения;

e) удобство сопровождения проекта архитектуры программного обеспечения.

7.4.3 Для предотвращения отказов из-за высокой сложности проект архитектуры программного обеспечения должен обладать следующими свойствами в случае применения принципов, перечис-

ленных в таблице 3:

a) модульностью;

b) инкапсуляцией;

c) простотой.

Таблица 3 – Принципы проектирования архитектуры программного обеспечения

Свойства		УПБА			
		A	B	C	D
1a	Иерархическая структура компонентов программного обеспечения	++	++	++	++
1b	Ограниченный размер компонентов программного обеспечения [a]	++	++	++	++
1c	Ограниченный размер интерфейсов [a]	+	+	+	+
1d	Высокая связность внутри каждого компонента программного обеспечения [b]	+	++	++	++
1e	Ограниченная связность между компонентами программного обеспечения [a] [b] [c]	+	++	++	++
1f	Надлежащие свойства планирования	++	++	++	++
1g	Ограниченное использование прерываний [a] [d]	+	+	+	++

[a] В методах 1b, 1c, 1e и 1g «ограниченный» означает минимальный по сравнению с влиянием других аспектов проекта.

[b] Методы 1d и 1e могут, например, быть реализованы путем разделения функций, относящихся к способности идентифицировать, инкапсулировать и манипулировать теми частями программного обеспечения, которые имеют отношение к определенной концепции, цели, задаче или назначению.

[c] Метод 1e связан с ограничениями внешних связей компонентов программного обеспечения.

[d] Все используемые прерывания должны быть реализованы на основе приоритетов.

Примечание – Поскольку методы, приведенные в таблице 3, не являются взаимоисключающими, может быть необходим приемлемый компромисс между этими методами.

7.4.4 Проект архитектуры программного обеспечения должен быть разработан до уровня, где определены все программные модули.

7.4.5 Проект архитектуры программного обеспечения должен описывать:

a) статические аспекты проекта компонентов программного обеспечения.

Примечания

1 Статическими аспектами проекта являются:

- структура программного обеспечения, включая ее иерархические уровни;
- логическая последовательность обработки данных;
- типы данных и их характеристики;
- внешние интерфейсы компонентов программного обеспечения;
- внешние интерфейсы программного обеспечения;
- ограничения, в том числе область применения архитектуры и внешние зависимости.

2 В случае разработки на основе модели, моделирование структуры является неотъемлемой частью всего процесса моделирования;

b) динамические аспекты проекта компонентов программного обеспечения.

Примечания

1 Динамическими аспектами проекта являются:

- функциональность и поведение;
- поток управления и параллелизм процессов;
- поток данных между компонентами программного обеспечения;
- поток данных на внешних интерфейсах;
- временные ограничения.

2 Для определения динамических характеристик (например, задач, временных интервалов и прерываний) рассматриваются различные режимы работы (например, включение питания, останов, нормальная работа, калибровка и диагностика).

3 Для описания динамического поведения (например, задач, временных интервалов и прерываний) специфицируются коммуникационные отношения и их распределение в аппаратных средствах системы (например, процессора и каналов связи).

7.4.6 Каждый связанный с безопасностью компонент программного обеспечения должен быть

классифицирован по следующим категориям:

a) вновь разрабатываемый;

b) используемый повторно с модификациями или

c) используемый повторно без изменений.

7.4.7 Связанные с безопасностью компоненты программного обеспечения, которые были вновь разработаны и используются повторно с модификациями, должны быть разработаны в соответствии с требованиями настоящего стандарта.

Примечание – В этих случаях требования раздела 12 ИСО 26262-8 не применяются.

7.4.8 Связанные с безопасностью компоненты программного обеспечения, которые используются повторно без изменений, должны быть квалифицированы в соответствии с требованиями раздела 12 ИСО 26262-8.

Примечание – Использование квалифицированных компонентов программного обеспечения не влияет на применимость требований разделов 10 и 11. Тем не менее, некоторые действия, описанные в разделах 8 и 9, могут быть опущены.

7.4.9 Требования безопасности к программному обеспечению должны быть распределены для программных компонентов. В результате каждый компонент программного обеспечения должен быть разработан в соответствии с самым высоким значением УПБА среди распределенных для него требований.

Примечание – После этого распределения может быть необходимо дальнейшее уточнение требований к программному обеспечению системы безопасности.

7.4.10 Если встроенное программное обеспечение реализуется с использованием компонентов программного обеспечения с различными значениями УПБА или с использованием компонентов программного обеспечения, связанных с безопасностью и не связанных с безопасностью, то все встроенное программное обеспечение должно рассматриваться в соответствии с самым высоким значением УПБА, если программные компоненты не отвечают критериям совместимости в соответствии с требованиями раздела 6 ИСО 26262-9.

7.4.11 Если используется разделение программного обеспечения (см. приложение D) для устранения влияния между компонентами программного обеспечения, то следует обеспечить, чтобы:

a) общие ресурсы использовались таким образом, чтобы отсутствовало влияние между разделами программного обеспечения.

Примечания

1 Задачи внутри раздела программного обеспечения не свободны от влияния со стороны друг друга.

2 Один раздел программного обеспечения не может изменить ни код или данные другого раздела программного обеспечения, ни команду неразделяемых ресурсов других разделов программного обеспечения.

3 Сервис, получаемый из общих ресурсов одним разделом программного обеспечения, не может зависеть от сервиса, получаемого другим разделом программного обеспечения. Он включает в себя такие параметры выполнения соответствующих ресурсов, как периодичность, задержка, уровень случайных искажений и длительность запланированного доступа к ресурсу;

b) разделение программного обеспечения поддерживается специальными функциями аппаратных средств или эквивалентными средствами (данное требование распространяется на значение УПБА, равное D, в соответствии с 4.3);

c) часть программного обеспечения, которая реализует разделение программного обеспечения, разрабатывается в соответствии с тем же или с более высоким значением УПБА, чем наивысшее значение УПБА, назначенное требованиями для разделов программного обеспечения.

Примечание – В общем случае операционная система предоставляет или поддерживает программы для создания разделов;

d) верификация разделения программного обеспечения в процессе интеграции и тестирования программного обеспечения (в соответствии с требованиями раздела 10) не выполняется.

7.4.12 Если при реализации требований безопасности к программному обеспечению предполагается отсутствие влияния или достаточная независимость между компонентами программного обеспечения, то анализ зависимых отказов осуществляется в соответствии с требованиями раздела 7 ИСО 26262-9.

7.4.13 На уровне архитектуры программного обеспечения в соответствии с требованиями раздела 8 ИСО 26262-9 осуществляется анализ системы безопасности для того, чтобы:

– определить или подтвердить связанные с безопасностью части программного обеспечения; а также

– поддержать спецификацию и проверку эффективности механизмов безопасности.

Примечание – Механизмы безопасности могут быть предназначены как для охвата случайных отказов аппаратных средств, так и сбоев программного обеспечения.

7.4.14 Для определения необходимых механизмов обеспечения безопасности программного обеспечения на уровне архитектуры программного обеспечения на основе результатов анализа безопасности в соответствии с требованиями 7.4.13 должны быть применены механизмы для обнаружения ошибок, перечисленные в таблице 4.

Примечание – Если технические требования к системе безопасности не распределяются непосредственно для программного обеспечения, то использование механизмов безопасности программного обеспечения рассматривается на уровне системы для анализа возможного воздействия на поведение системы.

Таблица 4 – Механизмы обнаружения ошибок на уровне архитектуры программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Проверки диапазона входных и выходных данных	++	++	++	++
1b	Проверка достоверности [a]	+	+	+	++
1c	Обнаружение ошибок в данных [b]	+	+	+	+
1d	Внешние средства мониторинга [c]	o	+	+	++
1e	Мониторинг потока управления	o	+	++	++
1f	Разнообразие в проекте программного обеспечения	o	o	+	++

[a] Проверки достоверности могут включать в себя использование эталонной модели желаемого поведения, проверки утверждений или сравнение сигналов от разных источников.

[b] Типы методов, которые могут быть использованы для обнаружения ошибок в данных, включают в себя коды обнаружения ошибок и сегментирование памяти данных.

[c] Внешним средством мониторинга может быть, например, СИС или иной элемент программного обеспечения, выполняющий функции сторожевого таймера.

7.4.15 Данное требование распространяется на значения УПБА (A), (B), C и D в соответствии с 4.3. Для спецификации необходимых механизмов безопасности программного обеспечения на уровне архитектуры программного обеспечения на основе результатов анализа безопасности, выполненного в соответствии с требованиями 7.4.13, должны применяется механизмы для обработки ошибок, представленные в таблице 5.

Примечания

1 Если технические требования к системе безопасности не распределяются непосредственно для программного обеспечения, то использование механизмов безопасности программного обеспечения рассматривается на уровне системы для анализа возможного воздействия на поведение системы.

2 Анализ возможных опасностей архитектуры программного обеспечения из-за аппаратных средств описан в ИСО 26262-5.

Таблица 5 – Механизмы обработки ошибок на уровне архитектуры программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Статический механизм восстановления [a]	+	+	+	+
1b	Постепенное снижение эффективности [b]	+	+	++	++
1c	Независимое постоянное резервирование [c]	o	o	+	++
1d	Коды исправления данных	+	+	+	+

[a] Статические механизмы восстановления могут включать в себя использование блоков восстановления, восстановление предыдущего состояния файла, восстановление без возврата и восстановление посредством повторения.

[b] Постепенное снижение эффективности на уровне программного обеспечения связано с расположением по приоритетам реализуемых функций для минимизации негативного влияния возможных отказов на функциональную безопасность.

[c] Независимое постоянное резервирование может быть реализовано, используя разнородное программное обеспечение в каждом параллельном канале.

7.4.16 Если в результате проектирования архитектуры программное обеспечение появились новые опасности, еще не охваченные существующей целью безопасности, то они должны быть учтены и оценены при анализе опасностей и оценке рисков в соответствии с требованиями процесса управления изменениями, представленными в разделе 8 ИСО 26262-8.

Примечание – Вновь выявленные опасности и неучтенные в цели безопасности, как правило, являются нефункциональными опасностями. Нефункциональные опасности выходят за рамки области применения настоящего стандарта, но они могут быть при анализе опасностей и оценке рисков снабжены следующим пояснением «Данной опасности значение УПБА не назначается, поскольку она выходит за рамки области применения настоящего стандарта». Тем не менее, значение УПБА может быть назначено в качестве рекомендации.

7.4.17 Должна быть выполнена верхняя оценка ресурсов, необходимых для встроенного программного обеспечения, в том числе:

a) времени выполнения;

b) объема памяти.

Пример – Оперативная память для стеков и динамически распределяемых областей памяти, ПЗУ для программ и энергонезависимых данных;

c) коммуникационных ресурсов.

7.4.18 Проект архитектуры программного обеспечения должен быть верифицирован в соответствии с требованиями раздела 9 ИСО 26262-8 и с помощью методов верификации проекта архитектуры программного обеспечения, перечисленных в таблице 6, для демонстрации следующих свойств:

a) соответствия требованиям безопасности к программному обеспечению;

b) совместимости с целевыми аппаратными средствами.

Примечание – Они включает в себя ресурсы, указанные в 7.4.17;

c) соблюдения руководств по проектированию.

Таблица 6 – Методы верификации проекта архитектуры программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Сквозной контроль проекта [a)]	++	+	o	o
1b	Проверка (контроль) проекта [a)]	+	++	++	++
1c	Моделирование динамических частей проекта [b)]	+	+	+	++
1d	Генерация прототипа	o	o	+	++
1e	Формальная верификация	o	o	+	+
1f	Анализ потока управления [c)]	+	+	++	++
1g	Анализ потока данных [c)]	+	+	++	++

[a)] В случае разработки на основе модели эти методы могут быть применены к модели.

[b)] Метод 1c требует использования исполняемых моделей для динамических частей архитектуры программного обеспечения.

[c)] Анализ потока управления и/или данных может быть ограничен связанными с безопасностью компонентами и их интерфейсами.

7.5 Результаты работы

7.5.1 Спецификация проекта архитектуры программного обеспечения

В результате выполнения требований 7.4.1 – 7.4.6, 7.4.9, 7.4.10, 7.4.14, 7.4.15 и 7.4.17.

7.5.2 План обеспечения безопасности (уточненный)

В результате выполнения требований 7.4.7.

7.5.3 Спецификация требований безопасности к программному обеспечению (уточненная)

В результате выполнения требований 7.4.9.

7.5.4 Отчет по анализу безопасности

В результате выполнения требований 7.4.13.

7.5.5 Отчет по анализу зависимых отказов

В результате выполнения требований 7.4.12.

7.5.6 Отчет по верификации программного обеспечения (уточненный)

В результате выполнения требований 7.4.18.

8 Проектирование и реализация модуля программного обеспечения

8.1 Цели

Первой целью настоящей подстадии является спецификация модулей программного обеспечения в соответствии с проектом архитектуры программного обеспечения и соответствующими требованиями безопасности к программному обеспечению.

Второй целью настоящей подстадии является реализация модулей программного обеспечения в соответствии со спецификацией.

Третьей целью настоящей подстадии является статическая верификация проекта модулей программного обеспечения и ее выполнение.

8.2 Общие положения

На основе проекта архитектуры программного обеспечения разрабатывается детальный проект модулей программного обеспечения. Такой детальный проект будет реализован в виде модели или непосредственно в виде исходного кода в соответствии с руководствами по моделированию или кодированию соответственно. Детальный проект и его реализация верифицируются статически прежде, чем перейти к стадии тестирования модуля программного обеспечения. Связанные с реализацией характеристики могут быть достигнуты на уровне исходного кода, если используется руководство по разработке кода. Если используется разработка на основе модели с автоматической генерацией кода, то эти характеристики применяются к модели и нет необходимости их применять к исходному коду.

При разработке проекта отдельного модуля программного обеспечения реализуются как требования безопасности к программному обеспечению, так и все не связанные с безопасностью требования. Следовательно, на данной подстадии связанные с безопасностью и н связанные с безопасностью требования рассматриваются в рамках единого процесса разработки.

Реализация модулей программного обеспечения включает в себя генерацию исходного кода и трансляцию в объектный код.

8.3 Входная информация

8.3.1 Предварительные требования

Следующая информация должна быть доступна:

- руководства по разработке и кодированию для языков моделирования и программирования в соответствии с 5.5.3;
- план верификации программного обеспечения (уточненный) в соответствии с 6.5.3;
- спецификация проекта архитектуры программного обеспечения в соответствии с 7.5.1;
- план обеспечения безопасности (уточненный) в соответствии с 7.5.2;
- спецификация требований безопасности к программному обеспечению (уточненная) в соответствии с 7.5.3;
- отчет по верификации программного обеспечения (уточненный) в соответствии с 7.5.6.

8.3.2 Дополнительная информация

Следующая информация может быть учтена:

- техническая концепция системы безопасности (см. 7.5.1 ИСО 26262-4);
- спецификация проекта системы (см. 7.5.2 ИСО 26262-4);
- руководства по применению инструментальных средств в соответствии с 5.5.4;
- спецификация программно-аппаратного интерфейса (уточненная) (см. 6.5.2);
- отчет по анализу безопасности в соответствии с 7.5.4;
- руководства по применению методов (из внешнего источника).

8.4 Требования и рекомендации

8.4.1 Требования настоящего подраздела должны соблюдаться, если модуль программного обеспечения связан с безопасностью.

Примечание – «Связанный с безопасностью» означает, что модуль реализует требования безопасности или что критерии совместимости (см. раздел 6 ИСО 26262-9) этого модуля с другими модулями не выполнены.

8.4.2 Для того, чтобы проект модуля программного обеспечения получал информацию, необходимую для обеспечения корректного и эффективного выполнения последующих действий по его разработке, проект модуля программного обеспечения должен быть описан с использованием представлений, приведенных в таблице 7.

Таблица 7 – Представления для проекта модуля программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Естественный язык	++	++	++	++
1b	Неформальные представления	++	++	+	+
1c	Полуформальные представления	+	++	++	++
1d	Формальные представления	+	+	+	+

Примечание – Если используется разработка на основе модели с автоматической генерацией кода, то эти способы представления разработки модуля программного обеспечения применяются к модели, которая служит основой для генерации кода.

8.4.3 Спецификация модулей программного обеспечения должна описывать функциональное поведение и внутреннее проектирование до уровня детализации, необходимой для их реализации.

Пример – Внутреннее проектирование может включать в себя ограничения на использование регистров и запоминающего устройства.

8.4.4 Принципы проектирования для разработки и реализации модуля программного обеспечения на уровне исходного кода, которые перечислены в таблице 8, должны применяться для достижения следующих свойств:

a) корректного порядка выполнения подпрограмм и функций внутри программных модулей, основанных на проекте архитектуры программного обеспечения;

b) согласованности интерфейсов между модулями программного обеспечения;

c) корректности потока данных и управления между и внутри программных модулей;

d) простоте;

e) читаемости и понимаемости;

f) надежности.

Пример – Методы предотвращения недостоверных значений, ошибок исполнения, деления на ноль, и ошибок потока данных и потока управления;

g) пригодности программного обеспечения для модификации;

h) тестируемости.

Таблица 8 – Принципы проектирования для разработки и реализации модуля программного обеспечения

Свойства		УПБА			
		A	B	C	D
1a	Одна точка входа и одна точка выхода в подпрограммах и функциях [a]	++	++	++	++
1b	Не использовать динамические объекты или переменные, либо выполнять тестирование в неавтономном режиме при их создании [a] [b]	+	++	++	++
1c	Инициализация переменных	++	++	++	++
1d	Не использовать многократно имена переменных [a]	+	++	++	++
1e	Избегать глобальных переменных либо обосновывать их использование [a]	+	+	++	++
1f	Ограничивать использование указателей [a]	o	+	+	++
1g	Исключать неявные преобразования типов [a] [b]	+	++	++	++
1h	Исключать скрытые потоки данных или потоки управления [c]	+	++	++	++
1i	Исключать безусловные переходы [a] [b] [c]	++	++	++	++
1j	Исключать рекурсии	+	+	++	++

[a] Методы 1a, 1b, 1d, 1e, 1f, 1g и 1i нельзя применять для представлений графического моделирования, используемых при разработке на основе модели.

[b] Методы 1g и 1i не применяются при программировании на ассемблере.

[c] Методы 1h и 1i сокращают возможность для моделирования потока данных и потока управления из-за переходов или глобальных переменных.

Примечание – MISRA C (для языка C) [3] включает многие из методов, перечисленных в таблице 8.

8.4.5 Разработка и реализация модуля программного обеспечения должны быть верифицированы в соответствии с требованиями раздела 9 ИСО 26262-8, применяя методы верификации, приведенные в таблице 9, чтобы продемонстрировать:

a) соответствие со спецификацией программно-аппаратного интерфейса (в соответствии с требованиями 6.4.10 ИСО 26262-5);

b) выполнение требований безопасности к программному обеспечению при их распределении модулям программного обеспечения (в соответствии с требованиями 7.4.9), используя прослеживание;

c) соответствие исходного кода его проектной спецификации

Примечание – В случае разработки на основе модели требование перечисления c) все еще применяется;

d) соответствие исходного кода принципам кодирования (см. 5.5.3);

e) совместимость реализации модуля программного обеспечения с целевыми аппаратными средствами.

Таблица 9 – Методы верификации разработки и реализации модуля программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Сквозной контроль [a]	++	+	o	o
1b	Проверка (контроль) [a]	+	++	++	++
1c	Полуформальная верификация	+	+	++	++
1d	Формальная верификация	o	o	+	+
1e	Анализ потока управления [b] [c]	+	+	++	++
1f	Анализ потока данных [b] [c]	+	+	++	++
1g	Статический анализ кода	+	++	++	++
1h	Семантический анализ кода [d]	+	+	+	+

[a] В случае разработки программного обеспечения на основе модели разработка и реализация модуля программного обеспечения может быть верифицирована на уровне модели.
[b] Методы 1e и 1f могут быть применены на уровне исходного кода. Эти методы применимы как для руководства разработкой кода, так и для разработки на основе модели.
[c] Методы 1e и 1f могут быть частью методов 1d, 1g или 1h.
[d] Метод 1h используется для математического анализа исходного кода, используя абстрактное представление возможных значений для переменных. Для этого не нужно транслировать и выполнять исходный код.

Примечание – В таблице 9 перечислены только статические методы верификации. Динамические методы верификации (например, методы тестирования) рассмотрены в таблицах 10 – 12.

8.5 Результаты работы

8.5.1 Спецификация проекта модулей программного обеспечения

В результате выполнения требований 8.4.2 – 8.4.4.

Примечание – В случае разработки на основе модели, модель реализации и документально оформленная поддержка, используя методы, перечисленные в таблице 8, специфицируют модули программного обеспечения.

8.5.2 Реализация модулей программного обеспечения

В результате выполнения требований 8.4.4.

8.5.3 Отчет о верификации программного обеспечения (уточненный)

В результате выполнения требований 8.4.5.

9 Тестирование модуля программного обеспечения

9.1 Цель

Целью требований данной подстадии является демонстрация того, что программный модуль удовлетворяет спецификации проекта модуля программного обеспечения и не содержит нежелатель-

ную функциональность.

9.2 Общие положения

Спецификация проекта модуля программного обеспечения устанавливает для него процедуру тестирования и испытания модуля программного обеспечения проводятся в соответствии с этой процедурой.

9.3 Входная информация

9.3.1 Предварительные требования

Следующая информация должна быть доступна:

- спецификация программно-аппаратного интерфейса (уточненная) в соответствии с 6.5.2;
- план верификации программного обеспечения (уточненный) в соответствии с 7.5.2;
- спецификация проекта модуля программного обеспечения в соответствии с 8.5.1;
- реализация модуля программного обеспечения в соответствии с 8.5.2;
- отчет о верификации программного обеспечения (уточненный) в соответствии с 8.5.3.

9.3.2 Дополнительная информация

Следующая информация может быть учтена:

- руководства по применению инструментальных средств в соответствии с 5.5.4;
- руководства по применению методов (из внешнего источника).

9.4 Требования и рекомендации

9.4.1 Требования настоящего подраздела должны соблюдаться, если модуль программного обеспечения связан с безопасностью.

Примечание – «Связанный с безопасностью» означает, что модуль реализует требования безопасности или что критерии совместимости этого модуля с другими модулями не выполнены.

9.4.2 Тестирование модуля программного обеспечения должно быть спланировано, специфицировано и выполнено в соответствии с требованиями раздела 9 ИСО 26262-8.

Примечания

1 В соответствии с определениями раздела 9 ИСО 26262-8 объектами испытаний при тестировании модуля программного обеспечения являются программные модули.

2 При разработке программного обеспечения на основе модели соответствующие части модели реализации также являются объектами планирования для испытания. В зависимости от выбранного процесса разработки программного обеспечения объектами испытаний может быть код, полученный из этой модели, или сама модель.

9.4.3 Должны применяться методы тестирования модулей программного обеспечения, перечисленные в таблице 10, чтобы продемонстрировать, что модуль программного обеспечения достигает:

a) соответствия со спецификацией проекта модуля программного обеспечения (в соответствии с требованиями раздела 8);

b) соответствия спецификации программно-аппаратного интерфейса (в соответствии с требованиями 6.4.10 ИСО 26262-5);

c) заданную функциональность;

d) уверенность в отсутствии непреднамеренной функциональности;

e) надежность.

Пример – Отсутствие недостижимого программного обеспечения, эффективность механизмов обнаружения и обработки ошибок;

f) достаточный уровень обеспечения ресурсами для поддержания его функциональности.

Таблица 10 – Методы тестирования модуля программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Тестирование на основе требований [a)]	++	++	++	++
1b	Тестирование интерфейса	++	++	++	++
1c	Испытания с введением неисправностей [b)]	+	+	+	++

Окончание таблицы 10

Методы		УПБА			
		A	B	C	D
1d	Тестирование используемых ресурсов [c]	+	+	+	++
1e	Сравнительное испытание между моделью и кодом, если оно применимо [d]	+	+	++	++

[a] Данный метод реализуется на основе требований к программному обеспечению на уровне модуля.
[b] Данный метод включает в себя введение произвольных неисправностей (например, путем повреждения значений переменных, введения мутаций кода или повреждения значений регистров процессора).
[c] Некоторые аспекты теста использования ресурсов могут быть оценены правильно, только когда тестирование модуля программного обеспечения выполняется на целевых аппаратных средствах или если эмулятор целевого процессора поддерживает тесты использования ресурсов.
[d] Данный метод требует модель, которая может имитировать функциональные возможности программных модулей. На вход модели и кода подаются одинаковые данные, а результаты сравниваются друг с другом.

9.4.4 Для получения подходящих тестов для тестирования модуля программного обеспечения в соответствии с требованиями 9.4.3 должны быть использованы методы, перечисленные в таблице 11.

Таблица 11 – Методы получения тестов для тестирования модуля программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Анализ требований	++	++	++	++
1b	Генерация и анализ классов эквивалентности [a]	+	++	++	++
1c	Анализ граничных значений [b]	+	++	++	++
1d	Предположение ошибок [c]	+	+	+	+

[a] Классы эквивалентности могут быть определены на основе разделения входных и выходных данных так, чтобы тестовые значения выбрались из каждого класса.
[b] Данный метод применяется к интерфейсам, у которых значения приближаются и пересекают границы, а также выходят за границы диапазона значений.
[c] Испытания с предположением ошибок могут быть основаны на данных, полученных из «обобщения опыта» и экспертной оценки.

9.4.5 Для оценки полноты тестов и демонстрации того, что непреднамеренная функциональность отсутствует, должен быть определен охват требований на уровне модуля программного обеспечения и должно быть измерено структурное покрытие с помощью метрик, перечисленных в таблице 12. Если достигаемое структурное покрытие считается недостаточным, то либо должны быть специфицированы дополнительные тесты, либо должно быть предусмотрено обоснование.

Примеры

1 Анализ структурного покрытия может выявить недостатки в контрольных примерах при тестировании на основе требований, несоответствия в требованиях, неисполняемый код, выводящий из строя код или непреднамеренную функциональность.

2 Для уровня покрытия, достигаемого на основе принятого неисполняемого кода (например, кода для отладки) или сегментов кода в зависимости от различных конфигураций программного обеспечения, может быть предоставлено обоснование; либо не охваченный покрытием код может быть верифицирован с помощью дополнительных методов (например, с помощью проверок).

Таблица 12 – Метрики структурного покрытия на уровне модуля программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Покрытие операторов	++	++	+	+
1b	Покрытие ветвей	+	++	++	++
1c	Изменение условий / Покрытие результатов решений (MC/DC)	+	+	+	++

Примечания

1 Структурное покрытие может быть определено путем использования соответствующих программных инструментальных средств.

2 В случае разработки на основе модели анализ структурного покрытия может быть выполнен на уровне модели с использованием аналогичных метрик структурного покрытия для моделей.

3 Если для определения степени покрытия используется код измерительного средства, то, может быть, необходимо показать, что измерительные средства никак не влияют на результаты испытаний. Это может быть сделано путем повторения испытаний без кода измерительного средства.

9.4.6 Тестовая среда для тестирования модуля программного обеспечения должна быть как можно более близка к целевой среде. Если тестирование модуля программного обеспечения выполняется не в целевой среде, то должны быть проанализированы различия исходного и объектного кодов, а также различия между тестовой средой и целевой средой для того, чтобы специфицировать дополнительные тесты в целевой среде в течение последующих стадий тестирования.

Примечания

1 Различия между тестовой средой и целевой средой могут возникнуть в исходном коде или в объектном коде, например, за счет разной разрядности слов данных и адресных слов процессоров.

2 В зависимости от объема тестов для выполнения модуля программного обеспечения используется соответствующая тестовая среда (например, целевой процессор, процессор эмулятор или разрабатываемая система).

3 Тестирование модуля программного обеспечения может быть выполнено в различных средах, например:

- тестирование с моделью в контуре обратной связи;
- тестирование с программой в контуре обратной связи;
- тестирование с процессором в контуре обратной связи;
- программно-аппаратное тестирование.

4 При разработке на основе модели тестирование модуля программного обеспечения может быть выполнено на уровне модели, сопровождаемое сравнением соответствующих результатов испытаний модели и объектного кода. Для обеспечения эквивалентности поведения модели в отношении целей тестирования и автоматически сгенерированного кода используется сравнительное тестирование.

9.5 Результаты работы

9.5.1 План верификации программного обеспечения (уточненный)

В результате выполнения требований 9.4.2 – 9.4.6.

9.5.2 Спецификация верификации программного обеспечения

В результате выполнения требований 9.4.2 и 9.4.4 – 9.4.6.

9.5.3 Отчет о верификации программного обеспечения (уточненный)

В результате выполнения требований 9.4.2.

10 Интеграция и тестирование программного обеспечения

10.1 Цели

Первая цель данной подстадии состоит в обеспечении интеграции элементов программного обеспечения.

Вторая цель данной подстадии состоит в демонстрации того, что проект архитектуры программного обеспечения реализован встроенным программным обеспечением.

10.2 Общие положения

На данной подстадии тестируются определенные уровни интеграции и интерфейсы между элементами программного обеспечения на соответствие проекту архитектуры программного обеспечения. Шаги интеграции и тестирования элементов программного обеспечения непосредственно соответствуют иерархической архитектуре программного обеспечения.

Встроенное программное обеспечение может состоять из связанных и не связанных с безопасностью элементов программного обеспечения.

10.3 Входная информация

10.3.1 Предварительные требования

Следующая информация должна быть доступна:

- спецификация программно-аппаратного интерфейса (уточненная) в соответствии с 6.5.2;
- спецификация проекта архитектуры программного обеспечения в соответствии с 7.5.1;
- план по обеспечению безопасности (уточненный) в соответствии с 7.5.2;
- реализация модуля программного обеспечения в соответствии с 8.5.2;
- план верификации программного обеспечения (уточненный) в соответствии с 9.5.1;
- спецификация верификации программного обеспечения в соответствии с 9.5.2;
- отчет о верификации программного обеспечения (уточненный) в соответствии с 9.5.3.

10.3.2 Дополнительная информация

Следующая информация может быть учтена:

– о доступных квалифицированных компонентах программного обеспечения (см. раздел 12 ИСО 26262-8;

– отчет о квалификации инструментального программного обеспечения в соответствии с 11.5.2 ИСО 26262-8;

– руководства по применению инструментальных средств в соответствии с 5.5.4;

– руководства по применению методов (из внешнего источника).

10.4 Требования и рекомендации

10.4.1 План интеграции программного обеспечения должен описывать шаги иерархической интеграции отдельных модулей программного обеспечения в компоненты программного обеспечения, пока не будет полностью интегрировано встроенное программное обеспечение, и должен рассматривать:

a) функциональные зависимости, которые важны для интеграции программного обеспечения, а также

b) зависимости между интеграцией программного обеспечения и программно-аппаратной интеграцией.

Примечание – При разработке, основанной на модели, интеграция программного обеспечения может быть заменена на интеграцию на уровне модели с последующей автоматической генерацией кода из интегрированной модели.

10.4.2 Тестирование интеграции программного обеспечения должно быть спланировано, специфицировано и выполнено в соответствии с требованиями раздела 9 ИСО 26262-8.

Примечания

1 На основании определений раздела 9 ИСО 26262-8 объектами испытаний интеграции программного обеспечения являются компоненты программного обеспечения.

2 При разработке, основанной на модели, объектами испытаний могут быть модели компонентов программного обеспечения.

10.4.3 Должны применяться методы тестирования интеграции программного обеспечения, приведенные в таблице 13, чтобы продемонстрировать, что как компоненты программного обеспечения, так и встроенное программное обеспечение достигают:

a) соответствия с проектом архитектуры программного обеспечения в соответствии с требованиями раздела 7;

b) соответствия со спецификацией программно-аппаратного интерфейса в соответствии с требованиями раздела 7 ИСО 26262;

c) заданной функциональности;

d) надежности.

Пример – Отсутствие недостижимого программного обеспечения; эффективность обнаружения и обработки ошибок;

f) достаточного уровня обеспечения ресурсами для поддержания их функциональности.

Таблица 13 – Методы тестирования интеграции программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Тестирование на основе требований [a)]	++	++	++	++
1b	Тестирование интерфейса	++	++	++	++
1c	Испытания с введением неисправностей [b)]	+	+	+	++
1d	Тестирование используемых ресурсов [c), d)]	+	+	+	++

Окончание таблицы 13

Методы		УПБА			
		A	B	C	D
1e	Сравнительное испытание между моделью и кодом, если оно применимо [e]	+	+	++	++

[a] Данный метод реализуется на основе требований к программному обеспечению на уровне архитектуры.
[b] Они включают в себя введение произвольных неисправностей для тестирования механизмов безопасности (например, путем повреждения программного обеспечения либо компонентов аппаратных средств).
[c] Для обеспечения выполнения требований проектом архитектуры аппаратных средств с достаточной устойчивостью должны быть определены свойства, такие как средняя и максимальная производительность процессора, минимальное или максимальное время выполнения, использование памяти (например, ОЗУ для стека и динамически распределяемых областей памяти, ПЗУ для программ и данных) и пропускная способность коммуникационных линий (например, шин данных).
[d] Некоторые аспекты теста использования ресурсов могут быть оценены правильно, только когда тестирование интеграции программного обеспечения выполняется на целевых аппаратных средствах или если эмулятор целевого процессора поддерживает тесты использования ресурсов.
[e] Данный метод требует модель, которая может имитировать функциональные возможности компонентов программного обеспечения. На вход модели и кода подаются одинаковые данные, а результаты сравниваются друг с другом.

10.4.4 Для получения подходящих тестов для методов тестирования интеграции программного обеспечения в соответствии с требованиями 10.4.3 должны быть использованы методы, перечисленные в таблице 14.

Таблица 14 – Методы получения тестов для тестирования интеграции программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Анализ требований	++	++	++	++
1b	Генерация и анализ классов эквивалентности [a]	+	++	++	++
1c	Анализ граничных значений [b]	+	++	++	++
1d	Предположение ошибок [c]	+	+	+	+

[a] Классы эквивалентности могут быть определены на основе разделения входных и выходных данных так, чтобы тестовые значения выбирались из каждого класса.
[b] Данный метод применяется к параметрам или переменным, значения которых приближаются и пересекают границы, а также выходят за границы диапазона значений.
[c] Испытания с предположением ошибок могут быть основаны на данных, полученных из «обобщения опыта» и экспертной оценки.

10.4.5 Для оценки полноты тестов и получения уверенности в том, что непреднамеренная функциональность отсутствует, должен быть определен охват требований тестами на уровне архитектуры программного обеспечения. При необходимости должны быть специфицированы дополнительные тесты, либо должно быть предусмотрено обоснование.

10.4.6 Данное требование распространяется на значения УПБА (A), (B), C и D в соответствии с 4.3. Для оценки полноты тестов и получения уверенности в том, что непреднамеренная функциональность отсутствует, должно быть измерено структурное покрытие в соответствии с метриками, перечисленными в таблице 15. Если достигнутое структурное покрытие считается недостаточным, то либо должны быть специфицированы дополнительные тесты, либо должно быть предусмотрено обоснование.

Пример – Анализ структурного покрытия может выявить недостатки в контрольных примерах при тестировании на основе требований, несоответствия в требованиях, неисполняемый код, выводящий из строя код или непреднамеренную функциональность.

Таблица 15 – Метрики структурного покрытия на уровне архитектуры программного обеспечения

Методы		УПБА			
		A	B	C	D
1a	Покрытие функций [a]	++	++	+	+
1b	Охват вызовов [b]	+	++	++	++

[a] Метод 1a оценивает процент выполненных функций в программном обеспечении. Эта величина может быть получена применением соответствующей стратегии интеграции программного обеспечения.
[b] Метод 1b оценивает процент выполненных вызовов функций программного обеспечения.

Примечания

1 Структурное покрытие может быть определено путем использования соответствующих программных средств.

2 В случае разработки на основе модели тестирование архитектуры программного обеспечения может быть выполнено на уровне модели с использованием аналогичных метрик структурного покрытия для моделей.

10.4.7 Необходимо убедиться, что встроенное программное обеспечение, которое должно быть включено как часть выпускаемых изделий в соответствии с требованиями раздела 11 ИСО 26262-4, содержит все специфицированные функции и содержит только такие неспецифицированные функции, которые не влияют на соответствие требованиям безопасности к программному обеспечению.

Пример – В этом контексте неспецифицированные функции включают в себя код, используемый для отладки, или измерительные средства.

Примечание – Если может быть обеспечена дезактивация таких неспецифицированных функций, то это является приемлемым способом соответствия данному требованию. В противном случае удаление такого кода является изменением (см. раздел 8 ИСО 26262-8).

10.4.8 Тестовая среда для тестирования интеграции программного обеспечения должна быть как можно более близка к целевой среде. Если тестирование интеграции программного обеспечения не выполняется в целевой среде, то должны быть проанализированы различия исходного и объектного кодов, а также различия между тестовой и целевой средами для того, чтобы специфицировать дополнительные тесты в целевой среде в течение последующих стадий тестирования.

Примечания

1 Различия между тестовой и целевой средами могут возникнуть в исходном коде или в объектном коде, например, за счет разной разрядности слов данных и адресных слов процессоров.

2 В зависимости от объема тестов и иерархического уровня интеграции для выполнения элементов программного обеспечения используется соответствующая тестовая среда (например, целевой процессор для окончательной интеграции или эмулятор процессора или разрабатываемая система для предыдущих шагов интеграции).

3 Тестирование интеграции программного обеспечения может быть выполнено в различных средах, например:

- тестирование с моделью в контуре обратной связи;
- тестирование с программой в контуре обратной связи;
- тестирование с процессором в контуре обратной связи;
- программно-аппаратное тестирование.

10.5 Результаты работы

10.5.1 План верификации программного обеспечения (уточненный)
В результате выполнения требований 10.4.1 – 10.4.6 и 10.4.8.
10.5.2 Спецификация верификации программного обеспечения (уточненная)
В результате выполнения требований 10.4.1, 10.4.2, 10.4.4, 10.4.5, 10.4.7 и 10.4.8.
10.5.3 Встроенное программное обеспечение
В результате выполнения требований 10.4.1.
10.5.4 Отчет о верификации программного обеспечения (уточненный)
В результате выполнения требований 10.4.2.

11 Верификация требований безопасности к программному обеспечению

11.1 Цель

Цель данной подстадии – продемонстрировать, что встроенное программное обеспечение отвечает требованиям безопасности к программному обеспечению.

11.2 Общие положения

Цель верификации требований безопасности к программному обеспечению – продемонстрировать, что встроенное программное обеспечение удовлетворяет этим требованиям в целевой среде.

11.3 Входная информация

11.3.1 Предварительные требования

Следующая информация должна быть доступна:

- спецификация проекта архитектуры программного обеспечения в соответствии с 7.5.1;
- план по обеспечению безопасности (уточненный) в соответствии с 7.5.2;
- спецификация требований к программному обеспечению системы безопасности (уточненная) в соответствии с 7.5.3;
- план верификации программного обеспечения (уточненный) в соответствии с 10.5.1;
- спецификация верификации программного обеспечения (уточненная) в соответствии с 10.5.2;
- отчет о верификации программного обеспечения (уточненный) в соответствии с 10.5.4;
- отчет о тестировании интеграции в соответствии с 8.5.3 ИСО 26262-4.

11.3.2 Дополнительная информация

Следующая информация может быть учтена:

- план подтверждения соответствия (уточненный) (см. 6.5.3 ИСО 26262-4);
- техническая концепция системы безопасности (см. 7.5.1 ИСО 26262-4);
- спецификация проекта системы (см. 7.5.2 ИСО 26262-4);
- руководства по применению инструментальных средств в соответствии с 5.5.4;
- руководства по применению методов (из внешнего источника).

11.4 Требования и рекомендации

11.4.1 Верификация требований безопасности к программному обеспечению должна быть спланирована, специфицирована и выполнена в соответствии с требованиями раздела 9 ИСО 26262-8.

11.4.2 Чтобы убедиться, что встроенное программное обеспечение отвечает требованиям безопасности к программному обеспечению, должны быть проведены испытания в тестовых средах, перечисленных в таблице 16.

Примечание – Могут быть повторно использованы контрольные примеры, которые уже использовались, например, для тестирования интеграции программного обеспечения.

Таблица 16 – Тестовые среды для выполнения верификации требований безопасности к программному обеспечению

Методы		УПБА			
		A	B	C	D
1a	Программно-аппаратное тестирование	+	+	++	++
1b	Сетевые среды электронного блока управления [a)]	++	++	++	++
1c	Транспортные средства	++	++	++	++

[a)] Например, тестовые стенды, частично или полностью включающие электрические системы транспортного средства, автомобили-лаборатории или базовые модели автомобилей для испытаний новых узлов.

11.4.3 Тестирование реализации требований безопасности к программному обеспечению должно быть выполнено на целевых аппаратных средствах.

11.4.4 Результаты верификации требований безопасности к программному обеспечению системы должны быть оценены на:

a) соответствие с ожидаемыми результатами;

b) охват требований безопасности к программному обеспечению;

c) выполнение или невыполнение критериев.

11.5 Результаты работы

11.5.1 План верификации программного обеспечения (уточненный)

В результате выполнения требований 11.4.1 – 11.4.3.

11.5.2 Спецификация верификации программного обеспечения (уточненная)

В результате выполнения требований 11.4.1 – 11.4.3.

11.5.3 Отчет о верификации программного обеспечения (уточненный)

В результате выполнения требований 11.4.1 и 11.4.4.

Приложение А
(справочное)

Обзор и последовательность выполняемых работ менеджмента разработки программного обеспечения изделия

Таблица А.1 содержит обзор целей, предварительных требований и результатов работы конкретных стадий разработки программного обеспечения изделия.

Таблица А.1 – Обзор разработки программного обеспечения изделия

Раздел	Цели	Предварительные требования	Результаты работы
5 Начальная подстадия разработки программного обеспечения изделия	Планирование и инициирование действий по обеспечению функциональной безопасности подстадии разработки программного обеспечения	План проекта (уточненный) (см. 5.5.1 ИСО 26262-4). План по обеспечению безопасности (уточненный) (см. 5.5.2 ИСО 26262-4). Техническая концепция системы безопасности в соответствии с 7.5.1 ИСО 26262-4. Спецификация проекта системы в соответствии с 7.5.2 ИСО 26262-4. План интеграции и тестирования устройства (уточненный) в соответствии с 8.5.1 ИСО 26262-4	5.5.1 План по обеспечению безопасности (уточненный). 5.5.2 Пдан верификации программного обеспечения. 5.5.3 Руководства по разработке и кодированию для языков моделирования и программирования. 5.5.4 Руководства по применению инструментальных средств
6 Спецификация требований безопасности к программному обеспечению	Формирование спецификации требований безопасности к программному обеспечению, которые выводятся из технической концепции системы безопасности и спецификации проекта системы. Формирование подробных требований к программно-аппаратному интерфейсу. Проверка согласованности требований безопасности к программному обеспечению и требований к программно-аппаратному интерфейсу с технической концепцией системы безопасности и спецификацией проекта системы	Техническая концепция системы безопасности в соответствии с 7.5.1 ИСО 26262-4. Спецификация проекта системы в соответствии с 7.5.2 ИСО 26262-4. Спецификация программно-аппаратного интерфейса (см. 7.5.6 ИСО 26262-4). План по обеспечению безопасности (уточненный) (см. 5.5.1). План верификации программного обеспечения (см. 5.5.2)	6.5.1 Спецификация требований безопасности к программному обеспечению. 6.5.2 Спецификация программно-аппаратного интерфейса (уточненная). 6.5.3 План верификации программного обеспечения (уточненный). 6.5.4 Отчет о верификации программного обеспечения
7 Проектирова-ние архитектуры программного обеспечения	Разработка проекта архитектуры программного обеспечения, реализующего требования безопасности к программному обеспечению. Верификация проекта архитектуры программного обеспечения	План по обеспечению безопасности (уточненный) (см. 5.5.1). Руководства по разработке и кодированию для языков моделирования и программирования. Спецификация программно-аппаратного интерфейса (см. 7.5.6 ИСО 26262-4). Спецификация требований безопасности к программному обеспечению (см. 6.5.1). План верификации программного обеспечения (уточненный) (см. 6.5.3). Отчет о верификации программного обеспечения (уточненный) (см. 6.5.4)	7.5.1 Спецификация проекта архитектуры программного обеспечения. 7.5.2 План по обеспечению безопасности (уточненный). 7.5.3 Спецификация требований безопасности к программному обеспечению (уточненная). 7.5.4 Отчет по анализу безопасности программного обеспечения. 7.5.5 Отчет по анализу зависимых отказов. 7.5.6 Отчет по верификации программного обеспечения (уточненный)

Продолжение таблицы А.1

Раздел	Цели	Предварительные требования	Результаты работы
8 Проектирование и реализация модулей программного обеспечения	Спецификация модулей программного обеспечения в соответствии с проектом архитектуры программного обеспечения и соответствующими требованиями безопасности к программному обеспечению. Реализация модулей программного обеспечения в соответствии со спецификацией. Статическая верификация проекта модулей программного обеспечения и их реализации	Руководства по разработке и кодированию для языков моделирования и программирования (См. 5.5.3). План верификации программного обеспечения (уточненный) (см. 6.5.3). Спецификация проекта архитектуры программного обеспечения (См. 7.5.1). План по обеспечению безопасности (уточненный) (см. 7.5.2). Спецификация требований безопасности к программному обеспечению системы (уточненная) (см. 7.5.3). Отчет о верификации программного обеспечения (уточненный) (См. 7.5.6)	8.5.1 Спецификация проекта модулей программного обеспечения. 8.5.2 Реализация модулей программного обеспечения 8.5.3 Отчет о верификации программного обеспечения (уточненный)
9 Тестирование модуля программного обеспечения	Демонстрация того, что программный модуль удовлетворяет специфи-кации проекта модуля программного обеспечения и не содержат нежелатель-ную функциональность	Спецификация программно-аппаратного интерфейса (уточненная) (см. 6.5.2). План верификации программного обеспечения (уточненный) (см. 6.5.3). План по обеспечению безопасности (уточненный) (см. 7.5.2). Спецификация проекта модулей программного обеспечения (см. 8.5.1). Реализация модулей программного обеспечения (см. 8.5.2). Отчет о верификации программного обеспечения (уточненный) (см. 8.5.3)	9.5.1 План верификации программного обеспечения (уточненный). 9.5.2 Спецификация верификации программного обеспечения. 9.5.3 Отчет о верификации программного обеспечения (уточненный)
10 Интеграция и тестирование программного обеспечения	Интеграция элементов программного обеспечения. Демонстрация того, что проект архитектуры программного обеспечения реализован встроенным программным обеспечением.	Спецификация программно-аппаратного интерфейса (уточненная) (см. 6.5.2). Спецификация проекта архитектуры программного обеспечения (см. 7.5.1). План по обеспечению безопасности (уточненный) (см. 7.5.2). Реализация модулей программного обеспечения (см. 8.5.2). План верификации программного обеспечения (уточненный) (см. 9.5.1). Спецификация верификации программного обеспечения(см. 9.5.2). Отчет о верификации программного обеспечения (уточненный) (см. 9.5.3)	10.5.1 План верификации программного обеспечения (уточненный). 10.5.2 Спецификация верификации программного обеспечения (уточненная). 10.5.3 Встроенное программное обеспечение. 10.5.4 Отчет о верифика-ции программного обес-печения (уточненный)

Окончание таблицы А.1

Раздел	Цели	Предварительные требования	Результаты работы
11 Верификация требований безопасности к программному обеспечению	Демонстрация того, что встроенное программное обеспечение отвечает требованиям безопасности к программному обеспечению	Спецификация проекта архитектуры программного обеспечения (см. 7.5.1). План по обеспечению безопасности (уточненный) (см. 7.5.2). Спецификация требований безопасности к программному обеспечению (уточненная) (см. 7.5.3). План верификации программного обеспечения (уточненный) (см. 10.5.1). Спецификация верификации программного обеспечения (уточненная) (см. 10.5.2). Отчет о верификации программного обеспечения (уточненный) (см. 10.5.4). Отчет о тестировании интеграции (см. 8.5.2 ИСО 26262-4)	11.5.1 План верификации программного обеспечения (уточненный). 11.5.2 Спецификация верификации программного обеспечения (уточненная). 11.5.3 Отчет о верификации программного обеспечения (уточненный)
Приложение С. Конфигурация программного обеспечения	Обеспечение управления изменениями программного обеспечения для различных применений	См. применяемые предварительные требования соответствующих стадий жизненного цикла системы безопасности, в которых применяется конфигурирование программного обеспечения	С.5.1 Спецификация данных о конфигурации. С.5.2 Спецификация данных о калибровке. С.5.3 План обеспечения безопасности (уточненный). С.5.4 Данные о конфигурации. С.5.5 Данные о калибровке. С.5.6 План верификации программного обеспечения (уточненный). С.5.7 Спецификация верификации. С.5.8 Отчет о верификации

Приложение В
(справочнное)

Разработка на основе модели

В.1 Цели

В настоящем приложении описывается основанная на модели концепция разработки программного обеспечения автомобиля и основные направления ее применения для разработки программного обеспечения изделий, применяемых в автотранспорте.

В.2 Общие положения

Математическое моделирование, которое широко используется во многих технических областях, также начинает активно использоваться при разработке встроенного программного обеспечения. В автомобильной промышленности моделирование используется для анализа реализуемой функциональности на концептуальном уровне (замкнутый / незамкнутый контур управления, контроль), а также для моделирования реального поведения физической системы (транспортного средства в окружающей среде).

Моделирование обычно осуществляется представленными на рынке коммерческими программными инструментальными средствами моделирования. Они поддерживают разработку и определение элементов системы или программного обеспечения, а также их связи и интерфейсы с помощью полуформальных графических моделей. Такие модели используют доступные для редактирования иерархические блок-схемы (например, схемы управления) и расширенные диаграммы изменения состояний (например, диаграммы состояний). Программные инструментальные средства включают необходимые средства описания, вычислительные методы и интерпретаторы / компиляторы. Графические редакторы позволяют интуитивно разрабатывать и описывать сложные модели. Для управления сложностью используется модульный принцип, основанный на иерархической структуризации. Модель состоит из функциональных блоков с четко определенными входами и выходами. Интерфейсы функциональных блоков соединены в блок-схеме с помощью ориентированных ребер, описывающих потоки сигналов. При этом в математической модели они описываются уравнениями, которые связывают интерфейсные переменные различных элементов. Линии соединения представляют собой причинно-следственные связи, задающие направления действия, связывают выходы одного блока со входами другого. Для организации выполнения и синхронизации могут быть также использованы специальные инструментальные средства моделирования с другими семантиками. Иерархия элементов может содержать несколько уровней.

Такие модели могут быть выполнены. Во время моделирования вычисление причинно-следственным связей следует заданным направлениям действий, пока вся модель не будет обработана. Существует целый ряд различных доступных систем для решения уравнений, описывающих модель. Для моделирования транспортных средств и окружающей среды в основном используются системы моделирования с переменным шагом. Для разработки встроенного программного обеспечения используются системы моделирования с фиксированным шагом, что является необходимым предварительным требованием для эффективной генерации кода.

Описанный стиль моделирования широко используется при разработке на основе модели встраиваемого в автомобиль программного обеспечения. Как правило, и исполнимая модель программного обеспечения системы управления (например, функциональная модель) и модель внешней системы (например, модель транспортного средства) и их окружения (например, модель внешней среды) создаются вначале цикла разработки и моделируются совместно. Таким образом, можно моделировать даже достаточно сложные систем автомобиля с высокой степенью детализации, с приемлемой скоростью расчета промоделировать их поведение достаточно близко к реальному поведению. В то время как в ходе разработки модель транспортного средства / внешней среды постепенно заменяется реальной системой и ее реальной средой, а функциональная модель может служить основой для реализации встроенного программного обеспечения блока управления, используя генерацию кода.

Одной из характерных черт парадигмы разработки на основе модели является тот факт, что функциональная модель не только определяет желаемые функции, но и предоставляет информацию для проектирования и, наконец, даже служит основой для реализации средствами генерации кода. Другими словами, такая функциональная модель включает в себя как аспекты спецификации, так и аспекты разработки и реализации. На практике эти различные аспекты нашли свое отражение в эволюции функциональной модели от модели спецификации на начальной стадии к модели проекта и затем к модели реализации и, наконец, ее автоматическое преобразование в код (эволюция модели). По сравнению с разработкой программного обеспечения на основе кода с четким разделением стадий, при разработке на основе модели можно отметить более тесное объединение стадий «Требования к программному обеспечению системы безопасности», «Проектирование архитектуры программного обеспечения» и «Проектирование и реализация модулей программного обеспечения». Более того, в процессе следующих друг за другом стадий разработки используется одно и то же графическое представление модели. Действия по верификации также могут рассматриваться по-другому, так как модели могут быть использованы в качестве полезного источника информации для процесса тестирования (например, тестирование на основе модели), или могут служить объектом верификации. «Бесшовное» использование моделей облегчает достаточно согласованную и эффективную разработку.

Примечание – Парадигма разработки на основе модели не зависит от типа упомянутых выше моделей. Могут быть использованы альтернативные модели, такие как UML.

Приложение С
(обязательное)

Конфигурация программного обеспечения

С.1 Цели

Цель конфигурирования программного обеспечения заключается в обеспечении управляемых изменений в поведении программного обеспечения для различных применений.

С.2 Общие положения

Конфигурируемое программное обеспечение позволяет разработку конкретного прикладного программного обеспечения с использованием данных о конфигурации и калибровки (см. рисунок С.1).

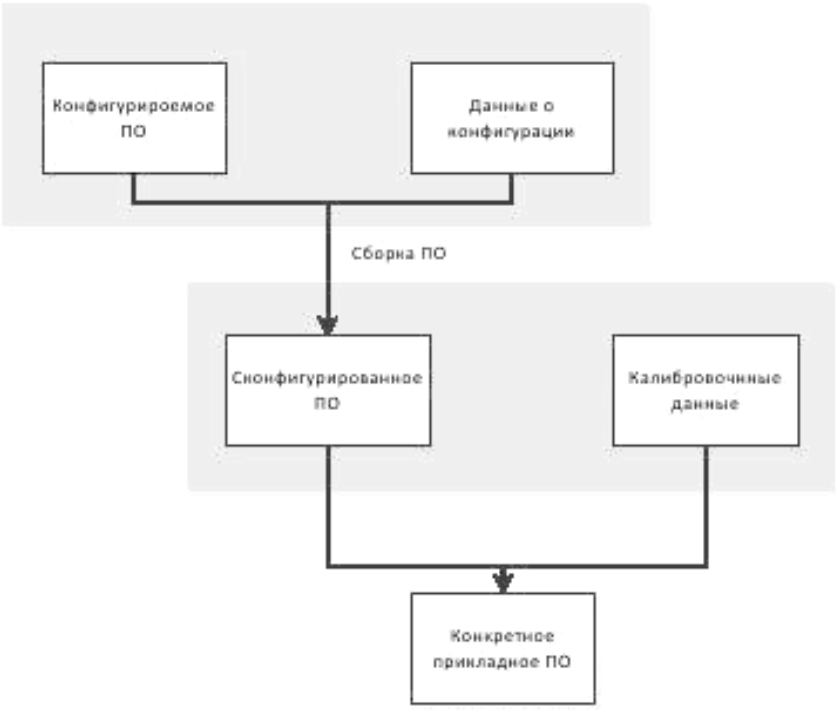


Рисунок С.1 – Создание конкретного прикладного программного обеспечения (ПО)

С.3 Входная информация

С.3.1 Предварительные требования

Предварительные требования определяются согласно соответствующим стадиям, в которых применяется конфигурирование программного обеспечения.

С.3.2 Дополнительная информация

Используется применимая дополнительная информация соответствующих стадий, в которых применяется конфигурирование программного обеспечения.

С.4 Требования и рекомендации

С.4.1 Должны быть специфицированы данные о конфигурации, чтобы обеспечить правильное использование конфигурируемого программного обеспечения в процессе жизненного цикла систем безопасности. Оно должно включать:

a) допустимые значения данных о конфигурации;
b) цель и использование данных о конфигурации;
c) диапазон, масштаб, единицы измерения;
d) взаимозависимость между различными элементами данных о конфигурации.

С.4.2 Должна быть выполнена верификация данных о конфигурации, чтобы обеспечить:

a) использование значений в заданном для них диапазоне;
b) совместимость с другими данными о конфигурации.

Примечание – Тестирование конфигурируемого программного обеспечения осуществляется на стадиях тестирования жизненного цикла программного обеспечения (см. разделы 9 «Тестирование модулей программного обеспечения», 10 «Интеграция и тестирование программного обеспечения», 11 «Верификация требований безопасности к программному обеспечению» и раздел 8 ИСО 26262-4 «Интеграция и тестирование устройства»).

С.4.3 Значение УПБА данных о конфигурации должно быть равно наибольшему значению УПБА конфигурируемого программного обеспечения, в котором эти данные используются.

С.4.4 Верификация конфигурируемого программного обеспечения должна планироваться, задаваться и выполняться в соответствии с требованиями раздела 9 ИСО 26262-8. Конфигурируемое программное обеспечение должно быть верифицировано на соответствие набору данных о конфигурации, которые должны быть применены для разрабатываемого устройства.

Примечание – На соответствие набору данных о конфигурации верифицируется только та часть встроенного программного обеспечения, поведение которой зависит от данных о конфигурации.

С.4.5 Для конфигурируемого программного обеспечения может быть применен упрощенный жизненный цикл программного обеспечения системы безопасности в соответствии с рисунками С.2 или С.3.

Примечание – Полную верификацию сконфигурированного программного обеспечения можно достигнуть с помощью комбинации следующих действий по верификации:

a) «верификация конфигурируемого программного обеспечения»,

b) «верификация данных о конфигурации»,

c) «верификация сконфигурированного программного обеспечения».

Это достигается либо верификацией диапазона допустимых данных о конфигурации при выполнении верификации, указанной в перечислении a), и демонстрацией соответствия этому диапазону выполнением верификации, указанной в перечислении b), либо демонстрацией соответствия диапазона допустимых данных о конфигурации выполнением верификации, указанной в перечислении в b) и выполнением верификации, указанной в перечислении c).

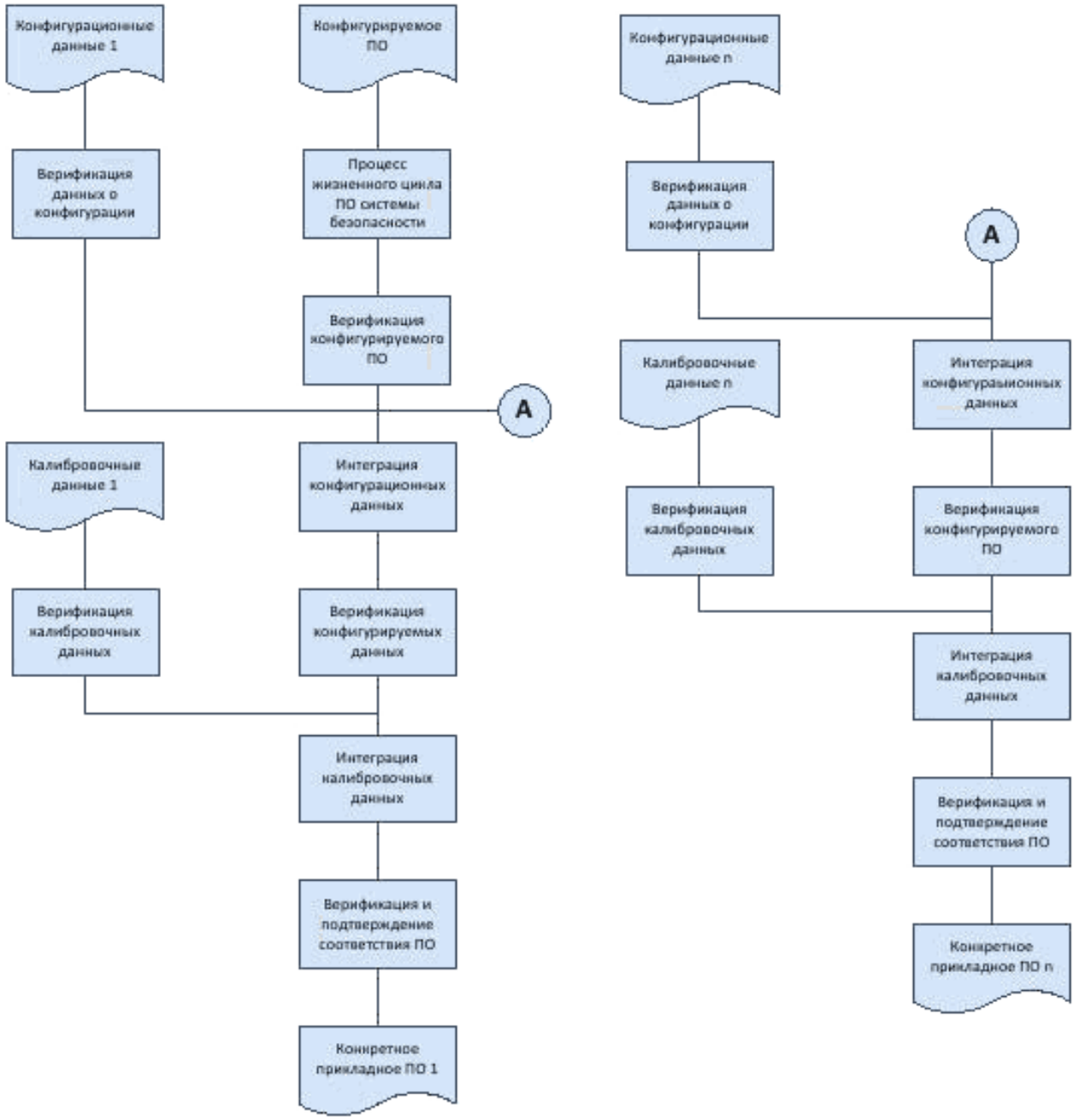


Рисунок С.2 – Варианты базовой модели стадии разработки программного обеспечения с конфигурируемым программным обеспечением (ПО) и различными данными о конфигурации

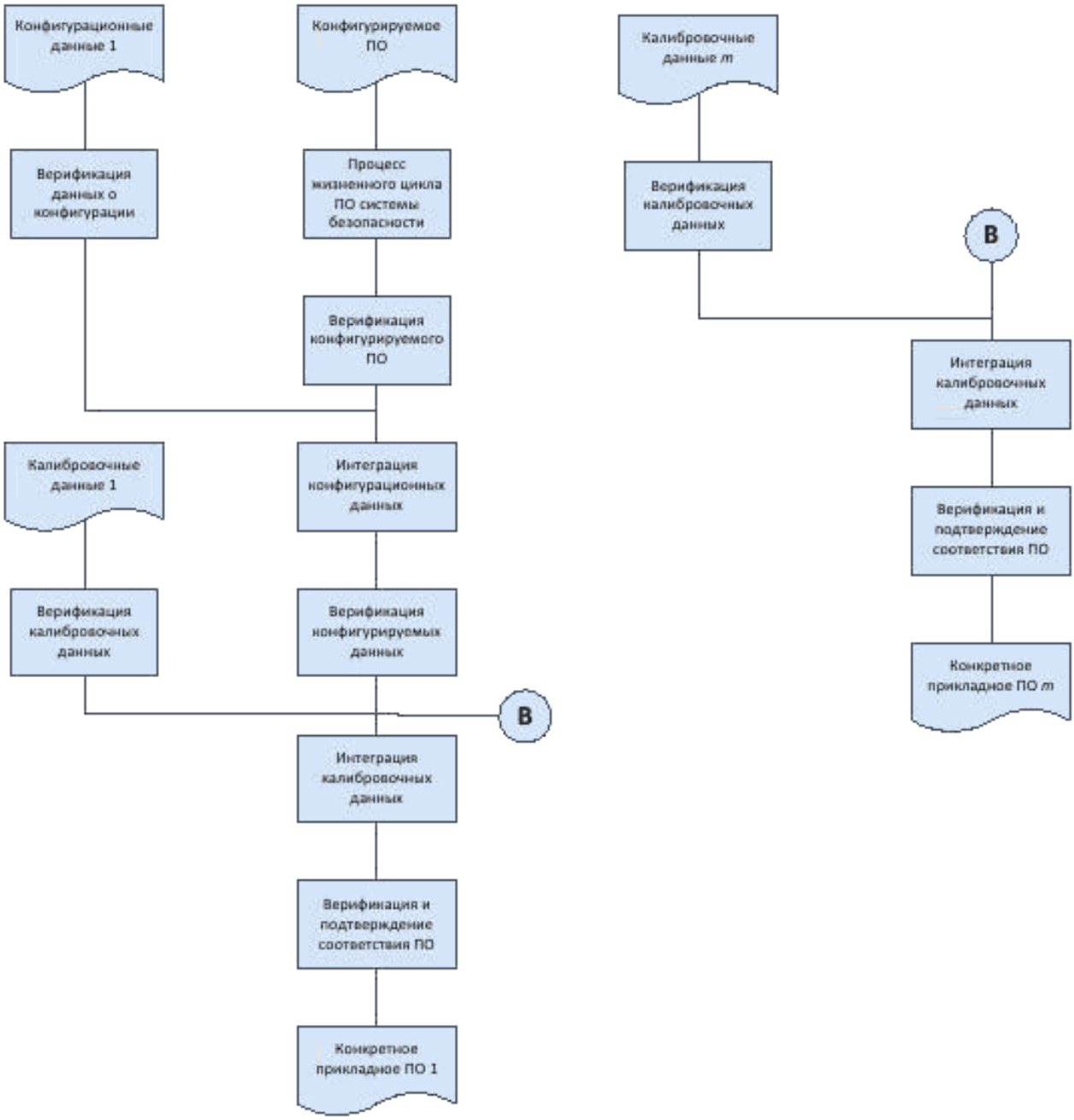


Рисунок С.3 – Варианты базовой модели стадии разработки программного обеспечения с конфигурируемым программным обеспечением (ПО) и различными данными о калибровке

С.4.6 Должны быть специфицированы данные о калибровке, связанные с компонентами программного обеспечения, чтобы обеспечить правильное функционирование и ожидаемые характеристики конфигурируемого программного обеспечения. Они должно включать:

a) допустимые значения данных о калибровке;

b) цель и использование данных о калибровке;

c) диапазон, масштаб и единицы измерения, если применимы, с их зависимостью от рабочего состояния;

d) известные взаимозависимости между различными данными о калибровке.

Примечание – Между данными о калибровке в рамках одного набора калибровочных данных или между данными о калибровке в разных наборах калибровочных данных могут существовать взаимозависимости такие же, как и для связанных с ними функций, реализованных в программном обеспечении отдельных ЭБУ;

d) известные взаимозависимости между данными о конфигурации и данными о калибровке.

Примечание – Данные о конфигурации могут оказать влияние на сконфигурированное программное обеспечение, которое использует данные о калибровке.

С.4.7 Верификация данных о калибровке должна планироваться, задаваться и выполняться в соответствии с требованиями раздела 9 ИСО 26262-8. Верификация данных о калибровке должна проверять, чтобы данные о калибровке были в пределах заданных для них границ.

Примечание – Верификация данных о калибровке также может быть выполнена в процессе верификации конкретного применения программного обеспечения или во время выполнения конфигурируемого программного обеспечения.

С.4.8 Значение УПБА данных о калибровке должно быть равно наибольшему значению УПБА требований безопасности к программному обеспечения, которое может быть нарушено этими данными.

С.4.9 Для обнаружения непреднамеренных изменений связанных с безопасностью данных о калибровке должны применяться механизмы для обнаружения непреднамеренных изменений данных, указанные в таблице С.1.

Таблица С.1 – Механизмы для обнаружения непреднамеренного изменения данных

Методы		УПБА			
		A	B	C	D
1a	Проверка достоверности данных о калибровке [a]	+	+	+	+
1b	Резервное хранение данных о калибровке	+	+	++	++
1c	Коды обнаружения ошибок [a]	o	o	+	++
[a] Коды обнаружения ошибок также могут быть реализованы в аппаратных средствах, в соответствии с ИСО 26262-5.					

С.4.10 При планировании получения и применения данных о калибровке должны быть специфицированы:

a) процедуры, которые должны соблюдаться;

b) инструментальные средства для создания калибровочных данных;

c) процедуры верификации данных о калибровке.

Примечание – Верификация данных о калибровке может включать в себя проверку диапазонов значений данных о калибровке или проверку взаимозависимости между различными данными о калибровке.

С.5 Результаты работы

С.5.1 Спецификация данных о конфигурации

В результате выполнения требований С.4.1 и С.4.3.

С.5.2 Спецификация данных о калибровке

В результате выполнения требований С.4.6.

С.5.3 План обеспечения безопасности (уточненный)

В результате выполнения требований С.4.1, С.4.4, С.4.5, С.4.9 и С.4.10.

С.5.4 Данные о конфигурации

В результате выполнения требований С.4.3.

С.5.5 Данных о калибровке

В результате выполнения требований С.4.8.

С.5.6 План верификации программного обеспечения (уточненный)

В результате выполнения требований С.4.2, С.4.4, С.4.7 и С.4.10.

С.5.7 Спецификация верификации

В результате выполнения требований С.4.4 и С.4.7.

С.5.8 Отчет о верификации

В результате выполнения требований С.4.1, С.4.4, С.4.7 и С.4.8.

Приложение D
(справочнное)

Отсутствие взаимного влияния между элементами программного обеспечения

D.1 Цели

Цель настоящего приложения состоит в том, чтобы привести примеры ошибок, которые могут вызвать взаимное влияние между элементами программного обеспечения (например, между элементами программного обеспечения различных разделов программного обеспечения). Кроме того, данное приложение рассматривает примеры возможных механизмов для предотвращения или обнаружения и смягчения последствий вышеупомянутых ошибок.

Примечание – Возможности и эффективность механизмов, используемых для предотвращения или обнаружения и смягчения соответствующих неисправностей, оцениваются в процессе разработки.

D.2 Общие положения

D.2.1 Достижение отсутствия взаимного влияния

Для обеспечения или оценки достижения отсутствия взаимного влияния между элементами программного обеспечения могут быть рассмотрены последствия типовых сбоев и распространение возможных возникающих в результате отказов.

D.2.2 Временная согласованность и выполнение

При временных ограничениях для элементов программного обеспечения, выполняемых в каждом разделе программного обеспечения, могут быть рассмотрены последствия следующих сбоев:

- блокирование выполнения;
- тупики;
- динамические взаимоблокировки;
- неправильные распределение времени выполнения;
- неправильная синхронизация между элементами программного обеспечения.

Пример – Могут быть рассмотрены такие механизмы, как: диспетчирование циклического выполнения; диспетчирование на основе фиксированного приоритета; диспетчирование, управляемое по времени; мониторинг времени работы процессора; мониторинг последовательности выполнения программы и мониторинг интенсивности входного потока.

D.2.3 Память

При доступе к памяти для элементов программного обеспечения, выполняемых в каждом разделе программного обеспечения, могут быть рассмотрены последствия следующих сбоев:

- повреждение содержания памяти;
- доступ к памяти по чтению или записи, выделенной для другого элемента программного обеспечения.

Пример – Могут быть использованы такие механизмы, как: защита памяти, биты четности, код с исправлением ошибок (ECC), контроль циклическим избыточным кодом (CRC), резервное хранение, ограниченный доступ к памяти, статический анализ доступа к памяти программного обеспечения и статическое распределение.

D.2.4 Обмен информацией

При обмене информацией для каждого отправителя или каждого получателя можно рассматривать перечисленные ниже причины сбоев или их последствия:

- повторение информации;
- потеря информации;
- задержка информации;
- ввод информации;
- нелегальное проникновение или неправильная адресации информации;
- неправильная последовательность информации;
- повреждение информации;
- асимметричная информация, посланная от отправителя к нескольким получателям;
- информация от отправителя, полученная только подмножеством получателей;
- блокирование доступа к каналу связи.

Примечание – Обмен информацией между элементами, выполняемыми в разных разделах программного обеспечения или в различных электронных блоках управления, включает сигналы, данные, сообщения и т.д.

Примеры

1 Для обмена информацией можно использовать устройства В/В, шины данных и т.д.

2 Могут быть использованы такие механизмы, как: протоколы связи, повторная передача информации, обратная передача данных, подтверждение приема информации, соответствующая конфигурация штырьков В/В, раздельные однонаправленные объекты коммуникации между двумя узлами, однозначно идентифицируемые двусторонние объекты коммуникации, асинхронная передача данных, синхронная передача данных, управляемые событиями шины данных, управляемые событиями шины данных с доступом, управляемым по времени, управляемые по времени шины данных, минисегментирование и управление доступом к шине по приоритету.

3 Коммуникационные протоколы могут содержать информацию такую, как идентификаторы для объектов коммуникации, подтверждающие активность сообщения, активные счетчики, порядковые номера, коды для обнаружения ошибок и коды с исправлением ошибок.

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов и документов национальным стандартам Российской Федерации

Таблица ДА

Обозначение ссылочного международного стандарта, документа	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО 26262-1:2011	–	*
ИСО 26262-2:2011	–	*
ИСО 26262-3:2011	–	*
ИСО 26262-4:2011	–	*
ИСО 26262-5:2011	–	*
ИСО 26262-7:2011	–	*
ИСО 26262-8:2011	–	*
ИСО 26262-9:2011	–	*
* Соответствующий национальный стандарт отсутствует. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта. Перевод данного международного стандарта находится в Федеральном информационном фонде технических регламентов и стандартов.		

Библиография

[1] ISO/IEC 12207, Systems and software engineering — Software life cycle processes

[2] IEC 61508 (all parts), Functional safety of electrical/electronic/programmable electronic safety-related systems

[3] MISRA-C:2004, Guidelines for the use of the C language in critical systems, ISBN 978-0-9524156-2-6, MIRA, October 2004

[4] MISRA AC AGC, Guidelines for the application of MISRA-C:2004 in the context of automatic code generation, ISBN 978-1-906400-02-6, MIRA, November 2007

УДК 62-783:614.8:331.454:006.354 ОКС 13.110

Ключевые слова: функциональная безопасность; жизненный цикл систем; транспортные средства; программируемые электронные компоненты и системы; программное обеспечение; разработка и верификация; архитектура программного обеспечения; модуль программного обеспечения.

Подписано в печать 20.01.2015. Формат 60x84 1/8.

Усл. печ. л. 4.65. Тираж 31 экз. Зак. 84

Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

ФГУП «СТАНДАРТИНФОРМ»
123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru