Russian Technological Journal.
2023;11(5):54–62

ISSN 2782-3210 (Print)
ISSN 2500-316X (Online)

Микро- и наноэлектроника. Физика конденсированного состояния

Micro- and nanoelectronics. Condensed matter physics

УДК 004.052.2
https://doi.org/10.32362/2500-316X-2023-11-5-54-62

НАУЧНАЯ СТАТЬЯ

Применение кодов с исправлением двух ошибок для защиты конфигурационной памяти программируемой логики от действия космической радиации

Е.С. Лепёшкина [@], Н.Д. Кустов, В.Х. Ханов

Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева, Красноярск, 660037 Россия
[@] Автор для переписки, e-mail: klepka1111.93@mail.ru

Резюме

Цели. Программируемая логика типа field programmable gate array (FPGA) на основе статической конфигурационной памяти широко применяется в электронике бортовых систем космических аппаратов. Под воздействием космической радиации в конфигурационной памяти FPGA могут возникать ошибки. Основными методами защиты от них являются различные варианты резервирования триггеров, а также применение помехоустойчивых кодов в специальных схемах детектирования и исправления ошибок. Цель работы – определение из группы помехоустойчивых кодов тех, которые с учетом их избыточности наилучшим образом подходят для реализации внутреннего скраббинга конфигурационной памяти программируемых логических интегральных схем.

Методы. В работе рассмотрены методы скраббинга конфигурационной памяти FPGA, которые применяются для ее очистки от ошибок, вызванных действием космической радиации. Предлагается для повышения эффективности внутреннего скраббинга конфигурационной памяти FPGA использовать коды, исправляющие как однократные, так и двукратные смежные ошибки SEC-DED-DAEC. В этом случае уменьшается необходимость выполнения внешнего скраббинга конфигурационной памяти путем ее перезаписи эталонной конфигурацией из энергонезависимой радиационно-стойкой памяти. Таким образом, снижается время неработоспособного состояния FPGA, вызванное процедурой внешнего скраббинга. В связи с тем, что известные коды SEC-DED-DAEC имеют ненулевую вероятность ошибочного детектирования, а затем – ошибочного исправления двойной несмежной ошибки, а также обладают разной избыточностью и сложностью реализации, было приведено исследование наиболее эффективного кода для внутреннего скраббинга.

Результаты. Результаты исследования показали, что наилучшими с указанных позиций являются коды Датта, Нила и Хоюна – Йонгсурка. Приведены результаты сравнения кодов по выбранным критериям. Даны рекомендации для выбора конкретного кода в зависимости от возможных требований к планируемой космической миссии.

Выводы. Проведенное исследование показало эффективность защиты памяти программируемой логики с помощью применения кодов с исправлением двух ошибок.

Ключевые слова: программируемая логическая интегральная схема, сбой в конфигурационной памяти, методы очистки конфигурационной памяти от сбоев, коды с исправлением двух смежных ошибок

• **Поступила:** 30.01.2023 • **Доработана:** 17.05.2023 • **Принята к опубликованию:** 07.07.2023

Для цитирования: Лепёшкина Е.С., Кустов Н.Д., Ханов В.Х. Применение кодов с исправлением двух ошибок для защиты конфигурационной памяти программируемой логики от действия космической радиации. *Russ. Technol. J.* 2023;11(5):54–62. https://doi.org/10.32362/2500-316X-2023-11-5-54-62

Прозрачность финансовой деятельности: Авторы не имеют финансовой заинтересованности в представленных материалах или методах.

Авторы заявляют об отсутствии конфликта интересов.

RESEARCH ARTICLE

Application of double-error correction codes to protect configuration programmable logic memory against space radiation

Ekaterina S. Lepeshkina [@], Nikita D. Kustov, Vladislav Kh. Khanov

Reshetnev Siberian State University of Science and Technology, Krasnoyarsk, 660037 Russia
[@] Corresponding author, e-mail: klepka1111.93@mail.ru

Abstract

Objectives. Programmable logic integrated circuits of the field programmable gate array (FPGA) type based on static configuration memory are widely used in the electronics of onboard spacecraft systems. Under the influence of space radiation, errors may occur in the FPGA configuration memory. The main methods of protection against such errors involve various options for reservation triggers, as well as the use of error-correcting codes in special error detection and correction circuits. The purpose of the present work is to determine which error-correcting codes are best suited to the implementation of internal scrubbing of the FPGA configuration memory taking redundancy into account.

Methods. The paper analyses various methods for scrubbing FPGA configuration memory, which are used to correct errors caused by the action of space radiation. It is proposed to increase the efficiency of internal scrubbing of the FPGA configuration memory using codes that correct both single- and double-adjacent SEC-DED-DAEC errors. In this case, the need to perform external scrubbing of the configuration memory is reduced by overwriting it with a reference configuration from non-volatile radiation-resistant memory; in this way, FPGA downtime caused by the external scrubbing procedure is reduced. Due to the known SEC-DED-DAEC codes having a non-zero probability of erroneous detection and subsequent erroneous correction of a double non-adjacent error, as well as various redundancy and implementation complexities, a study was made of the most efficient code for internal scrubbing.

Results. The results showed that the Datta, Neale and Hoyoon–Yongsurk codes are optimal from the indicated positions. Recommendations are given for selecting a specific code depending on the specific requirements for a particular planned space mission.

Conclusions. The study confirms the effectiveness of protecting the memory of programmable logic by using two-error-correcting codes.

Keywords: programmable logic integrated circuits, faults in configuration memory, methods for clearing configuration memory from faults, double-adjacent error-correcting codes

• **Submitted:** 30.01.2023 • **Revised:** 17.05.2023 • **Accepted:** 07.07.2023

For citation: Lepeshkina E.S., Kustov N.D., Khanov V.Kh. Application of double-error correction codes to protect configuration programmable logic memory against space radiation. *Russ. Technol. J*. 2023;11(5):54–62. https://doi.org/10.32362/2500-316X-2023-11-5-54-62

Financial disclosure: The authors have no a financial or property interest in any material or method mentioned.

The authors declare no conflicts of interest.

ВВЕДЕНИЕ

В настоящее время наблюдается значительный прогресс в развитии космической электроники, что позволило существенно расширить функционал современных космических аппаратов (КА), уменьшив при этом габариты, массу и энергопотребление бортовых систем. Вместе с тем остается актуальной задача защиты аппаратуры от радиационных сбоев в электронных компонентах, тем более что на малых КА начала широко использоваться дешевая коммерческая электронная компонентная база (ЭКБ), в которой отсутствуют многие механизмы защиты от сбоев, характерные для радиационно-устойчивой космической ЭКБ. В аппаратуре, основанной на коммерческой ЭКБ, в большей степени возможны к применению лишь методы защиты от одиночных сбоев (single event upset, SEU), вызванных разовыми событиями попадания заряженных частиц радиации в электронные компоненты [1]. Эти сбои вызывают логические ошибки без разрушения компонента, т.е. они не вызывают необратимых процессов в полупроводниковой структуре компонента [2]. Сбои SEU возникают в триггерах, которыми насыщена любая электронная аппаратура. Вероятность таких сбоев достаточна высока, в этом и состоит их опасность.

Основными методами защиты от сбоев SEU являются различные варианты резервирования, в т.ч. метод тройного модульного резервирования (triple modular redundancy, TMR) [3, 4] триггеров, а также применение помехоустойчивых кодов (error correction codes, ECC) в специальных схемах детектирования и исправления ошибок (error detection and correction, EDAC) [5, 6], обычно исправляющих однократную ошибку (ошибку в одном бите) в структуре из нескольких триггеров, например, в слове статической памяти SRAM (static random access memory).

Одним из широко используемых компонентов в космическом приборостроении являются программируемые логические интегральные схемы типа SRAM FPGA (static random access memory field programmable gate array). Данные программируемые логические интегральные схемы (ПЛИС), в основном относящиеся к коммерческой категории микросхем, отличаются большим количеством программируемых элементов, низким энергопотреблением, высокой скоростью работы имплементированных в них схем, но низкой сбоеустойчивостью к событиям SEU [7]. Ошибки SEU могут произойти в триггерах реализованной на SRAM FPGA схемы, а также в конфигурационной памяти SRAM. Последний вид ошибок является наиболее опасным, т.к. он изменяет структуру имплементированной схемы, что вызывает постоянные периодические сбои всякий раз, когда через поврежденный участок схемы проходят сигналы [8].

Основным методом защиты SRAM FPGA от SEU является скраббинг (от англ. scrubbing), который заключается в перезаписи эталонной конфигурацией сбойного содержимого конфигурационной памяти при обнаружении ошибки в имплементированной схеме [9, 10].

В данной работе рассмотрены различные варианты скраббинга, проанализированы возможности применения ECC для скраббинга. В этой связи предположено применение кодов типа SEC-DED-DAEC (single error correction, double error detection and double adjacent error correction), проведена оценка эффективности известных кодов SEC-DED-DAEC для целей скраббинга конфигурационной памяти SRAM FPGA.

СВЯЗАННЫЕ РАБОТЫ

Содержимое конфигурационной памяти SRAM FPGA может быть искажено при воздействии радиации. Поэтому для их нормальной работы в среде, подверженной радиации, например, в космическом пространстве, необходимо обнаруживать ошибки в конфигурационной памяти и быстро их устранять. Устранение ошибок в конфигурационной памяти производится с помощью скраббинга – метода ее перезаписи эталонной конфигурацией.

При этом существуют два способа перезаписи: полный и частичный. Полный скраббинг состоит в перезаписи всего содержимого конфигурационной памяти, частичный скраббинг – это поблочная перезапись конфигурационной памяти [11], например, последовательно блок за блоком. Во втором случае уменьшается время неработоспособности системы, вызванное процессом загрузки полной конфигурации.

Скраббинг может быть «слепой» и «зрячий» [12]. «Слепой» скраббинг состоит в периодической перезаписи полной или последовательно поблочной конфигурационной памяти. Период перезаписи

выбирают из предполагаемой интенсивности ошибок, например, в зависимости от высоты орбиты КА. «Зрячий» скраббинг выполняют после обнаружения ошибки, возможно, при локализации блока конфигурационной памяти, накоплении ошибок и невозможности исправить их каким-либо другим дополнительным методом.

Существуют прямой и косвенный способы определения ошибок в конфигурационной памяти. Косвенный метод заключается в нахождении ошибок во внутренней системе ПЛИС, сконфигурированной с помощью содержимого конфигурационной памяти. Если в конфигурационной памяти произошел сбой, то он проявится в системе в виде ошибки, которую могут выявить (и заодно исправить) блоки TMR, распределенные в системе. При неоднократной (последовательно 3–5 раза) выявленной ошибке следует запустить процедуру скраббинга [11].

Прямой способ заключается в периодическом поблочном сравнении содержимого конфигурационной памяти с соответствующим блоком эталонной конфигурации, находящимся в энергонезависимой и радиационно-устойчивой памяти. При этом для ускорения работы могут использоваться коды ECC. В этом случае в каждом блоке эталонной конфигурации в энергонезависимой памяти хранится также его контрольная сумма. При обратном чтении соответствующего блока из конфигурационной памяти ПЛИС рассчитывается его контрольная сумма, которая сравнивается с соответствующей суммой из энергонезависимой памяти. Если суммы не совпадают, значит, в данном блоке конфигурационной памяти произошел сбой и этот блок следует перезаписать [8].

Современные ПЛИС компании Xilinx[1] имеют основанные на применении EDAC и кодов single error correction and double error detection (SEC-DED) встроенные механизмы поблочного сканирования конфигурационной памяти и автоматического исправления в ней однократных ошибок, а также обнаружения двукратных ошибок в сканированном блоке [8]. Данный подход, в дальнейшем называемый в работе внутренним скраббингом, позволяет уменьшить количество запусков внешнего частичного скраббинга конфигурационной памяти для исправления ошибок двойной и большей кратности, а значит – уменьшить время простоя имплементированной в ПЛИС системы, связанного с выполнением внешнего скраббинга.

ПОСТАНОВКА ЗАДАЧИ

Известно, что коды SEC-DED исправляют один сбой в слове памяти. Используемый код SEC-DED должен быть малоизбыточным, быстрым и простым в реализации (для ПЛИС – это малое количество логических элементов для реализация конкретного кода). Этим критериям лучше всего соответствует код Хсяо (39, 32), где 39 – размер кодового слова в битах, 32 – информационного [13], принадлежащий к группе модифицированных кодов Хэмминга [14].

С развитием новых технологий создания ЭКБ и переходом к более тонким технологическим процессам производства компонентов повышается вероятность мультибитных сбоев, в первую очередь, двухбитных [15]. Одним из способов решения данной проблемы является применение кодов SEC-DED-DAEC [16]. Коды, принадлежащие к этой группе, исправляют 2 смежные ошибки (смежные – значит находящихся в двух соседних битах одного слова памяти). Они также принадлежат к группе модифицированных кодов Хэмминга, поэтому являются быстрыми и имеют малую избыточность. Любой SEC-DED-DAEC-код справляется с задачами коррекции одиночной и двойной смежной ошибки в полной мере. Но эти коды обладают одним специфическим недостатком: существует вероятность неправильного исправления несмежных двукратных ошибок, причем у одного кода из данной группы эта вероятность больше, а у другого – меньше.

Отметим, что вероятность смежной двойной ошибки в слове памяти существенно выше, чем вероятность появления двойной несмежной ошибки. Двойная смежная ошибка – это ошибка от одного события SEU, охватившего два соседних бита одного слова. Двойная несмежная ошибка – это накопление ошибок. Сначала произошел сбой, вызвавший ошибку в одном бите слова памяти, затем, по прошествии времени, произошел сбой, вызвавший ошибку в другом бите этого же слова памяти, и в результате образовалась двойная несмежная ошибка. Понятно, что такой процесс маловероятен, но исключить его нельзя. Поэтому исключить из анализа кодов SEC-DED-DAEC оценку вероятности неправильного исправления несмежных двукратных ошибок было бы неправильно.

Предлагается для повышения эффективности внутреннего скраббинга конфигурационной памяти ПЛИС использовать SEC-DED-DAEC-коды. В этом случае в конфигурационной памяти исправляются и однократные, и двукратные смежные ошибки, что позволит меньше прибегать к внешнему скраббингу конфигурационной памяти путем ее перезаписи эталонной конфигурацией из энергонезависимой радиационно-стойкой памяти.

Задача исследования состоит в следующем. В связи с тем, что количество SEC-DED-DAEC-кодов составляет некоторое множество, каждый из них имеет свои параметры, необходимо провести исследования для получения оценки вероятности ложного

[1] https://www.xilinx.com/ (in Russ.). Дата обращения 16.02.2023. / Accessed February 16, 2023.

обнаружения (а затем и ложного исправления) двукратной несмежной ошибки и требуемых ресурсов для реализации кодера/декодера, и таким образом определить из них те, которые, дополнительно, с учетом их избыточности, лучшим образом подходят к реализации внутреннего скраббинга конфигурационной памяти ПЛИС.

ПРОВЕДЕНИЕ ИССЛЕДОВАНИЯ

Как было отмечено ранее, для SEC-DED-DAEC-кодов существует вероятность ошибочного детектирования двукратной несмежной ошибки как двукратной смежной, что приводит впоследствии к ошибочной коррекции битов кодового слова. Определение такой вероятности для того или иного кода проведено при помощи функционального моделирования.

Для этого была разработана функциональная модель, представленная в виде программы на языке C++, разработанной в среде *Microsoft Visual Studio*[2]. Ниже описаны основные логические части программы:

1) блок инициализации переменных (порождающей **G**-матрицы и проверочной **H**-матрицы);

2) блок генерации информационного слова (одномерного булевого массива) с использованием псевдослучайной функции;

3) блок кодирования информационного слова путем перебора столбцов **G**-матрицы. При появлении единицы производится операция XOR над текущим битом кодового слова и соответствующим битом информационного слова;

4) блок внесения двукратной несмежной ошибки с использованием псевдослучайной функции для определения случайных несмежных позиций и последующего инвертирования битов кодового слова;

5) блок определения синдрома ошибки путем перебора столбцов **H**-матрицы. При появлении единицы производится операция XOR над текущим битом синдрома и соответствующим битом кодового слова;

6) блок детектирования двукратной смежной ошибки путем сравнения полученного синдрома с синдромами двухкратных смежных ошибок, являющихся результатом операции XOR над двумя соседними столбцами **H**-матрицы;

7) блок детектирования двукратной несмежной ошибки. Двукратная несмежная ошибка детектируется, если двухкратная смежная ошибка не обнаружена.

Алгоритм проходит большое количество итераций. Вероятность безошибочного детектирования двойной несмежной ошибки определяется отношением количества исходов детектирования двукратной несмежной ошибки к общему количеству кодовых слов с внесенной двукратной несмежной ошибкой, прошедших через алгоритм.

Для того, чтобы определить ресурсы, требуемые для реализации кодера и декодера кодов (количество логических элементов), проведено имитационное моделирование. Для этого была разработана программа на языке VHDL в среде разработки *Quartus*[3]. Функциональная отладка производилась в среде *ModelSim*[4].

Логические части кодера:

1) блок инициализации переменных (тактового сигнала CLK, входного информационного слова и выходного кодового слова);

2) блок кодирования. При изменении сигнала CLK запускается цикл кодирования битов контрольной суммы в соответствии с **G**-матрицей при помощи операции XOR;

3) блок вывода кодового слова (исходного информационного слова и контрольной суммы).

Логические части декодера:

1) блок инициализации переменных (тактового сигнала CLK, входного кодового слова, выходного откорректированного слова);

2) блок определения синдрома ошибки. При изменении сигнала CLK запускается цикл для битов кодового слова, и в соответствии с **H**-матрицей подсчитывается синдром ошибки;

3) блок декодирования. Сначала проверяется случай без ошибки (если синдром является нулевым). Если синдром ненулевой, происходит сравнение полученного синдрома с синдромами одиночных ошибок и коррекция ошибок. Если совпадений с синдромами одиночных ошибок нет, то происходит сравнение с синдромами двойных смежных ошибок и коррекция ошибок. Если совпадений с синдромами нет, то детектируется несмежная ошибка;

4) блок вывода откорректированного слова.

Имитационное моделирование проводилось для ПЛИС FPGA Cyclone IV E EP4CE6E22A7 (производитель – Intel, США). В результате синтеза подсчитывалось количество используемых логических элементов кодера и декодера.

В соответствии с представленным описанием были разработаны функциональные и имитационные модели для нескольких кодов. Основной критерий выбора исследуемых кодов – явное описание в научно-технической литературе **H**-матриц кодов. В этом

[2] https://visualstudio.microsoft.com/ru/ (in Russ.). Дата обращения 16.02.2023. / Accessed February 16, 2023.

[3] http://altera.ru/soft_quartus.html. Дата обращения 16.02.2023. / Accessed February 16, 2023.

[4] https://altera.co.uk/products/software/quartus-ii/modelsim/qts-modelsim-index.html. Дата обращения 16.02.2023. / Accessed February 16, 2023.

случае исключается вероятность некорректной генерации проверочной матрицы, упрощается процесс исследования и минимизируется вероятность получения погрешности в моделируемых результатах. Таким образом, в качестве исследуемых были выбраны следующие коды с 32-битным информационным словом: Дутта (39, 32) [17], Датта (42, 32) [18], Нила (42, 32) [19], Ревириего (39, 32) [20], Ча – Юна (39, 32) [21], Хоюна – Йонгсурка (41, 32) [22].

Помимо кодов SEC-DED-DAEC представлены для сравнения данные для одного SEC-DED-кода, а именно, для кода Хсяо [13], который широко используется при реализации механизма EDAC для памяти.

РЕЗУЛЬТАТЫ

Результаты проведенных исследований представлены в таблице. Приведенные данные показывают, что лидирующие позиции заняты кодами Датта, Нила и Хоюна – Йонгсурка. Код Хсяо показывает средние значения по выбранным критериям сравнения, при этом он не исправляет 2 смежные ошибки. Сравнивая их между собой применительно к использованию для сканирования конфигурационной SRAM-памяти ПЛИС, можно сделать следующие рекомендации.

Таблица. Сравнение кодов SEC-DED-DAEC по критериям безошибочного детектирования двойной несмежной ошибки и сложности реализации

Код	Вероятность безошибочного детектирования двойной несмежной ошибки	Количество логических элементов кодера/декодера
Хсяо (39, 32)	63.4	164/332
Дутта (39, 32)	43.5	170/407
Датта (40, 32)	78.9	164/354
Нила (42, 32)	84.4	57/391
Ревириего (39, 32)	38.4	175/373
Ча – Юна (39, 32)	60.7	123/272
Хоюна – Йонгсурка (41, 32)	95.7	178/384

Для кода Датта избыточность кодового слова выровнена к размерности байта, что может упростить построение конфигурационной памяти ПЛИС. Вероятность ошибочного детектирования двойной несмежной ошибки хотя и является худшей рассматриваемой тройки кодов, но немного уступает по этому показателю коду Нила. Сложности кодера и декодера кода низкие, сложность кодера средние.

Для кода Нила избыточность кодового слова не выровнена по границе байта. Вероятность ошибочного детектирования двойной несмежной ошибки средняя, но на достаточно высоком уровне. Сложность кодера очень низкая, сложность декодера самая высокая.

Избыточность для кода Хоюна – Йонгсурка также не выровнена по границе байта. Вероятность ошибочного детектирования двойной несмежной ошибки наилучшая. Сложность кодера наивысшая, сложность декодера средняя.

Выбор для реализации того или иного кода необходимо осуществить, учитывая исходные данные к планируемой космической миссии. Если в проекте критичны простота и скорость реализации при приемлемой вероятности ложного детектирования двойной несмежной ошибки в условиях низкой интенсивности радиационных сбоев, можно рекомендовать применять в EDAC-механизме конфигурационной памяти код Датта. Код Нила может также претендовать на эту позицию, но он имеет наибольшую избыточность, причем не выровненную по границе байта. Если планируемая космическая миссия долговременна и будет осуществляться в условиях высокой интенсивности сбоев, то лучше использовать код Хоюна – Йонгсурка, несмотря на его относительную сложность реализации и низкую производительность.

Кроме того, отметим, что сложность анализируемой тройки кодов, выраженная в количестве необходимых логических элементов для реализации кодера/декодера, в среднем соответствует сложности обычно применяемого кода Хсяо. Поэтому их реализация для целей сканирования конфигурационной памяти с целью скраббинга с точки зрения использования ресурсов ПЛИС не вызывает проблем.

ЗАКЛЮЧЕНИЕ

В работе предложено изменить алгоритм метода внутреннего сканирования для внутреннего скраббинга конфигурационной памяти SRAM ПЛИС: вместо одного из кодов SEC-DED, например, кода Хсяо, предлагается использовать один из кодов SEC-DED-DAEC. В этом случае в конфигурационной SRAM-памяти будет исправляться и одна ошибка, и две смежные ошибки, что уменьшит количество запусков внешнего частичного скраббинга конфигурационной памяти для исправления ошибок двойной и большей кратности, а значит, уменьшит время простоя имплементированной в ПЛИС системы, связанного с выполнением скраббинга.

Коды SEC-DED-DAEC обладают одним отрицательным свойством – ненулевой вероятностью ошибочного детектирования (а затем ошибочного исправления) двойной несмежной ошибки. Кроме

того, они обладают разной избыточностью для хранения контрольной суммы в SRAM-памяти, а также различной сложностью реализации, которую можно оценить по требуемому количеству логических элементов для реализации кодера/декодера. Было приведено исследование по определению наиболее эффективного кода в соответствии с данными критериями среди SEC-DED-DAEC-кодов с известными проверочными матрицами. Результаты исследования показали, что наилучшими с указанных позиций являются коды Датта, Нила и Хоюна – Йонгсурка. Код Хоюна – Йонгсурка обладает практически нулевой вероятностью ошибочного детектирования двойной смежной ошибки, но при этом имеет наибольшую сложность. Код Датта наиболее прост в реализации, но вероятность ошибочного детектирования двойной несмежной ошибки равна приблизительно 20%. Код Нила занимает промежуточную позицию. При проектировании выбор конкретного кода должен определяться требованиями к планируемой космической миссии.

Благодарности

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 19-38-90052.

Acknowledgments

The reported study was supported by the Russian Foundation for Basic Research, project No. 19-38-90052.

Вклад авторов. Все авторы в равной степени внесли свой вклад в исследовательскую работу.

Authors' contribution. All authors equally contributed to the research work.

СПИСОК ЛИТЕРАТУРЫ

1. Максименко С.Л., Мелехин В.Ф., Филиппов А.С. Анализ проблемы построения радиационно-стойких информационно-управляющих систем. *Информационно-управляющие системы*. 2012;2(57):18–25. URL: http://www.i-us.ru/index.php/ius/article/view/13788
2. Gaillard R. Single Event Effects: Mechanisms and Classification. In: Nicolaidis M. (Ed.). *Soft Errors in Modern Electronic Systems. Frontiers in Electronic Testing*. Boston, MA: Springer; 2011. V. 41. P. 27–54. https://doi.org/10.1007/978-1-4419-6993-4_2
3. Kastensmidt F.L., Sterpone L., Carro L., Reorda M.S. On the Optimal Design of Triple Modular Redundancy Logic for SRAM-based FPGAs. In: *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*. 2005. P. 1290–1295. https://doi.org/10.1109/DATE.2005.229
4. Cheng D., Qi D., Chen M. Radiation-hardened Test Design for Aerospace SoC. In: *2020 IEEE 5th International Conference on Integrated Circuits and Microsystems* (*ICICM*). 2020. P. 213–217. https://doi.org/10.1109/ICICM50929.2020.9292308
5. Mang I., Mang E., Popescu C. VHDL implementation of an error detection and correction module based on Hamming code. *J. Comput. Sci. Control Syst.* 2011;4(2):43–46.
6. Baviera E., Schettino G.M., Tuniz E., Vatta F. Software Implementation of Error Detection and Correction Against Single-Event Upsets. In: *2020 International Conference on Software, Telecommunications and Computer Networks* (*SoftCOM*). 2020. https://doi.org/10.23919/SoftCOM50211.2020.9238173
7. Максфилд К. *Проектирование на ПЛИС. Архитектура, средства и методы: курс молодого бойца*: пер. с англ. М.: Додэка-XXI; 2007. 408 с.

REFERENCES

1. Maksimenko S.L., Melekhin V.F., Filippov A.S. Analysis of the problem of radiation-tolerant information and control-systems implementation. *Informatsionno-upravlyayushchie sistemy = Information and Control Systems.* 2012;2(57):18–25 (in Russ.). Available from URL: http://www.i-us.ru/index.php/ius/article/view/13788
2. Gaillard R. Single Event Effects: Mechanisms and Classification. In: Nicolaidis M. (Ed.). *Soft Errors in Modern Electronic Systems. Frontiers in Electronic Testing*. Boston, MA: Springer; 2011. V. 41. P. 27–54. https://doi.org/10.1007/978-1-4419-6993-4_2
3. Kastensmidt F.L., Sterpone L., Carro L., Reorda M.S. On the Optimal Design of Triple Modular Redundancy Logic for SRAM-based FPGAs. In: *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*. 2005. P. 1290–1295. https://doi.org/10.1109/DATE.2005.229
4. Cheng D., Qi D., Chen M. Radiation-hardened Test Design for Aerospace SoC. In: *2020 IEEE 5th International Conference on Integrated Circuits and Microsystems* (*ICICM*). 2020. P. 213–217. https://doi.org/10.1109/ICICM50929.2020.9292308
5. Mang I., Mang E., Popescu C. VHDL implementation of an error detection and correction module based on Hamming code. *J. Comput. Sci. Control Syst.* 2011;4(2):43–46.
6. Baviera E., Schettino G.M., Tuniz E., Vatta F. Software Implementation of Error Detection and Correction Against Single-Event Upsets. In: *2020 International Conference on Software, Telecommunications and Computer Networks* (*SoftCOM*). 2020. https://doi.org/10.23919/SoftCOM50211.2020.9238173
7. Maksfild K. *Proektirovanie na PLIS. Arkhitektura, sredstva i metody: kurs molodogo boitsa* (*The Design Warrior's Guide to FPGAs: Devices, Tools and Flows*.): transl. from Engl. Moscow: Dodeka-XXI; 2007. 408 p. (in Russ.).

8. Виртлин М., Хардинт А. Гибридная очистка конфигурации для ПЛИС компании Xilinx. В кн.: *ПЛИС и параллельные структуры в аэрокосмической области. Программные ошибки и отказоустойчивое проектирование*; под ред. Ф. Канстеншмидт, П. Реха. М.: Техносфера; 2018. 326 с.
9. Соколов А. Программно-аппаратные методы повышения радиационной стойкости микросхем SRAM ПЛИС. *Современная электроника*. 2014;6:30–33.
10. Zhang R., Xiao L., Li J., Cao X., Li L. An Adjustable and Fast Error Repair Scrubbing Method Based on Xilinx Essential Bits Technology for SRAM-Based FPGA. *IEEE Transactions on Reliability*. 2020;69(2):430–439. https://doi.org/10.1109/TR.2019.2896897
11. Цетин Э., Диззель О., Ли Т. и др. Обзор и исследование методов обнаружения и устранения одиночных сбоев для гетерогенных систем на основе ПЛИС. В кн.: *ПЛИС и параллельные структуры в аэрокосмической области. Программные ошибки и отказоустойчивое проектирование*; под ред. Ф. Канстеншмидт, П. Реха. М.: Техносфера; 2018. 326 с.
12. Флеминг Ш.Т., Томас Д.В., Винтерстейн Ф. Энергосберегающая адаптивная платформа FDIR, применяющая модули гетерогенных систем на кристалле. В кн.: *ПЛИС и параллельные структуры в аэрокосмической области. Программные ошибки и отказоустойчивое проектирование;* под ред. Ф. Канстеншмидт, П. Реха. М.: Техносфера; 2018. 326 с.
13. Hsiao M.Y. A class of optimal minimum odd-weight-column SEC-DED codes. *IBM J. Res. Develop*. 1970;14(4):395–401. https://doi.org/10.1147/rd.144.0395
14. Hamming R.W. Error detecting and error correcting codes. *Bell System technical journal*. 1950;29(2):147–160. URL: https://www.sci-hub.ru/10.1002/j.1538-7305.1950.tb00463.x
15. Чумаков А.И., Согоян А.В., Боруздина А.Б. и др. Механизмы многократных сбоев в микросхемах памяти. *Проблемы разработки перспективных микро- и наноэлектронных систем* (*МЭС*). 2016;4:145–152. URL: http://www.mes-conference.ru/data/year2016/pdf/D188.pdf
16. Краснюк А.А., Петров К.А. Особенности применения методов помехоустойчивого кодирования в суб-100-нм микросхемах памяти для космических систем. *Микроэлектроника*. 2012;41(6):450–456.
17. Dutta A., Touba N.A. Multiple Bit Upset Tolerant Memory Using a Selective Cycle Avoidance Based SEC-DED-DAEC Code. In: *25th IEEE VLSI Test Symposium* (*VTS'07*). 2007. P. 349–354. https://doi.org/10.1109/VTS.2007.40
18. Datta R., Touba N.A. Exploiting unused spare columns to improve memory ECC. In: *27th IEEE VLSI Test Symposium*. 2009. P. 47–52. https://doi.org/10.1109/VTS.2009.52
19. Neale A., Sachdev M. A new SEC-DED error correction code subclass for adjacent MBU tolerance in embedded memory. *IEEE Transactions on Device and Materials Reliability.* 2013;13(1):223–230. https://doi.org/10.1109/TDMR.2012.2232671
20. Reviriego P., Liu S.S., Sánchez-Macián A., Xiao L., Maestro J.A. Unequal error protection codes derived from SEC-DED codes. *Electron. Lett*. 2016;52(8):619–620. https://doi.org/10.1049/el.2016.0077

[Maxfield C. *The Design Warrior's Guide to FPGAs: Devices, Tools and Flows*. Oxford, UK: Jordan Hill; 2004. 560 p.]

8. Virtlin M., Khardint A. Hybrid Clear Configuration for Xilinx FPGAs. In: *PLIS i parallel'nye struktury v aerokosmicheskoi oblasti. Programmnye oshibki i otkazoustoichivoe proektirovanie* (*FPGAs and Parallel Architectures for Aerospace Applications. Soft Errors and Fault-Tolerant Design*). Kastensmidt F., Rech P. (Eds.). Moscow: Tekhnosfera; 2018. 326 p. (in Russ.).
9. Sokolov A. Software and hardware methods of radiation hardening of microprocessors based control systems. *Sovremennaya Elektronika*. 2014;6:30–33 (in Russ.).
10. Zhang R., Xiao L., Li J., Cao X., Li L. An Adjustable and Fast Error Repair Scrubbing Method Based on Xilinx Essential Bits Technology for SRAM-Based FPGA. *IEEE Transactions on Reliability*. 2020;69(2):430–439. https://doi.org/10.1109/TR.2019.2896897
11. Tsetin E., Dizzel' O., Li T., et al. Review and study of methods for detecting and eliminating single failures for heterogeneous systems based on FPGAs. In: *PLIS i parallel'nye struktury v aerokosmicheskoi oblasti. Programmnye oshibki i otkazoustoichivoe proektirovanie* (*FPGAs and Parallel Architectures for Aerospace Applications. Soft Errors and Fault-Tolerant Design*). Kastensmidt F., Rech P. (Eds.). Moscow: Tekhnosfera; 2018. 326 p. (in Russ.).
12. Fleming Sh., Tomas D., Vintersteyn F. Energy-saving adaptive FDIR platform using heterogeneous system-on-chip modules. In book: *PLIS i parallel'nye struktury v aerokosmicheskoi oblasti. Programmnye oshibki i otkazoustoichivoe proektirovanie* (*FPGAs and Parallel Architectures for Aerospace Applications. Soft Errors and Fault-Tolerant Design*). Kastensmidt F., Rech P. (Eds.). Moscow: Tekhnosfera; 2018. 326 p. (in Russ.).
13. Hsiao M.Y. A class of optimal minimum odd-weight-column SEC-DED codes. *IBM J. Res. Develop*. 1970;14(4):395–401. https://doi.org/10.1147/rd.144.0395
14. Hamming R.W. Error detecting and error correcting codes. *Bell System technical journal*. 1950;29(2):147–160. Available from URL: https://www.sci-hub.ru/10.1002/j.1538-7305.1950.tb00463.x
15. Chumakov A.I., Sogoyan A.V., Boruzdina A.B., et al. Mechanisms of multiple cell upsets in memory. *Problemy razrabotki perspektivnykh mikro- i nanoelektronnykh sistem* (*MES*) = *Problems of Advanced Micro- and Nanoelectronic Systems Development.* 2016;4:145–152 (in Russ.).
16. Krasnyuk A.A., Petrov K.A. Application features of the error correction coding in sub-100-nm memory microcircuits for cosmic systems. *Russ. Microelectron.* 2013;42(1):53–58. https://doi.org/10.1134/S1063739712040087
[Original Russian Text: Krasnyuk A.A., Petrov K.A. Application features of the error correction coding in sub-100-nm memory microcircuits for cosmic systems. *Mikroelektronika.* 2012;41(6):450–456 (in Russ.).]
17. Dutta A., Touba N.A. Multiple Bit Upset Tolerant Memory Using a Selective Cycle Avoidance Based SEC-DED-DAEC Code. In: *25th IEEE VLSI Test Symposium* (*VTS'07*). 2007. P. 349–354. https://doi.org/10.1109/VTS.2007.40

21. Cha S., Yoon H. Efficient implementation of single error correction and double error detection code with check bit pre-computation for memories. *JSTS: J. Semiconductor Technol. Sci.* 2018;12(4):418–425. https://doi.org/10.5573/JSTS.2012.12.4.418
22. Hoyoon Jun, Yongsurk Lee. Protection of On-chip Memory Systems against Multiple Cell Upsets Using Double-adjacent Error Correction Codes. *Int. J. Computer Inform. Technol*. 2014;3(6):1316–1320. URL: https://www.ijcit.com/archives/volume3/issue6/Paper030621.pdf

18. Datta R., Touba N.A. Exploiting unused spare columns to improve memory ECC. In: *27th IEEE VLSI Test Symposium*. 2009. P. 47–52. https://doi.org/10.1109/VTS.2009.52
19. Neale A., Sachdev M. A new SEC-DED error correction code subclass for adjacent MBU tolerance in embedded memory. *IEEE Transactions on Device and Materials Reliability.* 2013;13(1):223–230. https://doi.org/10.1109/TDMR.2012.2232671
20. Reviriego P., Liu S.S., Sánchez-Macián A., Xiao L., Maestro J.A. Unequal error protection codes derived from SEC-DED codes. *Electron. Lett*. 2016;52(8):619–620. https://doi.org/10.1049/el.2016.0077
21. Cha S., Yoon H. Efficient implementation of single error correction and double error detection code with check bit pre-computation for memories. *JSTS: J. Semiconductor Technol. Sci.* 2018;12(4):418–425. https://doi.org/10.5573/JSTS.2012.12.4.418
22. Hoyoon Jun, Yongsurk Lee. Protection of On-chip Memory Systems against Multiple Cell Upsets Using Double-adjacent Error Correction Codes. *Int. J. Computer Inform. Technol*. 2014;3(6):1316–1320. Available from URL: https://www.ijcit.com/archives/volume3/issue6/Paper030621.pdf

Об авторах

Лепёшкина Екатерина Сергеевна, ассистент, кафедра безопасности информационных технологий; инженер, лаборатория «Малые космические аппараты», Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева (660037, Россия, Красноярск, просп. им. газеты «Красноярский рабочий», д. 31). E-mail: klepka1111.93@mail.ru. Scopus Author ID 57218577296, SPIN-код РИНЦ 8746-6555, https://orcid.org/0000-0001-5116-6260

Кустов Никита Дмитриевич, ассистент, кафедра безопасности информационных технологий; инженер, лаборатория «Малые космические аппараты», Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева (660037, Россия, Красноярск, просп. им. газеты «Красноярский рабочий», д. 31). E-mail: kustovnd@yandex.ru. Scopus Author ID 57218577358, SPIN-код РИНЦ 9031-4516, https://orcid.org/0000-0002-3362-3971

Ханов Владислав Ханифович, к.т.н., доцент, кафедра безопасности информационных технологий; заведующий лабораторией «Малые космические аппараты», Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева (660037, Россия, Красноярск, просп. им. газеты «Красноярский рабочий», д. 31). E-mail: khvkh@mail.ru. Scopus Author ID 56491191500, SPIN-код РИНЦ 5197-5699, https://orcid.org/0000-0001-6720-9405

About the authors

Ekaterina S. Lepeshkina, Assistant, Information Technology Security Department; Engineer, “Small Satellite” Laboratory, Reshetnev Siberian State University of Science and Technology (Reshetnev University). (31, Krasnoyarsky Rabochy pr., Krasnoyarsk, 660037 Russia). E-mail: klepka1111.93@mail.ru. Scopus Author ID 57218577296, RSCI SPIN-code 8746-6555, https://orcid.org/0000-0001-5116-6260

Nikita D. Kustov, Assistant, Information Technology Security Department; Engineer, “Small Satellite” Laboratory, Reshetnev Siberian State University of Science and Technology (Reshetnev University). (31, Krasnoyarsky Rabochy pr., Krasnoyarsk, 660037 Russia). E-mail: kustovnd@yandex.ru. Scopus Author ID 57218577358, RSCI SPIN-code 9031-4516, https://orcid.org/0000-0002-3362-3971

Vladislav Kh. Khanov, Cand. Sci. (Eng.), Associate Professor, Information Technology Security Department; Head of the “Small Satellite” Laboratory, Reshetnev Siberian State University of Science and Technology (Reshetnev University). (31, Krasnoyarsky Rabochy pr., Krasnoyarsk, 660037 Russia). E-mail: khvkh@mail.ru. Scopus Author ID 56491191500, RSCI SPIN-code 5197-5699, https://orcid.org/0000-0001-6720-9405