Стандартизация и сертификация

УДК 004.052.2

К. А. Бочков, д-р техн. наук,
С. Н. Харлап, канд. техн. наук,
Б. В. Сивко
Научно-исследовательская лаборатория
«Безопасность и ЭМС технических средств»,
Белорусский государственный университет транспорта

РАЗРАБОТКА ОТКАЗОУСТОЙЧИВЫХ СИСТЕМ НА ОСНОВЕ ДИВЕРСИТЕТНЫХ АКСИОМАТИЧЕСКИХ БАЗИСОВ

В статье рассматривается решение задачи, характерной для систем железнодорожной автоматики и телемеханики с применением диверситетных аксиоматических базисов – построение отказоустойчивой и безопасной микропроцессорной системы относительно заданных отказов. В качестве исследуемой системы выступает микропроцессорное устройство, выполняющее счет осей подвижного состава. На основании имитационных испытаний выполнено сравнение отказоустойчивости и безопасности таких систем с рассматриваемым диверситетом и без диверситета. Приведен пример последовательного усиления диверситета согласно аксиоматико-базисному подходу. Усиление диверситета возможно с помощью разделения памяти и регистров, адресов, множеств команд микропроцессора, а также защиты программного счетчика. Представлена формализация условий диверситета и основанная на нем защита от отказов по общей причине. На примере показано, что диверситет аксиоматических базисов и самотестирование общего базиса решают задачу обнаружения опасных отказов (диверситет решает задачу обнаружения опасных отказов диверситетных базисов, а самотестирование обнаруживает маскируемые отказы). В ходе эксперимента выявлено, что нарушение общего базиса ведет к отказам по общей причине и тем самым обоснован его обязательный контроль. Применение диверситетных аксиоматических базисов создает условия, при которых происходит усиление диверситета программного обеспечения. Выявлено, что во время проектирования и разработки диверситетного программного обеспечения возможно создание диверситетных высокоуровневых абстракций, что позволяет выбирать уровень абстракции диверситета. Рассмотрены особенности применения диверситетных аксиоматических базисов при разработке и верификации безопасных и отказоустойчивых систем.

отказоустойчивость; доказательство безопасности; диверситет; формальные методы; критические системы информационной инфраструктуры; обнаружение отказов

Введение

В настоящее время одной из актуальных проблем управления является влияние отказов по общей причине (CCF, common cause failure) на безопасность и отказоустойчивость многоканальных систем [1, 2]. Ее решение необходимо для разработки и верификации новых аппаратно-программных комплексов (АПК), относящихся к критическим системам информационной инфраструктуры (safety-critical systems). К данным системам относится множество устройств, влияющих на безопасность и отказоустойчивость и активно используемых на железнодорожном и морском транспорте, в гражданской авиации, телекоммуникациях, медицине, космосе, на опасном химическом производстве и др. [3, 4].

По причине влияния CCF происходит большое количество катастроф [3, 5, 6]. Согласно стандарту IEC 61508, учет фактора CCF настоятельно рекомендуется [1]. Сегодня основным способом защиты от CCF является программный и аппаратный диверситет, для реализации которого существует ряд методов, к которым относятся N-версионное программирование, выбор различных компиляторов и используемого программного обеспечения и др. [7, 8]. Для оценки степени диверситета может быть использован BETA-метод и модель BETAplus [1]. Однако имеющиеся решения CCF имеют ряд недочетов [4, 9–12]. Их характерной особенностью является экспертный характер диверситетных методов, а это ограничивает их эффективность и глубину решения, поэтому необходима их формализация.

В качестве решения проблемы CCF рассматривается аксиоматико-базисный подход (АБП) [13], позволяющий выполнять разработку на основе диверситетных аксиоматических базисов [14]. Данное решение дает возможность формализованно сравнивать системы по степени диверситета, последовательно усиливать диверситет во время разработки и наделять АПК целевыми формализованными свойствами. В статье исследуется практическое применение данного подхода на примере типовой задачи для систем железнодорожной автоматики и телемеханики (СЖАТ), относящейся к критическим системам информационной инфраструктуры. Рассматриваются разработка АПК без диверситета и с диверситетом согласно подходу, испытания на отказоустойчивость обеих решений и сравнение результатов, а также особенности применения подхода и построения диверситетных систем.

1 Описание типовой задачи

Рассмотрим систему, выполняющую счет осей подвижного состава [15–17]. Данная система представляет собой микроэлектронное устройство, которое получает информацию с двух датчиков о присутствии колесной пары на железнодорожном полотне (рис. 1). Необходимо подсчитать количество колесных пар,

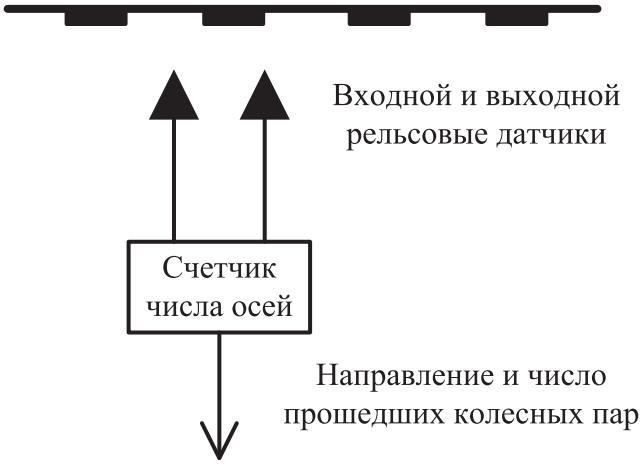

Рис. 1. Счетчик числа осей

которые прошли в прямом или обратном направлении, при этом результат подсчета передается на внешнее устройство. Другими словами, система представляет собой счетный пункт [15], а ее применение с постовым устройством [17] или счетным устройством [15] позволяет решать такие задачи СЖАТ, как контроль свободности участков пути и определение ординаты подвижного состава.

Датчики могут передавать на микропроцессорное устройство информацию о возникновении одного или двух событий, о наличии или отсутствии колесной пары в зоне расположения датчика. Соответственно при прохождении колесной пары слева направо значения датчиков изменяются в последовательности $00\rightarrow10\rightarrow11\rightarrow01\rightarrow00$ и счетчик инкрементируется. При движении в обратном направлении – $00\rightarrow01\rightarrow11\rightarrow10\rightarrow00$ и счетчик декрементируется. Результат подсчета (число) передается на индикатор или другое внешнее устройство.

В качестве устройства, осуществляющего данную функцию, выбран микропроцессор PIC16F877A [18]. Входные значения датчиков передаются напрямую на входные линии, а выходные значения для индикации отправляются в виде числа в двоичном коде на внешние порты.

В программное обеспечение (ПО) рассматриваемых систем добавляется функция самопроверки, которая заключается в контроле корректности переходов между состояниями. Некорректными считаются попытки перехода между 00 и 11 или между 01 и 10; в случае обнаружения такой ситуации система должна подать сигнал о том, что корректность ее работы нарушена. В то же время считается, что быстродействие датчиков намного выше скорости перемещения колесной пары и при нормальной работе такие переходы не должны возникать.

2 Описание эксперимента

Для решения задачи построения безопасной системы счета осей разработаны два варианта ПО, первый из которых выполнен без диверситета, а второй

представляет собой диверситетное двухканальное ПО, работающее на одном микроконтроллере. Второй вариант был получен из первого последовательным выполнением шагов согласно АБП на основе диверситетных аксиоматических базисов по алгоритму [14]. В ходе преобразования система приобрела отказоустойчивость относительно рассматриваемых отказов, повышен диверситет согласно АБП. Эксперимент заключается в проведении имитационных испытаний первого и второго вариантов ПО и сравнении результатов, что позволяет определить эффективность и особенности применения подхода на основе диверситетных аксиоматических базисов.

При использовании АБП на основе диверситетных аксиоматических базисов ставится задача усиления диверситета базисов [14], которые должны иметь наименьший общий базис (так происходит усиление диверситета согласно АБП). Далее, согласно АБП, отказы, не затрагивающие общий базис, будут проявляться в диверситетных каналах по-разному. С одной стороны, для таких отказов решается проблема CCF (отсутствие среди них опасных, так как в случае их проявления выходные сигналы будут различными). С другой стороны, только для общего базиса необходима проверка сторонними средствами, при этом данная проверка формализована в рамках АБП (предоставляется определенный набор утверждений, который требуется проверить, например, с помощью самотестирования или самопроверки).

Таким образом, отказы вне общего базиса при работе второго варианта ПО не должны быть опасными (они могут быть защитными или маскируемыми), это проверяется в ходе имитационных испытаний. Кроме того, процедуры самопроверки и самотестирования общего базиса должны свести количество опасных отказов к нулю, что также проверяется на имитационных испытаниях.

Множество учитываемых при разработке и испытаниях имитационных отказов выбирается на основании требований стандарта ГОСТ Р МЭК 61508 [19]. Соответственно все элементы памяти должны быть защищены от одиночных отказов константного нуля и единицы (stack-at-faults, SA), а также от коротких замыканий между линиями связи (bridging-faults, B). С учетом этого осуществляется проектирование (защита от указанных отказов согласно АБП) и последующее имитационное моделирование.

Входы и выходы двух вариантов исполнения показаны на рис. 2.

Для обеспечения строгости диверситета второго варианта ПО были проведены дополнительные мероприятия. Во-первых, разделены входные (IN1/IN2) и выходные (COUNTER1/COUNTER2, ERROR1/ERROR2) цепи диверситетных программ. Во-вторых, ПО в конечном итоге было построено таким образом, что каждый из каналов в любом случае и при любых рассматриваемых отказах передавал управление другому каналу, что было выполнено посредством гарантии завершения алгоритмом технологического цикла за конечное время. В-третьих, добавлена проверка программного счетчика на отказоустойчивость, внешне выражающаяся в сигнале SAFE_PCL, который находится в переменном

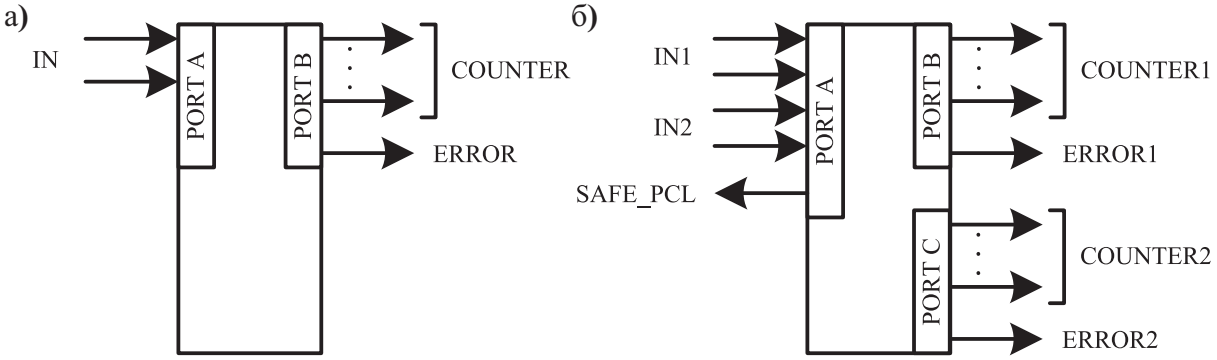

Рис. 2. Микроконтроллеры первого варианта исполнения (*а*) и диверситетного варианта (*б*):
IN – входные сигналы с датчиков; COUNTER – результат подсчета осей; ERROR – индикация обнаружения ошибки; SAFE_PCL – индикация обнаружения отказа программного счетчика

или постоянном состоянии, первое из которых говорит об отсутствии отказов программного счетчика. Считается, что в случае появления второго сигнала вся система будет переведена в безопасное состояние внешними средствами, например типовым импульсным декодером [20]. Данная проверка представляет собой периодическое самотестирование общего базиса [21].

Таким образом, в результате описанных дополнений можно диагностировать эффективность и строгость выполненного диверситета.

3 Усиление диверситета

С целью усиления диверситета рассматриваются две версии ПО (изначально это могут быть две идентичные программы). Для этих версий устанавливаются те аксиоматические базисы (например, A и B), на которых основывается функциональность ПО. Как правило, при рассмотрении новых неучтенных базисов обе версии зависят как от A, так и от B. В дальнейшем ПО одной из версий изменяется таким образом, что в ней исключается зависимость от одного из базисов (например, версия 1 перестает быть зависимой от базиса A). В свою очередь, для версии 2 исключается зависимость от базиса B. Описанные действия показаны на рис. 3. Очевидно, что конечная пара версий ПО обладает бóльшим диверситетом относительно базисов A и B, чем изначальные версии [14]. Конечным итогом эксперимента является то, что версия 1 реализует свою функциональность на основе базиса B, а версия 2 – на основе базиса A.

Во время решения типовой задачи согласно данному подходу диверситет усилен поэтапно, последовательно были осуществлены следующие шаги диверсификации:

– разделение по ячейкам оперативной памяти;
– разделение по адресам;

– разделение на множества команд;

– разделение по регистрам.

Описанный подход позволил на каждом из шагов сфокусироваться на соответствующих аспектах, изменить соответствующим образом ПО и добиться получения необходимых свойств.

Разделение по ячейкам оперативной памяти основано на использовании общих ячеек для двух программ. Выбор адресов производился с помощью кода Хэмминга с расстоянием 3.

Разделение множества команд и регистров было произведено так, как показано на рис. 4. В базис A попали команды, работающие с аккумулятором W и флагом Z (movlw, andlw, decfsz, …). В базис B попали команды, непосредственно работающие с памятью (bcf, clrf, btfss, …).

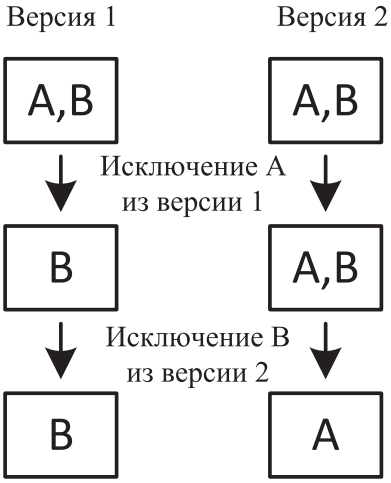

Рис. 3. Исключение базисов и усиление диверситета

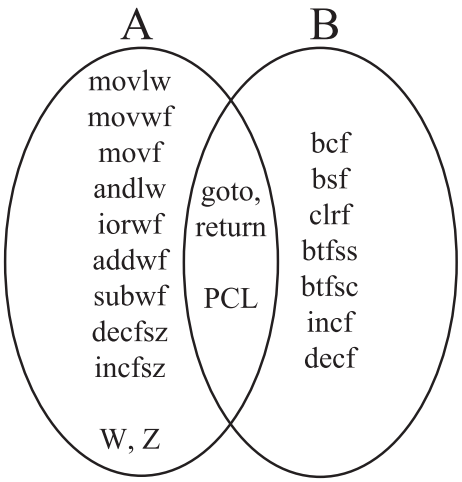

Рис. 4. Разделение команд и регистров

Разделение по регистрам было произведено таким образом, что первый канал использовал только регистр W и флаг Z, а для второго было достаточно оперативной памяти. Такие команды, как goto и return, а также регистр PCL остались в общем базисе. Их отказоустойчивость проверялась и диагностировалась сигналом SAFE_PCL [21].

Таким образом, диверситетными базисами являлись утверждения об отсутствии одиночных отказов в рассматриваемых ячейках памяти и регистрах при дешифровании команд и в адресной шине данных.

4 Испытания

Оценка эффективности метода должна проверяться на практике и в настоящее время для этого применяется имитационное моделирование. Такой способ

позволяет провести в лабораторных условиях полный цикл разработки и верификации с минимальными затратами.

В качестве инструмента моделирования выбран программный комплекс КИИБ [22], который предназначен для имитационных испытаний на функциональную безопасность в соответствии с IEC 61508, EN 50126, ОСТ 32.146 микропроцессорных систем управления ответственными технологическими процессами. С помощью КИИБ возможно внесение различного вида отказов технических средств, а также анализ последующего поведения рассматриваемой системы.

Анализ имитационного моделирования позволяет понять, в какое состояние переходит система в результате внесения отказа, и соответственно диагностировать тип отказа, который может быть опасным, защитным или маскируемым (табл. 1).

Таблица 1. Диагностика отказов

Корректность подсчета осей	Сигнал на переход в безопасное состояние	
	+	–
+	Защитный отказ	Маскируемый отказ
–	Защитный отказ	Опасный отказ

Если сигнал на переход в безопасное состояние появлялся во все время испытаний, он считался активным. Функцию подсчета осей система должна была выполнять корректно в каждый момент времени с учетом быстродействия микроконтроллера.

В программу испытаний были включены одиночные отказы SA и B [19]. Данные отказы вносились для всех используемых регистров и ячеек оперативной памяти. Для первого варианта ПО проводились испытания при внесении следующих отказов:

– SA0(W, PCL), SA1(W, PCL) – константные значения аккумулятора и программного счетчика – 32 отказа;

– B(W, PCL) – короткие замыкания между линиями связи аккумулятора и программного счетчика – 28 отказов;

– SA0(Z) и SA1(Z) – 2 отказа;

– B(Z) – короткие замыкания между линиями связи для регистра STATUS, затрагивающего флаг Z,– 4 отказа;

– SA0 и SA1 трех используемых ячеек памяти – 48 отказов;

– B-отказы трех используемых ячеек памяти – 42 отказа.

Для второго варианта ПО проводились испытания при внесении следующих отказов:

– A0(W, PCL) и SA1(W, PCL) – 32 отказа;

– B(W, PCL) – 28 отказов;

– SA0(Z) и SA1(Z) – 2 отказа;

– B(Z) – 4 отказа;
– SA0 и SA1 пяти используемых ячеек памяти – 80 отказов;
– B-отказы пяти используемых ячеек памяти – 70 отказов.

Поскольку каждый цикл ПО имел конечное время выполнения, была рассчитана максимальная длительность одного тела цикла [23] и для испытаний скорость прохождения сигналов была установлена такой, что цикл обязательно должен был завершиться в указанное время. Время испытания было задано так, что его было достаточно для прохождения такого количества осей, при котором происходило переполнение внутреннего счетчика.

Каждый из отказов проверялся четыре раза: при подсчете осей в обоих направлениях, в начале работы программы (влияющие на инициализацию), и во время работы микроконтроллера (в данном случае после 1000 тактов его работы).

Таким образом, для первого и второго варианта ПО было проведено 624 и 864 испытания соответственно.

КИИБ позволяет во время испытаний протоколировать все входные и выходные данные. Во время работы микроконтроллера изменяются его состояние и поведение, судя по внешним сигналам, и, как следствие, он должен выдавать соответствующие выходные данные. На их основании можно диагностировать корректность его работы и факт обнаружения отказа теми или иными средствами.

Например, в случае отказа SA1 младшего бита программного счетчика микроконтроллер перестает вести подсчет и выполнять самопроверку и выходные сигналы становятся постоянными. В ходе их анализа по завершении испытания определяется факт нарушения функциональности (подсчет прекратился), а также обнаруживается или нет отказ средствами самопроверки (сигнал ERROR), определяется работоспособность программного счетчика (постоянное значение SAFE_PCL в случае отказа и переменное, если PCL в рабочем состоянии).

5 Результаты испытаний

Результаты испытаний первого варианта ПО показаны в табл. 2.

Таблица 2. Испытания первого варианта исполнения

№	E	F	Отказ	Количество	%
1	0	0	Опасный	402	64,4
2	0	1	Маскируемый	150	24,0
3	1	0	Защитный	72	11,5
4	1	1			

Примечания. E – наличие сигнала перехода в безопасное состояние (результат самопроверки, ERROR); F – корректность функционирования во время испытания.

Результаты испытания второго варианта ПО с учетом диверситета, но без учета самотестирования показаны в табл. 3.

Таблица 3. Испытания второго варианта выполнения ПО без самотестирования

№	E1	E2	F1	F2	Отказ	Количество	%
1	0	0	0	0	Опасный	54	6,25
2	0	0	0	1	Защитный	211	24,4
3	0	0	1	0		144	16,7
4	0	0	1	1	Маскируемый	244	28,2
5	0	1	0	0	Защитный	22	2,55
6	0	1	1	0		32	3,7
7	1	0	0	0		22	2,55
8	1	0	0	1		113	13,1
9	1	1	0	0		22	2,55

Примечания. E1, E2 – наличие сигнала самопроверки (ERROR1 и ERROR2) для первого и второго канала соответственно; F1, F2 – корректность функционирования во время испытания для каждого из каналов. Для непоказанных комбинаций E1, E2, F1, F2 случаи отказа не зафиксированы.

Самопроверка обнаружила 24,4 % отказа на 211 испытаниях. В табл. 3 это № 5–9 – те отказы, во время которых один из сигналов, E1 или E2, имел состояние 1 (соответствующим каналом отмечена некорректность работы).

Сравнение результатов выполнения двух диверситетных программ позволило обнаружить 57,9 % отказов на 500 испытаниях. В табл. 3 это № 2, 3, 6, 8 – те отказы, для которых значения F1 и F2 различны. Опасные отказы имели место только для программного счетчика, который входит в общий базис, т. е. АБП решил задачу обнаружения одиночных отказов диверситетных базисов.

Значительный процент маскируемых отказов объясняется тем, что при испытаниях ячейки памяти проверялись полностью, несмотря на то что в них могли использоваться только некоторые биты. В таких случаях в большинстве своем отказы не влияют на работу системы. Однако, в некоторых случаях, когда команда микроконтроллера для выполнения операций задействует все слово, но функционально имеют смысл только несколько битов, отказы могут проявляться (например, операция вида AND 0x01 для выделения младшего бита выполняется со всей ячейкой памяти, и отказ SA1 старшего бита проявится в случае последующего прямого сравнения с единицей).

Обнаружение маскируемых отказов при АБП выполняется посредством взаимного тестирования аксиоматических базисов [13, 14, 21], его применение в данной работе не рассматривается.

Результаты испытания второго варианта выполнения ПО с учетом диверситета и самотестирования показаны в табл. 4.

Таблица 4. Испытания второго варианта выполнения ПО с самотестированием

№	E1	E2	F1	F2	SAFE_PCL	Отказ	Количество	%
1	0	0	0	0	~	Опасный	0	0
2	0	0	0	1	~	Защитный	211	24,4
3	0	0	1	0	~		144	16,7
4	0	0	1	1	~	Маскируемый	244	28,2
5	0	1	1	0	~	Защитный	32	3,7
6	1	0	0	1	~		113	13,1
7	0	0	0	0	–		54	6,25
8	0	1	0	0	–		22	2,55
9	1	0	0	0	–		22	2,55
10	1	1	0	0	–		22	2,55

Примечание. SAFE_PCL – состояние сигнала самотестирования программного счетчика: переменное (~), означающее что отказов не зафиксировано, и постоянное (–) – что зафиксирован отказ PCL.

Применение диверситета согласно АБП и самопроверка общего базиса позволили свести количество опасных отказов к нулю. Это было достигнуто в два этапа. Изначально диверсификация базисов позволила разделить факторы, ведущие к CCF, и тем самым дать возможность обнаружить 57,9 % отказов (результаты см. в табл. 3). В то же время разделение базисов позволило формализованно и целенаправленно, с помощью самотестирования подготовить средства (второй этап), которые успешно обнаруживают все оставшиеся 6,25 % опасных отказов, которые могли проявиться только в общем базисе (результаты см. в табл. 4).

6 Анализ результатов

Изначально теоретически было доказано, что отказы в независимых базисах будут проявляться независимо и это должно улучшить показатели безопасности и отказоустойчивости. Теперь данный тезис экспериментально подтвержден: с помощью диверситета были упразднены все опасные отказы, не влияющие на общий базис. Они либо обнаруживаются (один канал работает корректно, а второй с нарушениями), либо маскируются (во время функционирования системы никак не проявляются).

Вторым результатом является подтверждение утверждения о том, что диверситет позволяет обнаружить все одиночные отказы независимых базисов, но время их обнаружения не регламентируется. Пóзднее обнаружение может быть в случае, если имеется некий ресурс, который задействуется редко, из-за этого отказ будет проявляться также редко. В связи с этим для полноты решения

задач безопасности и отказоустойчивости необходима периодическая проверка базисов с помощью самотестирования.

Метод диверситетных базисов позволяет формализованно выделять общий базис, который подлежит отдельному рассмотрению. Это дает возможность выделить факторы, ведущие к CCF, и обратить на них особое внимание. Вместе с тем метод диверситетных базисов позволяет формализованно отнести факторы, ведущие к CCF, к разным базисам и тем самым целенаправленно решать проблему CCF.

Экспериментально показано, что общий базис (например, работоспособность общих ресурсов), нарушение которого ведет к CCF, необходимо обязательно контролировать, так как отказы, влияющие на его истинность, с большой вероятностью могут перевести систему в опасное состояние и диверситетное ПО не сможет решить данную проблему. Следует также отметить, что важной задачей является уменьшение общего базиса, что ведет к более эффективному обнаружению отказов средствами диверситета и облегчает задачу проверки общего базиса.

На примере показано, что диверситет и самотестирование полностью решили задачу обнаружения опасных отказов и тем самым отпадает необходимость самопроверки. При отказе возможен произвольный переход из допустимого состояния в другое допустимое (исходя из условий самопроверки), и такой переход не обнаруживается.

Усиление диверситета в данном эксперименте производилось без привлечения независимых разработчиков. Результаты испытаний показывают, что при усилении диверситета согласно АБП характеристики системы улучшаются.

Разделение регистров и независимое множество команд (во время выполнения шагов диверсификации) форсировало применение различных алгоритмов для решения одних и тех же задач. Например, обработка переходов между состояниями для трех версий была реализована по-разному, что показано на рис. 5 и 6.

Следует отметить, что диверситет может выполняться на разных уровнях: функциональных спецификаций, технологических алгоритмов, аппаратных средств и др. В нашем случае АБП предназначен для защиты от определенного множества отказов, что позволяет решать проблему CCF аппаратных средств на необходимом и достаточном уровне. Вместе с тем было замечено, что выполненные в ходе эксперимента действия способствовали повышению диверситета технологических алгоритмов и программы. Но для обоснования достаточности предлагаемого подхода требуется дополнительная экспертиза.

В ходе проектирования и разработки диверситетного ПО появилась возможность создания более высокоуровневых операций на каждом из базисов. Например, для операций копирования, передачи данных в порт, инкремента, декремента и других можно рассмотреть соответствующие им реализации на базисах (примеры показаны в табл. 5).

1	2	3

; Изменилось состояние?
movf STATE, 0
xorwf MEM_IN, 0
btfss STATUS, Z
goto end_body
; Корректно изменение?
movfw STATE
xorlw 3
xorwf MEM_IN, 0
btfsc STATUS, Z
goto to_safe
 	
; Исходное 00?
clrw
xorwf STATE, 0
btfsc STATUS, Z
goto s00
…
	; Вычисление индекса
	; для таблицы переходов
	movf IN_1,0
	andlw 0x03
	iorwf STATE_1,0
	; Переход в графе
	addwf PCL,1
	goto s0000
	goto s0001
	goto s0010
	goto s0011
	…
	; 00->00
	s0000:
	goto mout_1
	; 00->01
	s0001:
	movlw 0x04
	movwf STATE_1
	goto mminus_1
	…
		; Входные биты
		bsf MEM_IN_2, 0
		btfss IN1_2
		bcf MEM_IN_2, 0
		bsf MEM_IN_2, 1
		btfss IN2_2
		bcf MEM_IN_2, 1
		; Бинарный поиск
		; действия
		m: btfsc f0
		goto m1
		m0: btfsc f1
		goto m01
		m00: btfsc t0
		goto m001
		m000: btfsc t1
		goto mplus_2
		…

Рис. 5. Программа обработки состояний

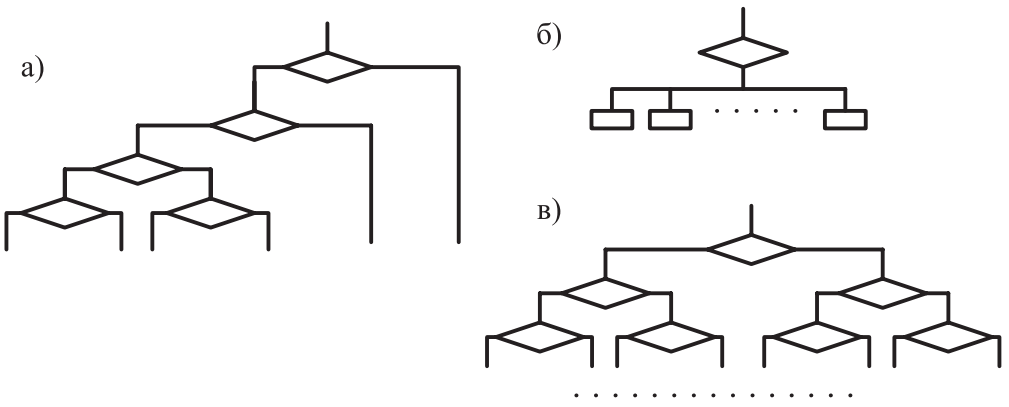

Рис. 6. Алгоритмы первого (*а*) и диверситетного (*б*, *в*) вариантов программы

Если на более высоком уровне реализуются одинаковые абстракции, то диверситетность разработки на разных базисах ограничивается рассматриваемым уровнем абстракции (рис. 7). Другими словами, если будут реализованы все необходимые высокоуровневые операции, то на их уровне абстракции проектирование ПО уже не будет зависеть от диверситетности базисов. Как результат, имеется возможность выбирать уровень абстракции диверситетных базисов.

Таблица 5. Высокоуровневые операции на разных базисах

Операция	Базис A	Базис B
Инкремент ячейки памяти	`movlw 1` `addwf MEM, 1`	`incf MEM`

Передача в порт	`movfw MEM`
	`movwf PORT`
		`; 8 раз, для каждого бита`
		`btfsc MEM, 1`
		`bsf PORT, 1`
		`btfss MEM, 1`
		`bcf PORT, 1`
		…

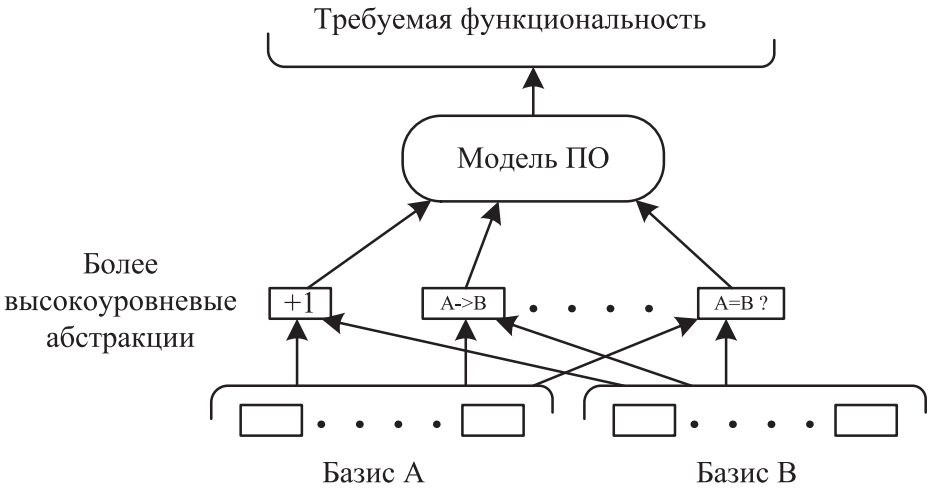

Рис. 7. Реализация высокоуровневых абстракций на различных базисах

Заключение

В статье, при использовании экспериментальных данных имитационных испытаний относительно применения АБП на основе диверситетных аксиоматических базисов, показано, что:

– возможно формализованное построение и доказательство безопасности безопасных и отказоустойчивых систем;

– используемый подход позволяет целенаправленно усиливать и обосновывать диверситет без привлечения сторонних разработчиков;

– применение данного подхода понуждает разработчиков к усилению диверситета;

– используемый подход позволяет формализованно определять факторы, ведущие к CCF, и в дальнейшем организовать целенаправленную защиту от их влияния;

– общий базис необходимо контролировать (например, с помощью самотестирования), так как влияющие на него отказы могут привести систему в опасное состояние, а диверситетное ПО не может решить эту проблему;

– для построения безопасной системы необходимо применение как диверситетных методов, так и самотестирования;

– диверсификация базисов может быть осуществлена на новом уровне абстракции, что позволяет создавать диверситетное относительно базисов ПО в рамках одной высокоуровневой модели.

Результаты имитационного моделирования подтверждают, что АБП на основе диверситетных аксиоматических базисов позволяет разрабатывать отказоустойчивые и безопасные системы с необходимыми свойствами. Показано, что возможно целенаправленное усиление диверситета базисов и уменьшение общего базиса, – это улучшает его способность обнаруживать опасные отказы и способствует повышению уровня безопасности и отказоустойчивости.

Таким образом, экспериментально подтверждена эффективность применения АБП на основе диверситетных аксиоматических базисов и положено начало его практическому применению.

Библиографический список

1. Smith D. J. Safety Critical Systems Handbook. A Straightforward Guide to Functional Safety, IEC 61508 and Related Standards, Including Process IEC 61511 and Machinery IEC 62061 and ISO 13849 / D. J. Smith, Kenneth G. L. Simpson // Oxford, UK, Elsevier Ltd, 2010. – 270 p.
2. Parry G. W. Common Cause Failure Analysis: A Critique and Some Suggestions / G. W. Parry. – Gaithersburg, Maryland, USA, Reliability Engineering and System Safety, 1991. – Vol. 34. – Issue 3. – Pp. 309–326.
3. Neumann P. G. Computer-Related Risks / P. G. Neumann. – N.Y., USA, Addison-Wesley Professional, 1995. – 384 p.
4. Leveson N. Safeware: System Safety and Computers / N. Leveson. – N.Y., USA, Addison-Wesley, 1995. – 680 p.
5. Weil V. Professional Responsibility for Harmful Actions / V. Weil, B. Ferry. – Kendall Hunt, Dubuque, Iowa, 1984. – Pp. 402–411.
6. Sagan S. D. The Limits of Safety: Organizations, Accidents, and Nuclear Weapons / S. D. Sagan. – Princeton University Press, Princeton, N. J., 1993. – 302 p.
7. Бочков К. А. Микропроцессорные системы автоматики на железнодорожном транспорте : учеб. пособие / К. А. Бочков, А. Н. Коврига, С. Н. Харлап. – Гомель : БелГУТ, 2013. – 254 с.
8. Chen L. N-Version Programming: A Fault-Tolerance Approach to Reliability of Software Operation / L. Chen, A. Avizienis // FTCS-8: Proceedings of the Eighth Annual International Conference on Fault Tolerant Computing. – Toulouse, France, 1978. – Pp. 3–9.
9. Knight J. C. An Experimental Evaluation of the Assumption of Independence in Multiversion Programming / J. C. Knight, N. G. Leveson // IEEE Transactions on Software Engineering. – USA, N.J., 1986. – Vol. 12. – Issue 1. – Pp. 96–109.

10. Brilliant S. Analysis of Faults in an N-Version Software Experiment // IEEE Transactions on Software Engineering / S. Brilliant, J. C. Knight, N. G. Leveson. – Virginia Univ., Charlottesville, VA, USA, 1990. – Vol. 16. – Issue 2. – Pp. 238–247.
11. Brilliant S. The Consistent Comparison Problem in N-Version Programming / S. Brilliant, J. C. Knight, N. G. Leveson// IEEE Transactions on Software Engineering. – Virginia Commonwealth Univ., Richmond, VA, USA, 1989. – Vol. 15. – Issue 11. – Pp. 1481–1485.
12. Шубинский И. Б. Функциональная надежность информационных систем. Методы анализа / И. Б. Шубинский. – Ульяновск : Изд-во журнала «Надежность», 2012. – 216 с.
13. Сивко Б. В. Аксиоматико-базисный подход для разработки безопасных и отказоустойчивых систем / Б. В. Сивко // Автоматика на транспорте. – 2015. – Т. 1. – № 4. – С. 381–399.
14. Сивко Б. В. Диверситетные аксиоматические базисы для разработки безопасных и отказоустойчивых систем / Б. В. Сивко // Вестник БелГУТа : Наука и транспорт. – 2014. – № 1 (28). – С. 19–23.
15. Кириленко А. Г. Счетчики осей в системах железнодорожной автоматики и телемеханики : учеб. пособие / А. Г. Кириленко, А. В. Груша. – Хабаровск : Изд-во ДВГУПС, 2003. – 75 с.
16. Сапожников Вал. В. Методы построения безопасных микроэлектронных систем железнодорожной автоматики / Вал. В. Сапожников, Вл. В. Сапожников, Х. А. Христов, Д. В. Гавзов. – М. : Транспорт, 1995. – 272 с.
17. Тильк И. Г. Новые устройства автоматики и телемеханики железнодорожного транспорта / И. Г. Тильк. – Екатеринбург : УрГУПС, 2010. – 168 с.
18. Martin B. PIC Microcontrollers An Introduction to Microelectronics / B. Martin. – Meppel, The Netherlands, Elsevier, 2nd edition, 2004. – 372 p.
19. ГОСТ Р МЭК 61508-2–2012. Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Ч. 2 : ввод в действие с 2013-08-01. – М., 2012.
20. РТМ 32 ЦШ 1115842.01–94. Руководящий технический материал. Безопасность железнодорожной автоматики и телемеханики. Методы и принципы обеспечения безопасности микроэлектронных СЖАТ. – СПб., 1994. – 119 с.
21. Харлап С. Н. Разработка высоконадежных систем на основе метода взаимной проверки аксиоматических базисов / С. Н. Харлап, Б. В. Сивко // Надежность. – 2016. – № 1.
22. Бочков К. А. Методы и средства доказательства функциональной безопасности микроэлектронных систем железнодорожной автоматики / К. А. Бочков, С. Н. Харлап, Д. Н. Шевченко // Електромагнітна сумісність та безпека на залізничному транспорті. – Днепропетровск : ДНУЗТ, 2011. – № 2. – С. 73–81.
23. Бочков К. А. Оценка временных параметров функционирования микропроцессорных устройств связи с объектами систем железнодорожной автоматики и телемеханики / К. А. Бочков, С. Н. Харлап, Б. В. Сивко // Вестник БелГУТа. Наука и транспорт. – 2012. – № 2 (25). – С. 12–15.

Bochkov Konstantin A.,
Kharlap Sergey N.,
Sivko Boris V.

Research laboratory
«Safety and electromagnetic compatibility of technical facilities»,
Belarusian State University of Transport, Gomel

Design of axiomatic based fault-tolerant systems

The article covers the solution of the problem of building a fault-tolerant and trustworthy microprocessor systems of railway automation and remote control using diversity axiomatic bases. As a system under the study the microprocessor device, performing the calculation of rolling stock axes, are used. Based on the simulation tests the qualitative comparison of fault tolerance and trustworthy of such systems with and without considered diversity are carried out. The article presents an example of sequential increasing of diversity, according axiomatic-based approach. The considered steps are: separation of memory and registers, separation of addresses, separation of sets of commands of the microprocessor and the protection of the software counter. It is also presents the formalization of conditions of diversity and the general cause failure protection, based on this. The example shows that diversity of axiomatic bases and self-testing of a common base solved the problem of dangerous failure detection. In this case the diversity solves the problem of detection of dangerous failures of diversity bases, and self-testing detects a maskable failures. It is also experimentally determined, that breakdown of a common base results in a general cause failures, and thus its mandatory control is justified. It was found, that application of this approach accelerates the development for rising the software diversity. It was revealed, that during the design and development of diversity, it is possible to create diversity high-level abstractions, that allows to select the level of diversity abstraction. The article covers the particularities of widespread application of this approach for development and verification of trustworthy and fault-tolerant systems.

fault tolerance; safety proof; diversity; formal methods; critical systems of data infrastructure; failure detection

References

1. Smith D. J., Kenneth G. L. Simpson. Safety Critical Systems Handbook. A Straightforward Guide to Functional Safety, IEC 61508 and Related Standards, Including Process IEC 61511 and Machinery IEC 62061 and ISO 13849. Oxford, UK, Elsevier Ltd, 2010, 270 p.
2. Parry G. W. Common Cause Failure Analysis: A Critique and Some Suggestions. Gaithersburg, Maryland, USA, Reliability Engineering and System Safety, 1991, vol. 34, issue 3, pp. 309–326.
3. Neumann P. G. Computer-Related Risks. NY, USA, Addison-Wesley Professional, 1995, 384 p.

4. Leveson N. Safeware: System Safety and Computers. NY, USA, Addison-Wesley, 1995, 680 p.
5. Weil V., Ferry B. Professional Responsibility for Harmful Actions. Kendall Hunt, Dubuque, Iowa, 1984, pp. 402–411.
6. Sagan S. D. The Limits of Safety: Organizations, Accidents, and Nuclear Weapons. Princeton University Press, Princeton, N. J., 1993, 302 p.
7. Bochkov K. A., Kovriga A. N., Kharlap S. N. Railway transport microprocessor automation systems: textbook. Gomel, BelGUT, 2013, 254 p.
8. Chen L., Avizienis A. N-Version Programming: A Fault-Tolerance Approach to Reliability of Software Operation. FTCS-8: Proceedings of the Eighth Annual International Conference on Fault Tolerant Computing, Toulouse, France, 1978, pp. 3–9.
9. Knight J. C., Leveson N. G. An Experimental Evaluation of the Assumption of Independence in Multiversion Programming. IEEE Transactions on Software Engineering, USA, N.J., 1986, vol. 12, issue 1, pp. 96–109.
10. Brilliant S., Knight J. C., Leveson N. G. Analysis of Faults in an N-Version Software Experiment. IEEE Transactions on Software Engineering, Virginia Univ., Charlottesville, VA, USA, 1990, vol. 16, issue 2, pp. 238–247.
11. Brilliant S., Knight J. C., Leveson N. G. The Consistent Comparison Problem in N-Version Programming. IEEE Transactions on Software Engineering, Virginia Commonwealth Univ., Richmond, VA, USA, 1989, vol. 15, issue 11, pp. 1481–1485.
12. Shubinsky I. B. Functional reliability of data systems. Methods of analysis. Ul'yanovsk, Publishing house «Nadezhnost'», 2012, 216 p.
13. Sivko B. V. Axiomatic-based approach for development of trustworthy and fault-tolerant systems. Automation on Transport (Avtomatika na transporte), 2015, vol. 1, issue 4, pp. 381–399.
14. Sivko B. V. Diversity axiomatic bases for development of trustworthy and fault-tolerant systems. Bulletin of BelGUT, Science and transport (Nauka i transport), 2014, issue 1 (28), pp. 19–23.
15. Kirilenko A. G., Grusha A. V. Axis couter in railway automation and remote control systems: textbook, Khabarovsk, Publishing house DVGUPS, 2003, 75 p.
16. Sapozhnikov Val. V., Sapozhnikov Vl. V., Khristov Kh. A., Gavzov D. V. Methods for bulding up safe microelectronic systems of railway automation. Moscow, Transport, 1995, 272 p.
17. Til'k I. G. New railway automation and remote control devices. Ekaterinburg, UrGUPS, 2010, 168 p.
18. Martin B. PIC Microcontrollers An Introduction to Microelectronics. Meppel, The Netherlands, Elsevier, 2nd edition, 2004, 372 p.
19. GOST R IEC 61508-2–2012. Functional safety of electrical/electronic/programmable electronic safety-related systems. Part 2. Publication date 2013-08-01.
20. RTM 32 CSh 1115842.01–94. Technical guides. Safety of railway automation and remote control. Methods and principles for microelectronic SzhaT safety. St. Petersburg, 1994, 119 p.
21. Khaclap S. N., Sivko B. V. Development of high-reliable systems based on the method of cross-check of axiomatic bases. Nadejnost', 2016, No 1.

22. Bochkov K.A., Kharlap S.N., Shevchenko D.N. Methods and tools in the proof of functional safety of railway automation systems. Electromagnetic compability and safety in railway transport. Dnepropetrovsk, DIIT, 2011, № 2, pp. 73–81.
23. Bochkov K.A., Kharlap S.N., Sivko B.V. Time parameters assessment of microprocessor-based devices for communication with components of railway automation and remote control systems. Bulletin of BelGUT, Science and transport (Nauka i transport), 2012, № 2 (25), pp. 12–15.

Статья представлена к публикации членом редколлегии Д.С. Марковым
Поступила в редакцию 06.08.2015, принята к публикации 16.10.2015

БОЧКОВ Константин Афанасьевич – проректор по научной работе, доктор технических наук, профессор, руководитель научно-исследовательской лаборатории «Безопасность и ЭМС технических средств» Белорусского государственного университета транспорта.
e-mail: bochkov1999@mail.ru

ХАРЛАП Сергей Николаевич – кандидат технических наук, доцент Белорусского государственного университета транспорта.
e-mail: hsn2007@belsut.gomel.by

СИВКО Борис Витальевич – инженер-программист, магистр технических наук, сотрудник научно-исследовательской лаборатории «Безопасность и ЭМС технических средств» Белорусского государственного университета транспорта.
e-mail: bsivko@gmail.com