Федеральное государственное учреждение «Федеральный научный центр
Научно-исследовательский институт системных исследований
Российской академии наук»
(ФГУ ФНЦ НИИСИ РАН)

ТРУДЫ НИИСИ РАН

ТОМ 10 № 4

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ СЛОЖНЫХ СИСТЕМ:

ТЕОРЕТИЧЕСКИЕ И ПРИКЛАДНЫЕ АСПЕКТЫ

МОСКВА
2020

Тематика номера:

Проектирование и моделирование СБИС, математическое моделирование и визуализация систем виртуального окружения, информационные и компьютерные технологии

Журнал публикует оригинальные статьи по следующим областям исследований: математическое и компьютерное моделирование, обработка изображений, визуализация, системный анализ, методы обработки сигналов, информационная безопасность, информационные технологии, высокопроизводительные вычисления, оптико-нейронные технологии, микро- и наноэлектроника, математические исследования и вопросы численного анализа, история науки и техники.

The topic of the issue:

Design and modeling of VLSI, mathematical modeling and visualization of virtual environment systems, information and computer technologies

The Journal publishes novel articles on the following research arias: mathematical and computer modeling, image processing, visualization, system analysis, signal processing, information security, information technologies, high-performance computing, optical-neural technologies,
micro- and nanoelectronics, mathematical researches and problems of numerical analysis,
history of science and of technique.

Заведующий редакцией: В.Е.Текунов

Издатель: ФГУ ФНЦ НИИСИ РАН,
117218, Москва, Нахимовский проспект 36, к. 1

СОДЕРЖАНИЕ

I. ПРОЕКТИРОВАНИЕ И МОДЕЛИРОВАНИЕ СБИС

А.О. Балбеков, М.С. Горбунов Методики моделирования воздействия ТЗЧ на ИС в маршруте проектирования……4

Ю.В. Катунин, В.Я. Стенин Моделирование устойчивости КМОП элементов к помехам при воздействии одиночных частиц с использованием TCAD……...14

А.Ю. Богданов Отладка графической подсистемы СнК с использованием аппаратного комплекса для прототипирования Palladium Z1……21

II. МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ВИЗУАЛИЗАЦИЯ СИСТЕМ ВИРТУАЛЬНОГО ОКРУЖЕНИЯ

М.В. Михайлюк, Д.В. Омельченко, Д.А. Кононов, Д.М. Логинов Параметры камеры просмотра видео 360 градусов …….....26

Е.В. Страшнов, И.Н. Мироненко, Л.А. Финагин Командный режим управления виртуальным двуногим шагающим роботом.....33

III. ИНФОРМАЦИОННЫЕ И КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ

А.А. Рыбаков, А.Д. Чопорняк Повышение производительности векторного кода с помощью мониторинга плотности масок в векторных инструкциях…….40

Методики моделирования воздействия ТЗЧ на ИС в маршруте проектирования

А.О. Балбеков[1], М.С. Горбунов[2]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, balbekov@cs.niisi.ras.ru;

[2]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, gorbunov@cs.niisi.ras.ru

Аннотация. Одиночные сбои, вызванные воздействием заряженных частиц на интегральные схемы, представляют опасность для корректной работы бортовых систем космических аппаратов. Разработчики интегральных схем прилагают значительные усилия для защиты от одиночных сбоев. Эффективность принятых мер необходимо оценить в процессе разработки, прежде чем устройство будет подвергнуто длительной и дорогой процедуре испытаний. В данной работе представлен обзор методов оценки стойкости интегральных схем к одиночным сбоям при воздействии тяжелых заряженных частиц космического пространства. В статье разобраны современные достижения в области моделирования воздействия тяжелых заряженных частиц на интегральные схемы, реакции устройства на это воздействие и расчета параметров сбоеустойчивости. Представлен отечественный и зарубежный опыт.

Ключевые слова: ТЗЧ, сбоеустойчивость, частота сбоев (SER), одиночный переходный процесс (SET), одиночный сбой (SEU)

1. Введение

Одной из важных характеристик интегральных схем (ИС), предназначенных для применения в космосе, является устойчивость к радиации. На современном уровне развития техники это является одним из основных факторов, определяющих срок жизни космического аппарата.

Микросхемы, выполненные по суб-100 нм технологическим нормам, демонстрируют высокую чувствительность к одиночным эффектам, вызванным тяжелыми заряженными частицами (ТЗЧ). При движении ТЗЧ через материал полупроводникового устройства происходит генерация зарядов. Эти заряды вызывают различные эффекты, один из которых – импульс тока и напряжения на узлах электрической схемы. Устоявшимся термином для такой помехи является Single Event Transient (SET). SET в триггере или ячейке памяти может переключить хранимое состояние, что приведет к сбою – Single Event Upset (SEU). SET в комбинационной логике будет распространяться по схеме, если он совпадет с определенным состоянием тактового сигнала, то может быть зарегистрирован триггером или несколькими триггерами в разных частях схемы как действительный сигнал [1].

Перед использованием ИС в космических миссиях необходимо проверить уровень ее сбоеустойчивости. Наиболее точным методом является облучение образцов на ускорителях частиц. Само такое испытание дорогое и времязатратное. Если в результате испытаний в проект ИС придется вносить изменения, то все процедуры, предусмотренные маршрутом проектирования, и изготовление придется повторить. По этой причине развиваются методики моделирования, позволяющие оценить сбоеустойчивость ИС на этапе проектирования, а испытания на ускорителе проводить для финальной аттестации.

2. Аналитические подходы

Главным параметром сбоеустойчивости аппаратуры является частота сбоев на орбите. Частота сбоев ν вычисляется по формуле (1) [2]:

$$\nu = \int \sigma(L)\phi(L)dL, \qquad (1)$$

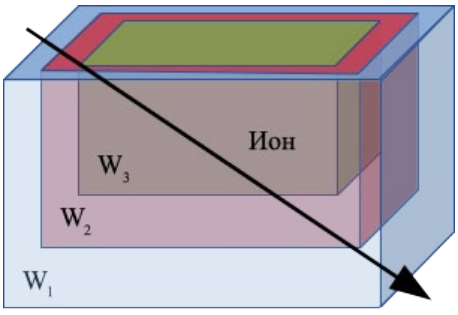
где σ – сечение сбоев, ϕ – дифференциальный поток частиц, L – ЛПЭ-спектр частиц. Развитием аналитического подхода являются методы, описанные в [3], [4], [5]. Для них требуется знать параметры аппроксимации экспериментальных данных сечения сбоев, размеры и расположение чувствительных объемов.

Поток частиц и ЛПЭ-спектр задаются параметрами миссии и программами-коллекциями моделей радиационной обстановки в околоземном пространстве: COSRAD [6], OMERE [7], CREME96 [8], SPENVIS [9]. Источниками данных о сечении сбоев являются результаты экспериментов по облучению ИС на ускорителях частиц или данные моделирования. Получение данных о сечении сбоев в моделировании основано на расчете количества энергии, которую частица теряет при прохождении через чувствительный объем. Если выделение энергии частицей привело к превышению критического значения, то регистрируется сбой. В качестве

критического параметра может выступать энергия, заряд, длительность, форма и амплитуда SET. Значение критического параметра может быть получено в SPICE или TCAD моделировании или из литературы.

В работах [10] и [11] описано применение методов Монте-Карло для определения сечения множественных сбоев. Существуют специальные программы для моделирования взаимодействия частиц с веществом, в их основе лежат методы Монте-Карло. В [12] для оценки частоты множественных сбоев используется программа Geant4 [13], что позволяет учесть влияние продуктов ядерных реакций в сложной структуре ИС, состоящей из нескольких слоев металла, диэлектрика и полупроводника. В [14] Geant4 и расчет критического заряда использовались для оценки стойкости к нейтронному облучению памяти, выполненной по технологическим нормам 65 нм и 45 ним.

Для тех же целей используется Geant4 в ПО MRED [15], [16], [17]. Образование сбоя определяется количеством заряда, который генерируется в чувствительном объеме при прохождении через него ТЗЧ. Если количество заряда превосходит критическое значение (вычисляется отдельным TCAD моделированием), то детектируется сбой. MRED трактует устройство, как набор вложенных чувствительных объемов (рис. 1). Каждому объему назначается свой коэффициент сбора, который определятся экспериментом или TCAD моделированием. Общий собранный заряд — сумма зарядов, собранных этими объемами. Благодаря использованию модифицированной библиотеки Geant4, MRED может моделировать воздействие радиации на микросхемы, выполненные по современным и перспективным технологиям. Управляющая оболочка MRED написана на языке Python и включает инструменты для интегрирования MRED с другими программами.



$$Q_{\text{сбор}} = \sum_i W_i Q_i$$

Рис. 1. Модель вложенных чувствительных объемов. W_i – весовые коэффициенты, $Q_{\text{сбор}}$ – полный собранный заряд, Q_i – заряд собранный в каждом вложенном объеме

3. Моделирование на уровне приборов

Самым точным способом моделирования сбора зарядов и реакции устройства на воздействия ТЗЧ является моделирование в технологических САПР (TCAD). Недостатком такого подхода является необходимость использовать большое количество технологических параметров, которые являются коммерческой тайной фабрик. Моделирование в TCAD отличается высокой ресурсоемкостью и ограничивает размер моделируемого устройства до нескольких десятков транзисторов. В частности, авторы [18] используют TCAD для оценки сбоеустойчивости предложенного ими варианта DICE ячейки при разных углах падения ТЗЧ.

Компания Cogenda предлагает набор программного обеспечения (ПО) для моделирования воздействия ТЗЧ на ИС: TCAD симулятор – VisualTCAD, ПО для создания точной модели топологии ИС – GDS2MESH, ПО для моделирование взаимодействия ТЗЧ с материалами ИС – VisualParticle [19]. Авторы [20] используют набор ПО Cogenda для оценки сбоеустойчивости предложенной ими DICE ячейки памяти, в [21] его используют для верификации предложенных стандартных ячеек.

Для вычисления реакции электрической схемы на воздействие ТЗЧ, ПО MRED можно совместить со SPICE симулятором [22]. Заряды, сгенерированные в разных вложенных чувствительных объемах, пересчитываются в параметры источников тока при помощи системы поправочных коэффициентов, которые калибруются по экспериментальным данным. Далее следует SPICE моделирование и детектирование сбоя. Карта чувствительных объемов устройства набирается из результатов множества TCAD моделирований. Расположение чувствительных объемов на карте будет зависеть от состояний на входах и выходах устройства и от динамики внутренних процессов (транзисторы будут открываться и закрываться). Таким образом, для моделирования устройства в динамике нужно будет построить по карте чувствительных объемов на каждый временной шаг моделирования.

В [23] [24] описано применение ПО iRoC TFIT [25]. Данное ПО поставляется вместе с базой импульсов токов, которая характеризует реакцию транзисторов, выполненных по определенной технологии, на воздействие нейтронов и альфа-частиц. Для использования TFIT с проектами, разработанными под разные технологии для, каждой должна быть использована соответствующая база, поставляемая авторами TFIT. База импульсов токов составляется сле-

дующим образом: в TCAD создаются модели транзисторов, на которые накладывается воздействие частиц. База импульсов токов в дальнейшем используется для серии SPICE моделирований, которая должна показать сбоеустойчивость проектируемого устройства.

В [26] и [27] описывается ПО MC-Oracle. На первом этапе рассчитывается взаимодействие протонов, нейтронов и ионов с веществом, для этого используются ПО SRIM [28] и DHORIN [29], в результате получаются треки первичных и вторичных частиц в веществе. На втором этапе рассчитывается распространение и сбор зарядов, которые были сгенерированы частицами. Для этого анализируется топология ИС в формате GDS, из нее извлекается расположение и форма чувствительных областей, что проиллюстрировано на рис. 2. Чувствительные области разбиваются на участки, для каждого из которых рассчитывается собранный заряд, из которого далее вычисляются параметры источника тока, который будет использован в SPICE моделировании. В результате второго этапа получается база данных, которая содержит в себе параметры всех частиц и связанных с ними источников тока. Далее следует серия SPICE моделирований, где источники тока подключаются к соответствующим транзисторам. Моделирование должно продемонстрировать реакцию устройства на воздействие частицы, в результате регистрируется наличие или отсутствие сбоя.

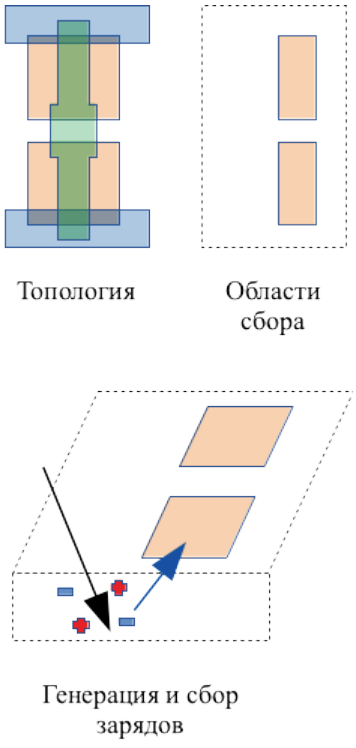


Рис. 2. Расчет сбора зарядов с учетом топологии ИС в MC-Oracle

По аналогичному принципу работает ПО MUSCA SEP3 [30] [31] [26]. По данным топологии из GDS файла строится объемная модель устройства. Прохождение частиц через вещество, ядерные реакции, образование и распространение осколков в материале устройства моделируется при помощи библиотеки Geant4. После расчета генерации, распределения и сбора зарядов составляется набор параметров для источников тока в SPICE моделировании, которое показывает наличие или отсутствие сбоя.

Методика SPICE моделирования воздействия одиночных частиц, учитывающая топологию устройства и позволяющая строить карты чувствительности, описана в [32], [33]. На топологию накладывается сетка, каждая прямоугольная область является областью воздействия. Источники тока будут подключены только к тем транзисторам, которые попали в область воздействия. Распределение зарядов по источникам тока характеризуется в TCAD. Анализ топологии устройства позволяет определить узлы, куда необходимо подключить источники тока. Распространение зарядов по объему и паразитный биполярный эффект не учитываются.

В [34] и [35] описываются аналогичные разработки. Посредством моделирований в TCAD создается таблица, в которой набор параметров импульсов тестового воздействия ставиться в соответствие расстоянию от места входа частицы. Далее соответствующее ПО определяет устройства, попавшие в область воздействия. К пораженным устройствам подключаются источники тока с параметрами, взятыми из таблицы. После проведения серии SPICE моделирований собирается статистика сбоев с разными параметрами воздействия. Контакты к телу транзисторов в [35] подключатся к земле или питанию, поэтому не учитываются эффекты, связанные с растеканием зарядов по подложке. В [34] контакты тела к телу транзисторов подключается к земле или питанию через резисторы. Способ генерации и подключения этих резисторов не описан, не ясно насколько подробно учитывается топология устройства. Обе работы [34] и [35] не моделируют эффект включения паразитного биполярного транзистора в объемных технологиях.

4. Моделирование на уровне электрической схемы

Моделирование электрической схемы устройства в SPICE симуляторах позволяет повысить скорость набора данных, за что приходится расплачиваться снижением точности. Традиционным способом моделирования воз-

действия ТЗЧ на ИС является подключение источников тока с импульсом специальной формы к транзисторам в SPICE моделировании. Форма импульса тока может быть задана кусочно-линейной аппроксимацией, динамической Verilog-A моделью или таблицей импульсов, построенной на основе множества TCAD моделирований [36]. Авторы [37] для анализа сбоеустойчивости мажорирующих ячеек используют динамический Verilog-A источник тока, включенный в модель транзистора. Самым популярным способом задания формы импульса тока является «двойная экспонента» (2) [38]:

$$I(t) = \frac{Q}{\tau_r - \tau_f}\left(e^{-\frac{t}{\tau_r}} - e^{-\frac{t}{\tau_f}}\right), \qquad (2)$$

где Q – собранный заряд, τ_r – постоянная времени нарастания импульса, τ_f – постоянная времени спада импульса, t – время. Если источник тока будет подключен к закрытому транзистору, то на его узлах можно будет наблюдать SET, который будет распространяться по электрической схеме, и может вызвать сбой. Параметры τ_r и τ_f являются характеристиками технологии и могут быть рассчитаны аналитически или получены в TCAD моделировании. Значение Q может быть рассчитано из значения ЛПЭ частицы или получено при помощи Geant4. На рис. 3 и 4 показано моделирование SEU в ячейке памяти при помощи источника тока специальной формы.

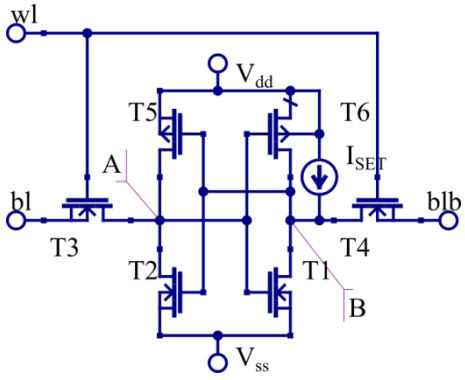


Рис. 3. Использование источника тока с импульсом специальной формы для моделирования воздействия ТЗЧ в SPICE моделировании

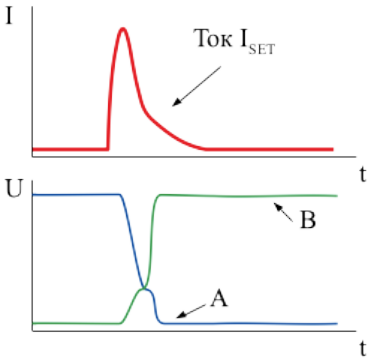


Рис. 4. Переходные процессы на узлах A и B, вызванные воздействием источника тока I_{SET} (рис. 3)

Авторы [39] используют подключение источника тока в электрическую схему для исследования сбоеустойчивости DICE ячейки памяти. DICE ячейка защищена от сбоя, если воздействие подается только на один транзистор. Авторы проанализировали топологию ячейки и определили несколько чувствительных областей, источники тока подключали к нескольким транзисторам, располагающимся в одной чувствительной области.

Для оценки сбоеустойчивости больших схем нужно собрать статистику воздействия ТЗЧ на разные транзисторы с разными параметрами, много работ посвящено автоматизации этого процесса. Автоматизация может использоваться для последовательного перебора транзисторов и параметров источников тока. В частности, в работе [40] исследуется сбоеустойчивость тракта чтения регистрового файла, а в работе [41] – частота сбоев ячеек из стандартной библиотеки элементов. В [42] описана система SPICE моделирования воздействия частицы, которая случайным образом выбирает цель для подключения источника тока, для учета множественных сбоев может быть выбрано одновременно несколько целей.

Описанные выше системы используют в качестве входных данных электрическую схему устройства в формате SPICE. Задание нескольких целей для подключения источников тока возможно или вручную, или случайным выбором, что ограничивает учет влияния топологии на сбоеустойчивость. Эта проблема решена в работах [43] и [44], где описан автоматизированный анализ топологии в формате GDS для определения целей. На топологию накладывается окружность, определяющая область воздействия (область диффузии зарядов, сгенерированных ТЗЧ). Источники тока подключаются к транзисторам, которые попали в эту окружность.

5. Моделирование на уровне регистровых передач

Производители программируемых логических интегральных схем (ПЛИС) встраивают в некоторые свои продукты возможность «прицельного» изменения состояния конфигурационной памяти, что может быть использовано для инжекции сбоев [45]. Компания Xilinx предоставляет коммерческое ПО для внесения и коррекции сбоев в конфигурационной памяти ПЛИС их производства [46]. В [47] средства внесения сбоев в ПЛИС использовано для анализа методов частичного троирования. В [48] рассматриваются влияние сбоев в памяти, отвечающей за разводку, на целостность и работу прошивки ПЛИС. Все методы внесения сбоев, основанные на ПЛИС, пригодны только для полностью цифровых устройств. Разработчики ИС работают с разнообразными заказными и цифро-аналоговыми блоками, их эмуляция в ПЛИС зачастую невозможна. Характеристики скорости работы и емкости ПЛИС ограничивают параметры исследуемого устройства и возможности по внесению сбоев. При переносе проекта из ПЛИС в ИС САПР СБИС изменит электрическую схему. С целью выполнения заданных требований по задержкам будет добавлено множество буферов. На стадии ПЛИС невозможно предсказать количество и размещение этих буферов и невозможно определить их влияние на сбоеустойчивость ИС.

Уровень регистровых передач – один из уровней абстракции в маршруте разработки ИС. Внесение сбоев на этом уровне позволяет оценить сбоеустойчивость больших цифровых блоков. Разработка методов внесения сбоев на уровне регистровых передач ведется с тех пор, как стали доступны достаточно развитые симуляторы [49]. Один из методов заключается в следующем: необходимо модифицировать схему при помощи специальной программы, чтобы внедрить в соединения между вентилями специальные модули. Контроллер теста будет активировать эти модули в определенное время, запуская SET или SEU. В [50] такой подход применяется для поиска уязвимых частей в цифровой ФАПЧ и для проверки способности системы к восстановлению после сбоя. Авторы [51] и [52] предлагают модифицировать библиотеку моделей стандартных ячеек, добавив дополнительные порты управления, которые будут запускать SET и SEU в моделировании. Такой подход требует от поставщиков библиотек стандартных ячеек поставлять разработчикам специальные версии библиотек. Третьим подходом является использование встроенных возможностей симуляторов для изменения состояния выбранных сигналов в определенный момент времени в моделировании. Авторы [53] разработали инжектор сбоев, способный менять состояние триггеров в определенный момент моделирования. В [54] SystemC библиотека Universal Verification Methodology (UVM) была адаптирована для верификации с внесением сбоев блоков шифрования по алгоритмам AES и CRC. Аналогичный подход на базе VHDL описан в [55].

Методы внесения сбоев на уровне регистровых передач получают развития благодаря внедрению алгоритмов анализа топологии, что позволяет моделировать множественные сбои. Топология ИС может быть сохранена в формате DEF. В [56], [55], [57], [58] он используется для поиска ячеек, куда будет инжектирован сбой. В общем случае ПО для анализа топологии определяет, какие именно ячейки попали под воздействие частицы, и на некоторое время инвертирует состояние их выходов [58], метод проиллюстрирован на рис. 5.

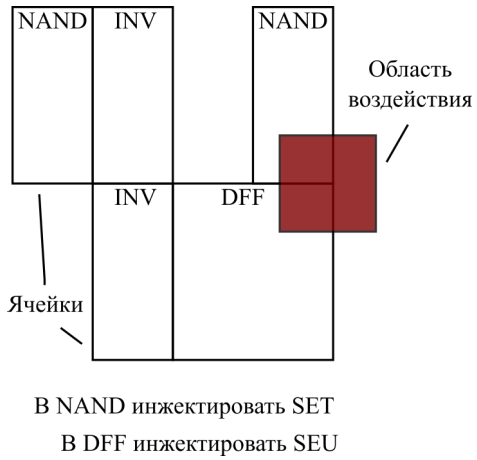


Рис. 5. Инжекция сбоев в RTL с учетом топологии

Помимо анализа топологии, авторы [59] и [56] используют TCAD для создания базы данных зависимости формы и длительности SET от координат попадания частицы в ячейки. Комбинационная схема набирается из стандартных ячеек, для учета влияния отдельной стандартной ячейки на уязвимость схемы, каждая ячейка отдельно характеризуется. Процедура характеризации заключается в наложении на ячейку прямоугольной сетки. При помощи TCAD проводится моделирование внесения зарядов в каждый из узлов сетки, специальное ПО собирает информацию о формах SET, таблица параметров SET является целью характеризации. Далее на топологию устройства накладывается область воздействия, определяется, какие узлы сетки каких ячеек в нее попали, в соответствующие узлы инжектируются SET. Аналогичный подход используется в [60]. Библиотека стан-

дартных ячеек содержит в себе от нескольких десятков до нескольких сотен ячеек. Характеризация каждой ячейки посредством множества моделирований в TCAD может занять много времени.

Инжекция сбоев с учетом топологии хорошо подходит для проверки эффективности геометрических методов повышения сбоеустойчивости. Если стандартные ячейки малы и расположены плотными группами, то множественные сбои будут появляться часто. В [55] инжекция сбоев с учетом топологии применяется для анализа метода топологической группировки ячеек, логически связанных друг с другом определенным образом. В момент появления множественного сбоя переходные процессы на выходах этих ячеек должны погасить друг друга. Инжектор определяет, куда именно попала частица: в n-карман или в p-подложку, на основании чего она решает, какое состояние должно быть включено на выходах ячейки.

В [57] место инжекции сбоев вычисляется исходя из координаты попадания частицы, ее ЛПЭ и расстояния между контактами к карману/подложке, длительность SET зависит от ЛПЭ. Моделирование ядра процессора Leon3 показало значительную переоценку числа сбоев относительно эксперимента. Авторы ввели поправку – вероятность инжекции: случайным образом определяется, будет ли проведена инжекция в данном моделировании. Вероятность инжекции рассчитывается как отношение экспериментального сечения сбоев к площади устройства.

6. Моделирование на уровне системы

Инжекция сбоев в память процессора описана в [61], [62]. Использовалась модель процессора, запрограммированная в ПЛИС, где был запущен инжектор, способный вносить сбои во внутреннюю и внешнюю память процессора. Инжектор управляется внешним программно-аппаратным комплексом, для внесения сбоев во внутреннюю память процессора требуется его остановка, внесение сбоев во внешнюю память остановки не требует. Аналогичный подход описан в [63] и [64], отличие в том, что инжектор управляется самим исследуемым процессором.

Инжекция сбоев на системном уровне используется для отладки программ. В [65] упоминается наличие в сбоеустойчивом микроконтроллере программного механизма внесения ошибок, предназначенного для отладки подсистемы диагностики сбоев. Инжектор сбоев может быть встроен в комплекс программного моделирования аппаратного обеспечения [66]. Комплекс позволяет разрабатывать программы для процессоров при отсутствии самих процессоров в «железе», т.е. для программной модели процессора. Позволяет внедрять сбои на системном уровне: в процессоры, шины, интерфейсы.

7. Заключение

Моделирование воздействия ТЗЧ на ИС позволяет во время разработки устройства оценить вклад различных мер повышения сбоеустойчивости, а дорогостоящие испытания проводить для финальной сертификации. Моделировать можно на разных этапах маршрута проектирования. Существует баланс между точностью и скоростью моделирования.

Аналитические подходы представляют собой оценку частоты сбоев по аппроксимированным данным испытаний аналогичных устройств. Аналитический подход может быть дополнен моделированием ядерных реакций в ИС при помощи библиотек типа Geant4.

Самые точные методики основываются на моделировании физических процессов в полупроводниковых приборах при помощи TCAD. Эти методики наиболее медленные и требуют использования технологических параметров, являющихся секретом фабрики. В более быстрых и менее точных подходах TCAD совмещается с SPICE симулятором, где TCAD моделирует только часть устройства.

Методы на основе моделирования электрических цепей в SPICE симуляторе занимают место в середине спектра точность-скорость и способны работать со схемами из сотен транзисторов. Воздействие ТЗЧ моделируется подключением импульсных источников тока к определенным узлам схемы. Точность этих методов может быть увеличена, если ввести в моделирование учет топологии устройства. Для работы этих методов требуется гораздо меньше технологических параметров.

Методы моделирования на уровне регистровых передач практически оторваны от технологии производства ИС и физики происходящих процессов. В основе этих методов лежит изменение логического состояния определенных ячеек в нужный момент времени. Позволяют быстро моделировать схемы из тысяч стандартных ячеек, могут быть дополнены учетом топологии.

Разработчики сложных интегральных схем, таких как процессоры, встраивают в свои устройства (или поставляют отдельно) средства отладки программ, позволяющие менять внутренние состояния различных регистров. Эти

средства отладки могут быть использованы для внесения сбоев на системном уровне, что позволит оценить корректность работы ПО при наличии сбоев от ТЗЧ.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований по теме «Архитектурные и схемотехнические методы снижения энергопотребления и повышения сбоеустойчивости микропроцессоров и коммуникационных контроллеров высокопроизводительных ЭВМ» (№ 0065-2019-0008).

A Design Flow Compatible Simulation Techniques for the Heavy Ion Impact on the Integrated Circuit

A.O. Balbekov, M.S. Gorbunov

Abstract. Single event upsets (SEU) caused by the heavy-ion impact on an integrated circuit (IC) pose a danger for the correct functioning of satellite hardware. IC designers put much effort into SEU mitigation. It is critical to evaluate the effectiveness of the mitigation measures in the design stage, before the cost- and time-consuming procedure of testing in the irradiation facility. This work presents a review of the estimation techniques for fault tolerance of IC for heavy ions in the space environment. The paper discusses modern achievements in the field of simulation of the impact of heavy-ions on IC, device reaction to the impact, and fault tolerance analysis.

Keywords: heavy ion, fault tolerance, soft error rate (SER), single event transient (SET), single event upset (SEU)

Литература

1. P. Shivakumar, M. Kistler, S.W. Keckler, D. Burger, L. Alvisi. Modeling the effect of technology trends on the soft error rate of combinational logic. «Proceedings International Conference on Dependable Systems and Networks2, USA, DC, Washington, 23 – 26 June 2002, IEEE, 389 – 398.

2 А.И. Чумаков. Действие космической радиации на интегральные схемы. М., Радио и связь, 2004.

3. E.L. Petersen, J.C. Pickel, E.C. Smith, P.J. Rudeck, J.R. Letaw. Geometrical factors in SEE rate calculations. «IEEE Transactions on Nuclear Science», V. 40 (1993), № 6, 1888 – 1909.

4. J.N. Bradford. Geometric Analysis of Soft Errors and Oxide Damage Produced by Heavy Cosmic Rays and Alpha Particles. «IEEE Transactions on Nuclear Science», V. 27 (1980), № 1, 941 – 947.

5. А.И. Чумаков. Оценка многократных сбоев в интегральных схемах от воздействия тяжелых зараженных частиц. «Микроэлектроника», Т. 43 (2014), № 2, 83 – 87.

6. COSRAD. http://smdc.sinp.msu.ru/index.py?nav=model-cosrad

7. OMERE. https://www.trad.fr/en/space/omere-software/

8. CREME96. https://creme.isde.vanderbilt.edu/

9. SPENVIS. https://www.spenvis.oma.be/

10. G.I. Zebrev, K.S. Zemtsov. Multiple cell upset cross-section modeling: A possible interpretation for the role of the ion energy-loss straggling and Auger recombination. «Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment», V. 827 (2016), 1 – 7.

11. А.В. Согоян, А.И. Чумаков, А.А. Смолин. Оценка частоты одиночных радиационных эффектов для современных СБИС. «Проблемы разработки перспективных микро- и наноэлектронных систем», Т. 4 (2018), 170 – 176.

12. A.M. Galimov, R.M. Galimova, G.I. Zebrev. GEANT4 simulation of nuclear interaction induced soft errors in digital nanoscale electronics: Interrelation between proton and heavy ion impacts. «Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment», V. 913 (2019), 65 – 71.

13. Geant4. http://geant4.cern.ch/

14. И.В. Елушов. Оценка стойкости элементов СОЗУ к одиночным сбоям при воздействии высокоэнергитичных нейтронов. «Технологии электромагнитной совместимости» (2019), № 2(69), 6 – 12.

15. K.M. Warren, B.D. Sierawski, R.A. Reed, R.A. Weller, C. Carmichael, A. Lesea, M.H. Mendenhall, P.E. Dodd, R.D. Schrimpf, L.W. Massengill, T. Hoang, H. Wan, J.L. De Jong, R. Padovani, J.J. Fabula. Monte-Carlo Based On-Orbit Single Event Upset Rate Prediction for a Radiation Hardened by Design Latch. «IEEE Transactions on Nuclear Science», V. 54 (2007), № 6, 2419 – 2425.

16. R.A. Reed, R.A. Weller, M.H. Mendenhall, D.M. Fleetwood, K.M. Warren, B.D. Sierawski, M.P. King, R.D. Schrimpf, E.C. Auden. Physical Processes and Applications of the Monte Carlo Radiative Energy Deposition (MRED) Code. «IEEE Transactions on Nuclear Science», V. 62 (2015), № 4, 1441 – 1461.

17. J.D. Black, J.A. Dame, D.A. Black, P.E. Dodd, M.R. Shaneyfelt, J. Teifel, J.G. Salas, R. Steinbach, M. Davis, R.A. Reed, R.A. Weller, J.M. Trippe, K.M. Warren, A.M. Tonigan, R.D. Schrimpf, R.S. Marquez. Using MRED to Screen Multiple-Node Charge-Collection Mitigated SOI Layouts. «IEEE Transactions on Nuclear Science», V. 66 (2019), № 1, 233 – 239.

18. Ю.В. Катунин, В.Я. Стенин. TCAD моделирование эффектов воздействия одиночных ядерных частиц на ячейки памяти STG DICE. «Микроэлектроника», V. 47 (2018), № 1, 23 – 37.

19. S. Chen, G. Ding, Z. Li-Sang, J. Dong-Mei. A framework for fully-physical, statistically-enhanced Monte-Carlo simulation of SEU. «Proceedings of RADECS 2012», France, Biarritz, 24 – 28 Sep 2012, IEEE.

20. M. Lavania, N. Surana, I. Anand, J. Mekie. Read-Decoupled Radiation Hardened RD-DICE SRAM Cell for Low-Power Space Applications. «2019 IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC)», China, Xi'an, 12 – 14 June 2019, IEEE.

21. R. Trivedi, N.M. Devashrayee, U.S Mehta, N.M. Desai, Himanshu Patel. Development of Radiation Hardened by Design(RHBD) primitive gates using 0.18µm CMOS technology. «2015 19th International Symposium on VLSI Design and Test», India, Ahmedabad, 26 – 29 June 2015, IEEE.

22. K.M. Warren, A.L. Sternberg, R.A. Weller, M.P. Baze, L.W. Massengill, R.A. Reed, M. H. Mendenhall, R.D. Schrimpf. Integrating Circuit Level Simulation and Monte-Carlo Radiation Transport Code for Single Event Upset Analysis in SEU Hardened Circuitry. «IEEE Transactions on Nuclear Science», V. 55 (2008), № 6, 2886 – 2894.

23. J. Noh, V. Correas, S. Lee, J. Jeon, I. Nofal, J. Cerba, H. Belhaddad, D. Alexandrescu, Y. Lee, S. Kwon. Study of Neutron Soft Error Rate (SER) Sensitivity: Investigation of Upset Mechanisms by Comparative Simulation of FinFET and Planar MOSFET SRAMs. «IEEE Transactions on Nuclear Science», V. 62 (2015), № 4, 1642 – 1649.

24. S. Yoshimoto, T. Amashita, D. Kozuwa, T. Takata, M. Yoshimura, Y. Matsunaga, H. Yasuura, H. Kawaguchi, M. Yoshimoto. Multiple-bit-upset and single-bit-upset resilient 8T SRAM bitcell layout with divided wordline structure. «2011 IEEE 17th International On-Line Testing Symposium», Greece, Athens, 13 – 15 July 2011, IEEE.

25. TFIT™: Cell Level Soft Error Analysis. https://www.iroctech.com/solutions/transistorcell-level-fault-simulation-tools-and-services/

26. Y.Q. Aguiar, F. Wrobel, J.-L. Autran, P. Leroux, F. Saigné, V. Pouget, A. D. Touboul. Mitigation and Predictive Assessment of SET Immunity of Digital Logic Circuits for Space Missions. «Aerospace», V. 7 (2020), № 2, 12.

27. F. Wrobel, F. Saigné. MC-ORACLE: A tool for predicting Soft Error Rate. «Computer Physics Communications», V. 182 (2011), 317 – 321.

28. J.F. Ziegler, M.D. Ziegler, J.P. Biersack. SRIM – The stopping and range of ions in matter (2010). «Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms», V. 268 (2010), № 11 – 12, 1818 – 1823.

29. F. Wrobel. Detailed history of recoiling ions induced by nucleons. «Computer Physics Communications», V. 178 (2008), № 2, 88 – 104.

30. G. Hubert, S. Duzellier, C. Inguimbert, C. Boatella-Polo, F. Bezerra, R. Ecoffet. Operational SER Calculations on the SAC-C Orbit Using the Multi-Scales Single Event Phenomena Predictive Platform (MUSCA SEP^3). «IEEE Transactions on Nuclear Science», V. 56 (2009), № 6, 3032 – 3042.

31. G. Hubert, L. Artola. Single-Event Transient Modeling in a 65-nm Bulk CMOS Technology Based on Multi-Physical Approach and Electrical Simulations. «IEEE Transactions on Nuclear Science», V. 60 (2013), № 6, 4421 – 4429.

32. E. Do, V. Liberali, A. Stabile, C. Calligaro. Layout-oriented simulation of non-destructive single event effects in CMOS IC blocks. «2009 European Conference on Radiation and Its Effects on Components and Systems», Belgium, Brugge, 14 – 18 Sep 2009, 217 – 224.

33. G. Bozzola, L. Frontini, V. Liberali, S.R. Shojaii, A. Stabile. Improvement of radiation tolerance in CMOS ICs through layout-oriented simulation. «2016 5th International Conference on Modern Circuits

and Systems Technologies (MOCAST)», Geece, Thessaloniki, 12 – 14 May 2016, 1 – 4.

34. J.S. Kauppila, T.D. Haeffner, D.R. Ball, A.V. Kauppila, T.D. Loveless, S. Jagannathan, A.L. Sternberg, B.L. Bhuva, L.W. Massengill. Circuit-Level Layout-Aware Single-Event Sensitive-Area Analysis of 40-nm Bulk CMOS Flip-Flops Using Compact Modeling. «IEEE Transactions on Nuclear Science», V. 58 (2011), № 6, 2680 – 2686.

35. A.M. Francis, D. Dimitrov, J. Kauppila, A. Sternberg, M. Alles, J. Holmes, H.A. Mantooth. Significance of Strike Model in Circuit-Level Prediction of Charge Sharing Upsets. «IEEE Transactions on Nuclear Science», V. 56 (2009), № 6, 3109 – 3114.

36. А.А. Смолин, А.Б. Боруздина, А.В Уланова, А.В. Яненко, А.В. Согоян, А.Ю. Никифоров, В.А. Телец, А.И. Чумаков, Н.А. Шелепин. Схемотехническое моделирование одиночных эффектов при воздействии тяжелых заряженных частиц в КМОП СБИС с суб-200-ни проектными нормами. «Известия высших учебных заведений. Электроника», V. 22 (2017), № 5, 447 – 459.

37. I.A. Danilov, M.S. Gorbunov, A.A. Antonov. SET Tolerance of 65 nm CMOS Majority Voters: A Comparative Study. «IEEE Transactions on Nuclear Science», V. 61 (2014), № 4, 1597 – 1602.

38. G.C. Messenger. Collection of Charge on Junction Nodes from Ion Tracks. «IEEE Transactions on Nuclear Science», V. 29 (1982), № 6, 2024 – 2031.

39. В.Я. Стенин, И.Г. Черкасов. Влияние топологии субмикронных КМОП ячеек памяти DICE на чувствительность ОЗУ к воздействию отдельных ядерных частиц. «Микроэлектроника», V. 40 (2011), № 3, 184 – 190.

40. А.О. Балбеков. Моделирование переходных процессов в системе чтения регистровых файлов вследствие одиночных событий. «Вопросы атомной науки и техники. Серия: физика радиационного воздействия на радиоэлектронную аппаратуру» (2015), № 4, 11 – 19.

41. Д.И. Тельпухов, А.И. Деменева, В.В. Надоленко. Исследоваие и разработка автоматизированных средств моделирования случайных сбоев в современных комбинационных КМОП ИМС. «Известия ЮФУ. Технические науки» (2019), № 4 (206), 207 – 219.

42. V.Sh. Melikyan, A. Petrosyan, A.Kh. Mkhitaryan, A.K. Hayrapetyan, Z. Avetisyan, A.E. Mkrtchyan. The Single Event Upset Forecasting in Digital and Analog Integrated Circuits in SAED 14nm FinFet Technology. «Problems of advanced micro- and nanoelectronic systems development» (2018), № 4, 76 – 81.

43. G.I. Paliaroutis, P.T., N. Evmorfopoulos, G. Dimitriou, G.I. Stamoulis. SET Pulse Characterization and SER Estimation in Combinational Logic with Placement and Multiple Transient Faults Considerations. «Technologies», V. 8 (2020), № 1, 5.

44. A. Balbekov, M. Gorbunov, S. Bobkov. Layout-aware simulation of soft errors in sub-100 nm integrated circuits. «International Conference on Micro- and Nano-Electronics 2016», Russian Federation, Zvenigorod, 3 – 7 Oct 2016, SPIE, 305 – 310.

45. R. Zhang, L. Xiao, J. Li, X. Cao, C. Qi. A Fault Injection Platform Supporting Both SEU and Multiple SEUs for SRAM-Based FPGA. «IEEE Transactions on Device and Materials Reliability», V. 18 (2018), № 4, 599 – 605.

46. Soft Error Mitigation Controller v4.1. 4 Apr 2018. https://www.xilinx.com/support/documentation/ip_documentation/sem/v4_1/pg036_sem.pdf.

47. A. Ramos, R.G. Toral, P. Reviriego, J.A. Maestro. An ALU Protection Methodology for Soft Processors on SRAM-Based FPGAs. «IEEE Transactions on Computers», V. 68 (2019), № 9, 1404 – 1410.

48. L. Bozzoli, C. De Sio, L. Sterpone, C. Bernardeschi. PyXEL: An Integrated Environment for the Analysis of Fault Effects in SRAM-Based FPGA Routing. «2018 International Symposium on Rapid System Prototyping (RSP)», Italy, Torino, 4 – 5 Oct 2018, IEEE, 70 – 75.

49. F. Vargas, A. Amory, R. Velazco. Estimating circuit fault-tolerance by means of transient-fault injection in VHDL. «Proceedings 6th IEEE International On-Line Testing Workshop (Cat. No.PR00646)», Spain, Palma de Mallorca, 3 – 5 June 2000, IEEE, 67– 72.

50. W. Yang, C. He, X. Du, Y. Fan, Y. Yuan. Design and simulation of SET and radiation-harden on DPLL. «2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)», China, Xi'an, 3– 5 Oct 2016, IEEE, 1457 – 1461.

51. Д.Е. Матафонов. Способы инжекции сбоев в ПЛИС в рамках разработки методов повышения сбоеустойчивости. «Сборник трудов XIII Всероссийского совещания по проблемам управления ВСПУ-2019. Институт проблем управления им. В.А. Трапезникова РАН. 2019», 17 – 20 Мая 2019, Институт проблем управления им. В. А. Трапезникова РАН, 2422 – 2427.

52. A. Mochizuki, N. Onizawa, A. Tamakoshi, T. Hanyu. Multiple-event-transient soft-error gate-level simulator for harsh radiation environments. «TENCON 2015 – 2015 IEEE Region 10 Conference», China, Macao, 1 – 4 Nov 2015, IEEE, 1 – 6.

53. П.Н. Осипенко, А.А. Антонов, С.А. Левадский. Исследование архитектурной чувствительности к сбоям с использованием метода статического внесения сбоев. «Программные продукты и ситемы» (2010), № 4, 12 – 15.

54. D. Lohmann, F. Maziero, E. J. dos Santos, D. Lettnin. Extending universal verification methodology with fault injection capabilities. «2018 IEEE 9th Latin American Symposium on Circuits Systems (LASCAS)», Mexico, Puerto Vallarta, 25 – 28 Feb 2018, IEEE, 1 – 4.

55. B.T. Kiddie, W.H. Robinson. Alternative Standard Cell Placement Strategies for Single-Event Multiple-Transient Mitigation. «2014 IEEE Computer Society Annual Symposium on VLSI», USA, FL, Tampa, 9 – 11 July 2014, IEEE, 589 – 594.

56. Y. Du, S. Chen. A Novel Layout-Based Single Event Transient Injection Approach to Evaluate the Soft Error Rate of Large Combinational Circuits in Complimentary Metal-Oxide-Semiconductor Bulk Technology. «IEEE Transactions on Reliability», V. 65 (2016), № 1, 248 – 255.

57. C. Bottoni, B. Coeffic, J.-M. Daveau, G. G., L. Naviner, P. Roche. A layout-aware approach to fault injection for improving failure mode prediction. «Proceedings of Workshop on Silicon Errors in Logic-System Effects» USA, TX, Austin, 30 Mar – 1 Apr 2015, IEEE, 2015.

58. I.A. Danilov, A.I. Shnaider-Khazanova, A.O. Balbekov, M.S. Gorbunov. Standard Verification Flow Compatible 3-Aware Fault Injection 1nique for Single Event Effects Tolerant ASIC Design. «European Conference on Radiation and its Effects on Components and Systems», France, Montpellier, 16 – 20 Sep 2019.

59. S. Chen, Y. Du, B. Liu, J. Qin. Calculating the Soft Error Vulnerabilities of Combinational Circuits by Re-Considering the Sensitive Area. «IEEE Transactions on Nuclear Science», V. 61 (2014), № 1, 646 – 653.

60. J. Li, J. Draper. Joint Soft-Error-Rate (SER) Estimation for Combinational Logic and Sequential Elements. «2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)», USA, PA, Pittsburgh, 11 – 13 July 2016, IEEE, 737 – 742.

61. Е.С. Лепешкина, С.А. Чекмарев. Метод внутрикристального инъектирования сбоев в реальном времени для тестирования сбоеустойчивых микропроцессоров типа система-на-кристалле. ««Орбита молодежи» и перспективы развития российской космонавтики сборник докладов Всероссийской молодежной научно-практической конференции», 18 – 22 Сентября 2014, Национальный исследовательский Томский политехнический университет, 113 – 114.

62. С.А. Чекмарев, В.Х. Ханов, А.С. Тимохович. Технология инъектирования сбоев для тестирования сбоекстойчивости микропроцессоров, предназначеных к использованию в бортовой аппаратуре космических аппаратов. «Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева», V. 17 (2016), № 3, 768 – 781.

63. О.В. Мамутова, О.В. Ненашев, А.С. Филлипов. Оснащение систем на кристалле средствами эмуляции сбоев в памяти. «Известия высших учебных заведений. Электроника», V. 20 (2015), № 1, 50 – 57.

64. О.В. Мамутова. Оценка надежности при одиночных сбоях в кэш-памяти в маршруте проектирования системы на кристалле. «Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС)» (2016), № 4, 166 – 171.

65. С. Шумилин, М. Какоулин. Сбоеустойчивый микроконтроллер на базе ядра ARM Cortex-M4F для систем с повышенными требованиями надежности, разработанный ЗАО «ПКК Миландр». «Компоненты и технологии» (2013), № 12 (149), 90 – 92.

66. J. Engblom, D. Ekblom. SIMICS: A COMMERCIALLY PROVEN FULL-SYSTEM SIMULATION FRAMEWORK. 2006. http://www.engbloms.se/publications/engblom-sesp2006.pdf.

Моделирование устойчивости КМОП элементов к помехам при воздействии одиночных частиц с использованием TCAD

Ю.В. Катунин[1], В.Я. Стенин[1, 2]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, e-mail: katunin@cs.niisi.ras.ru;

[2]НИЯУ МИФИ, Москва, Россия, e-mail: vystenin@mephi.ru

Аннотация. Представлены результаты моделирования формирования импульсов помех логическими элементами при сборе заряда с треков одиночных частиц в широком диапазоне линейного переноса энергии частицей на трек 10–90 МэВ·см^2/мг. Исследование выполнено с использованием 3D TCAD физических моделей КМОП транзисторов объемной технологии 65 нм с неглубокой траншейной изоляцией транзисторных групп. Главное, что происходит при сборе заряда с трека, проходящего через группу NMOП транзисторов элемента ИЛИ, а также в случае через группу PMOП транзисторов элемента И, заключается в удержании этих транзисторов группы в режиме, когда все транзисторы одновременно собирают заряды, в частности находясь в инверсном смещении. Это формирует задержку импульсов помех, образующихся на выходах элементов, что и уменьшает длительность импульсов помех. При этом длительность импульсов помехи снижается в среднем до уровня 250–350 пс для точек трека в группу NMOП транзисторов элемента ИЛИ и до уровня 140–150 пс для точек трека в группу PMOП транзисторов элемента И.

Ключевые слова: импульс помехи, логический элемент, моделирование, мажоритарный элемент, одиночная ядерная частица, топология, трек частицы, сбор заряда

1. Введение

Моделирование средствами 3-D TCAD с использованием физических моделей образования носителей заряда на треке частицы является, по сути, виртуальной экспериментальной базой получения данных о поведении нано-размерных элементов при воздействии одиночных частиц. Таким моделированием предсказано [1] снижение помехоустойчивости КМОП логики по объемной технологии при проектной норме менее 100 нм, а также переход NMOП транзисторов в инверсный режим смещения [2] с увеличением длительности помех до 300–500 пс при 30 МэВ·см^2/мг. Установлено, что совместный сбор заряда с трека [3] транзисторами элемента может привести к уменьшению длительности импульсов помех.

В результате TCAD моделирования [4] установлено, что при треках с линейной передачей энергии частицей на них (linear energy transfer – LET) 60 МэВ·см^2/мг помехи с наибольшими длительностями образуются в группе NMOП транзисторов элемента ИЛИ и группе PMOП транзисторов элемента И. В этих группах большая часть заряда с трека собирается транзисторами, выполненными с общей областью стоков. В данной работе приводятся результаты анализа параметров импульсов помех, образующихся в тех же группах транзисторов, но в широком диапазоне LET = 10–90 МэВ·см^2/мг.

Целью работы является изучение возможной минимизации импульсов помех, образующихся при сборе заряда с трека одиночной ядерной частицы в элементах в составе мажоритарного элемента. Это сбор заряда NMOП транзисторами элемента ИЛИ, содержащего элемент И-НЕ и инвертор, а также PMOП транзисторами элемента И, содержащего элемент И-НЕ и инвертор, которые конструктивно выполняются в ограниченном объеме кремния, окруженном неглубокой траншейной изоляцией.

На рис. 1 приведена схема тройного мажоритарного элемента на логических КМОП элементах И (D1-D3) и ИЛИ (D4). Элементы И (D1) и ИЛИ (D4) на рис. 1 изображены в виде электрических схем, а D2 и D3 в виде функциональных условных обозначений. Схема элемента D1 включает в себя элемент И-НЕ и инвертор. Схема элемента D4 включает в себя элемент ИЛИ-НЕ и инвертор.

NMOП транзисторы элементов И и ИЛИ выполнены как группы Gr1N и Gr4N. Аналогично выполнены группы Gr1P и Gr4P и PMOП транзисторы элементов И и ИЛИ. Все эти группы ограничены штриховыми линиями на рис. 1.

2. Моделирования импульсов помех средствами TCAD

Воздействие на МОП элементы СБИС одиночной ядерной частицы приводит к образова-

нию вдоль её трека неравновесных носителей заряда. Заряды выводятся в виде импульсов тока через обратно смещенные стоковые *pn* переходы МОП транзисторов, вызывая импульсы помех, которые могут приводить к образованию ложных выходных сигналов элементов, искажающих логические уровни на выходе комбинационной логики.

На рис. 2 изображен эскиз части 3-D физической модели приборной структуры мажоритарного элемента. Модель включает КМОП транзисторы элементов И и ИЛИ. Транзисторы выполнены по объемной 65-нм технологии на основе моделей по методике, представленной в работе [5] с подбором технологических параметров соответствия моделям, поставляемым фабрикой.

Кремниевые области групп Gr1N и Gr1P элемента И имеют размеры 885×400×400 нм^3, а групп Gr4N и Gr4P элемента ИЛИ - 1180×800×400 нм^3. Ширина транзисторов элемента И составляет 400 нм, а у элемента ИЛИ 800 нм.

В качестве тестовых воздействий на ячейку памяти STG DICE в работе были использованы воздействия зарядом с треков частиц с разными линейными потерями энергии. При моделировании использованы треки типа T1P и T4N, проходящие через области стоков транзисторов в группах Gr1P и Gr1N. Энергетическая составляющая воздействия одиночной ядерной частицы характеризуется линейной передачей энергии на трек (linear energy transfer – LET) [6].

Для наглядности структуры приборной части модели из рис. 2 убрано изображение областей разделительного мелкого слоя оксида толщиной 400 нм, охватывающего кремниевые области транзисторов групп Gr1N, Gr1P и Gr4N, Gr4P в реальной конструкции. Области с обозначениями *n*+ и *p*+ на рис. 2 являются фрагментами защитных колец.

3. Особенности сбора заряда с трека в элементе ИЛИ

На рис. 3 приведен эскиз топологии элемента И. Точки входа трека находятся в группе NMOП транзисторов Gr4N, линейная передача энергии на трек 60 МэВ·см^2/мг. Маркеры «звездочка» на рис. 3 соответствуют точкам входа трека одиночной частицы с направлением трека по нормали к поверхности приборной модели. «Звездочки», отмеченные красным цветом, соответствуют трекам, моделирование которых проведено при линейной передачи энергии на трек в диапазоне 10–90 МэВ·см^2/мг.

На рис. 4 приведены зависимости, иллюстрирующие формирование импульсов помех

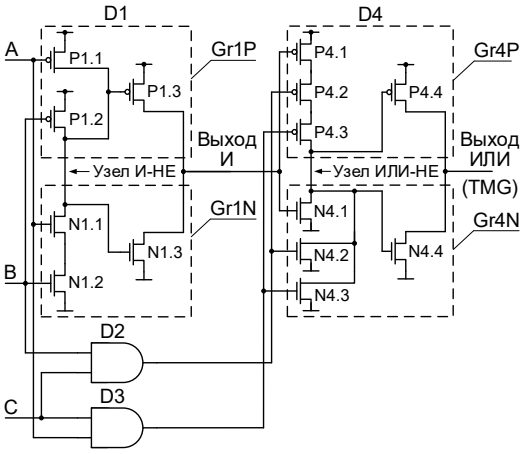


Рис. 1. Схема тройного мажоритарного элемента на двухвходовых КМОП логических элементах И (D1-D3) и ИЛИ (D4). Элементы И (D1) и ИЛИ (D4) на рис. 1 изображены в виде электрических схем. КМОП транзисторы элементов И и ИЛИ объединены в отдельные группы по типам проводимости Gr1N, Gr4N и Gr1P, Gr4P

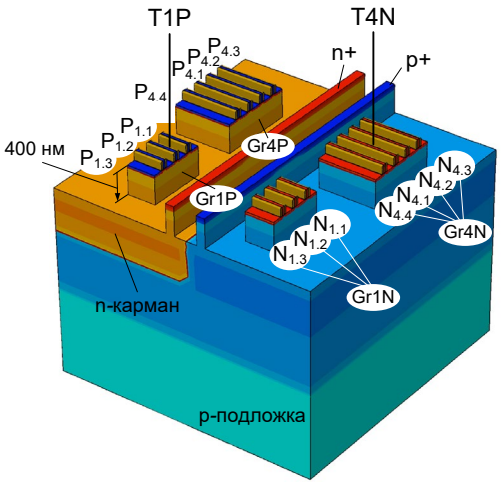


Рис. 2. 3-D приборная физическая TCAD модель логических элементов И (D1) и ИЛИ (D4) тройного мажоритарного элемента; направления треков T1P и T4N по нормали к поверхности кристалла; n+ и p+ области – фрагменты защитных колец. Ширина транзисторов элемента И составляет 400 нм, а элемента ИЛИ 800 нм. Кремниевые области групп транзисторов Gr1N и Gr1P имеют размеры 885×400×400 нм^3, а групп Gr4N и Gr4P – размеры 1180×800×400 нм^3

на узлах элемента ИЛИ при сигналах на входах тройного мажоритарного элемента A = B = C = 0. Трек возникает при t = 100 пс.

В случае элемента ИЛИ с точкой входа трека 4n после переключения инвертора при образовании трека происходит снижение напряжения на стоке NMOП транзистора инвертора до минимума (рис. 4) при сборе заряда (электронов) с

трека частицы. Этот уровень напряжения сохраняется, попадая в диапазон $V_{МИН.ИЛИ4n}$ = +(0.02–0.7) В для треков с LET = 60–90 МэВ·см^2/мг. При этом на выходе ИЛИ образуется «плато» с незначительно возрастающим напряжением во времени, что формирует задержку начала формирования импульса помехи, что дает сокращение длительности импульса помехи.

Для элемента ИЛИ с входной точкой трека 5n минимальное напряжение на его выходе при сборе электронов не опускается ниже $V_{МИН.ИЛИ.5n}$ = 0.086–0.1 В даже для LET ≥ 80 МэВ·см^2/мг (зависимости импульсов для 5n на рис. 5). Это относится к элементу ИЛИ с входной точкой 5n после переключения инвертора в начале сбора заряда с трека и снижения напряжения на стоке NMOП транзистора инвертора (выход ИЛИ). В этом случае для входной точки трека 5n «плато» почти постоянного напряжения на выходе ИЛИ не образуется, а сразу начинается процесс формирования импульса помехи положительной полярности.

Но некоторая задержка образования импульса помехи на выходе ИЛИ по уровню 0.7 В все-таки происходит (и уменьшение длительности импульса помехи по этому уровню) из-за снижения ниже 0.7 В (рис. 5) минимального напряжения уровня импульса отрицательной полярности, возникающего при большом сборе заряда электронов NMOП транзистором инвертора при передаче энергии на трек больше, чем 40 МэВ·см^2/мг.

Фронт импульса помехи на выходе элемента ИЛИ для трека с точками входа как 4n, так и 5n, начинает формироваться при увеличении напряжения на узле выхода ИЛИ за счет заряда емкости узла током открытого PMOП транзистора инвертора за 25–40 пс до окончания сбора и вывода заряда NMOП транзисторами узла ИЛИ-НЕ. Амплитудные значения импульсов помех на выходе ИЛИ для точек входа трека как 4n, так и 5n при всех LET = 10–90 МэВ·см^2/мг формируются при напряжениях на узле ИЛИ-НЕ в одном и том же диапазоне 0–0.1 В при запертом по затвору NMOП транзисторе инвертора.

Пример на рис. 4 демонстрирует это для входных точек трека 4n и 5n при LET = 60 МэВ·см^2/мг. Это выполняется для всех треков с LET = 10–90 МэВ·см^2/мг в режиме, когда NMOП транзисторы этого узла ИЛИ-НЕ заперты, а PMOП транзисторы открыты (рис. 5).

PMOП транзистор инвертора открыт и продолжает заряжать емкость выходного узла ИЛИ и после того, как импульс помехи на выходе ИЛИ превысит уровень 0.7 В и начнется формирование вершины импульса помехи. В это

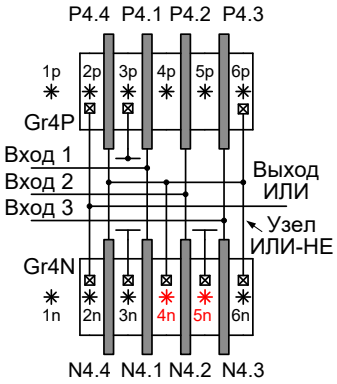


Рис. 3. Эскиз топологии элемента ИЛИ, маркер «звездочка» обозначает точки входа трека

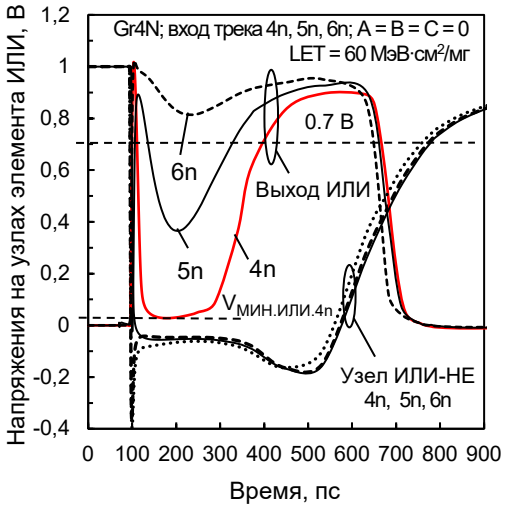


Рис. 4. Зависимости напряжений на узлах элемента ИЛИ для точек входа трека 4n, 5n и 6n в группу Gr4N при входах мажоритарного элемента A = B = C = 0, LET = 60 МэВ·см^2/мг

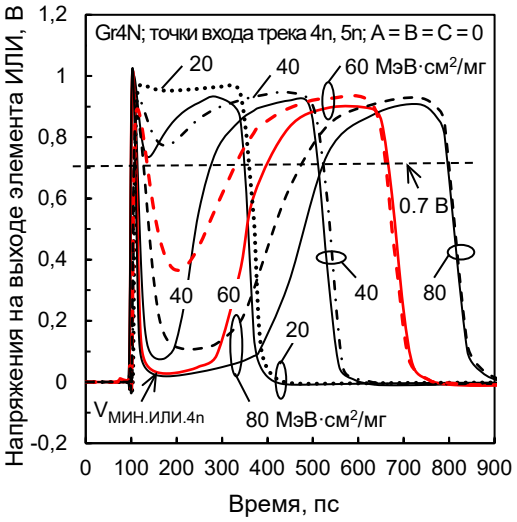


Рис. 5. Импульсы помех на выходе ИЛИ для точек входа трека 4n, 5n в группу Gr4N при передаче энергии на трек в диапазоне LET = 20-80 МэВ·см^2/мг

время напряжение между стоком и истоком этого РМОП транзистора становится менее 0.3 В, это переводит его из пологой в крутую область вольтамперной характеристики, где ток стока падает с уменьшением напряжения на стоке. Это вызывает замедление заряда емкости выходного узла ИЛИ и замедление формирования вершины импульса, что увеличивает длительность импульса помехи.

Когда напряжение на узле ИЛИ-НЕ и входе инвертора превышает 0.3 В, импульс помехи на выходе ИЛИ достигает уровня 0.7 В после прохождения амплитудного значения. Затем инвертор переключается по входу и происходит резкое понижение напряжения на выходе ИЛИ, что возвращает в исходное стационарное состояние выход элемента ИЛИ.

4. Особенности сбора заряда с трека в элементе И

На рис. 6 приведен эскиз топологии элемента И. Маркеры «звездочка» на рис. 6, отмеченные красным цветом, соответствуют трекам, моделирование которых проведено в диапазоне линейной передачи энергии на трек 10–90 МэВ·см^2/мг.

На рис. 7 приведены зависимости, иллюстрирующие формирование импульсов помех на узлах элемента И при входах мажоритарного элемента A = B = 1, C = 0. Точки входа треков 3p, 4p, 5p находятся в группе РМОП транзисторов Gr1P, линейная передача энергии на трек 60 МэВ·см^2/мг.

Запертый РМОП транзистор инвертора после образования трека и переключения инвертора начинает собирать заряд (дырки), что увеличивает напряжение на выходе инвертора и формирует на выходе И перепад напряжения положительной полярности, по окончанию которого формируется фронт импульса помехи отрицательной полярности.

На рис. 8 для зависимостей «Выход И» для случая трека 4p показано, что напряжение на выходе И не поднимается выше отмеченного максимума $V_{МАКС.И}$ = 0.74 В при LET = 90 МэВ·см^2/мг при сборе заряда РМОП транзистором инвертора. В результате перед образованием фронта импульса помехи отрицательной полярности не создается «плато» напряжения. В этом случае сразу начинает формироваться импульс помехи. Некоторая задержка образования импульса помехи по уровню 0.3 В все-таки происходит за счет превышения напряжения 0.3 В импульсом положительной полярности при сборе большого заряда РМОП транзистором инвертора для LET>30 МэВ·см^2/мг (рис. 8).

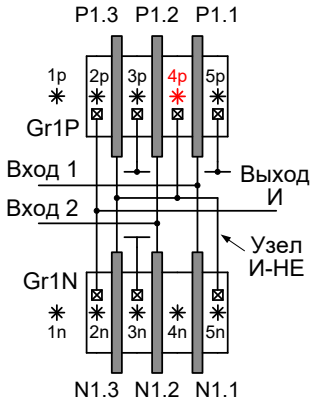


Рис. 6. Эскиз топологии элемента ИЛИ, маркер «звездочка» обозначает точки входа трека

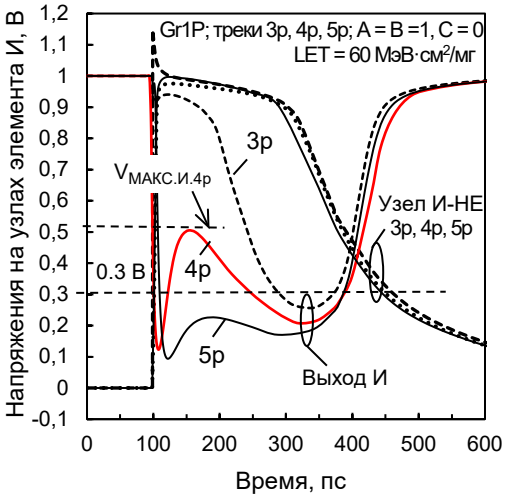


Рис. 7. Зависимости напряжений на узлах элемента И для точек входа трека 3p, 4p и 5p в группу Gr4N при входах мажоритарного элемента A = B = C = 0, LET = 60 МэВ·см^2/мг

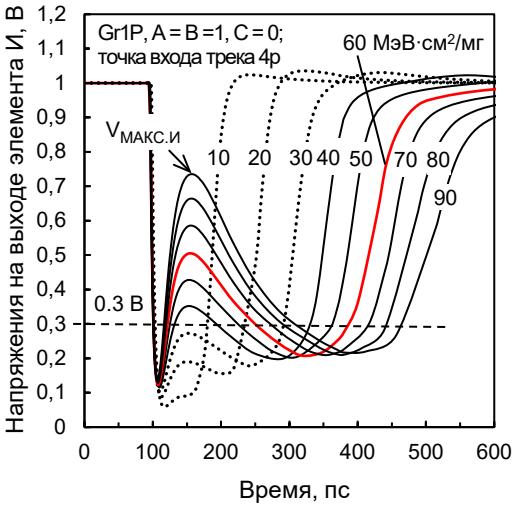


Рис. 8. Импульсы помех на выходе ИЛИ для точки входа трека 4p в группу Gr1P при передаче энергии на трек в диапазоне 10-90 МэВ·см^2/мг

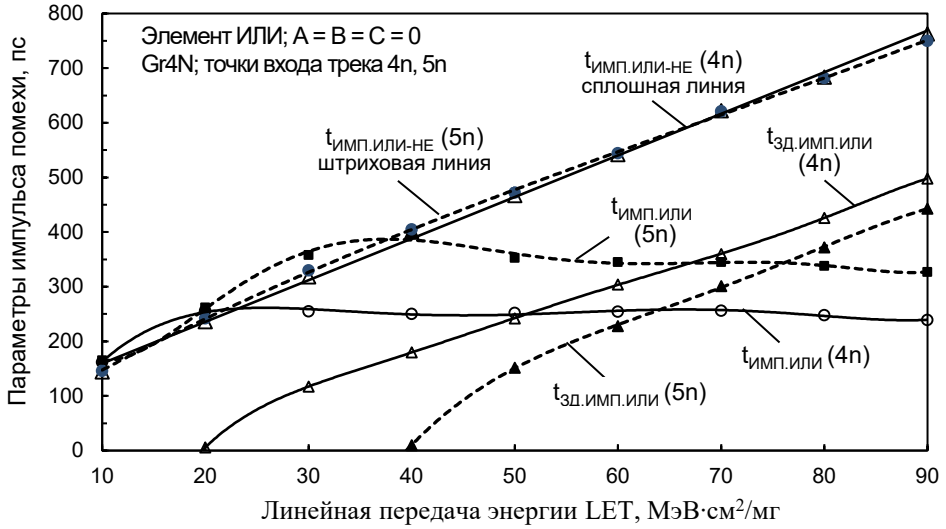


Рис. 9. Временные параметры импульсов помех в группе транзисторов Gr4N элемента ИЛИ в зависимости от линейного переноса энергии частицей на трек при точках входа трека 4n и 5n

Амплитудные значения импульсов помех на выходе И при всех LET в диапазоне 10–90 МэВ·см^2/мг образуются, когда на узле И-НЕ напряжение в диапазоне 0.75–0.85 В и РМОП транзистор этого узла заперт. После достижения помехой амплитудного значения начинается переключение инвертора. При этом на выходе И увеличивается напряжение после того, как на узле И-НЕ и входе инвертора напряжение опускается ниже 0.7 В относительно напряжения на общей шине элемента (пример для LET = 60 МэВ·см^2/мг дан на рис. 7). В итоге на выходе И восстанавливается стационарное состояние логической единицы, соответствующее сигналам на входах мажоритарного элемента.

Образование импульсов помех на выходе элемента И происходит аналогично как для элемента ИЛИ при прохождении трека через точку входа 5n без формирования «плато» перед фронтом импульса помехи.

5. Анализ результатов сбора заряда элементами с трека

5.1. Оценки значений параметров

На рис. 9 приведены зависимости временных параметров импульсов помех элемента ИЛИ с точками входа трека 4n и 5n в диапазоне 10–90 МэВ·см^2/мг линейного переноса энергии на трек при сигналах на входах мажоритарного элемента A = B = C = 0. Эти зависимости включают длительности импульсов на узле ИЛИ-НЕ $t_{имп.или-не}$ по уровню 0.3 В, длительности импульсов помех на выходе ИЛИ $t_{имп.или}$ по уровню 0.7 В, а также задержки нарастания импульсов до уровня 0.7 В на выходе ИЛИ $t_{зд.имп.или}$.

Длительности импульсов на узле ИЛИ-НЕ для точек входа трека 4n и 5n являются линейными функциями переноса энергии на трек. Эти зависимости на рис. 9 одинаковы в диапазоне 10–90 МэВ·см^2/мг, не зависят от входных точек трека и характеризуют суммарный сбор заряда в группе Gr4N.

Длительности задержек нарастания импульсов помех до уровня 0.7 В на выходе ИЛИ $t_{зд.имп.или}$ являются практически линейными функциями значений LET, начиная с 30 МэВ·см^2/мг для входной точки трека 4n, и с 50 МэВ·см^2/мг для точки 5n. Разница между длительностью импульса на узле ИЛИ-НЕ $t_{имп.или-не}$ и длительностью задержки импульса помехи на выходе ИЛИ $t_{зд.имп.или}$ составляет 200–265 пс при точке входа трека 4n и 310–330 пс при точке входа трека 5n. Для точки входа трека 4n значения задержки увеличиваются на 60 пс при каждом приращении LET на 10 МэВ·см^2/мг из-за удлинения «плато» уровня напряжения перед импульсом помехи. Задержки в случае точки входа трека 5n при отсутствии «плато» меньше на 60–110 пс.

Длительность импульса помехи на выходе ИЛИ $t_{имп.или}$ для точки входа трека 4n практически повторяет длительность импульса на узле ИЛИ-НЕ $t_{имп.или-не}$ (рис. 9) в интервале LET = 10–20 МэВ·см^2/мг. Для входной точки трека 5n это повторение происходит в интервале LET = 10–40 МэВ·см^2/мг. Электронов, образованных на треке при малых значениях передачи энергии, недостаточно для эффективной диффузии к NMOП транзистору инвертора. В этом случае происходит просто инверсия на выход ИЛИ импульса с узла ИЛИ-НЕ при небольшом увеличении его длительности (рис. 9).

Снижение длительности импульса помехи

выходе ИЛИ происходит при LET более 30 МэВ·см^2/мг для трека с точкой входа 4n и при LET более 40 МэВ·см^2/мг для трека с точкой входа 5n. При этом длительность импульсов помехи снижается до уровня 250 пс для точки трека 4n и до 350 пс для точки входа трека 5n. Длительность помехи на выходе ИЛИ $t_{имп.или}$ для точки входа трека 5n оказывается в 1.4–1.6 раза больше (рис. 9) в диапазоне LET = 30–90 МэВ·см^2/мг, чем длительности импульсов помех для точки входа трека 4n.

На рис. 10 зависимости характеризуют параметры импульсов помех элемента И с точкой трека 4p при входах мажоритарного элемента A = B = 1, C = 0. Это длительности импульсов на узле И-НЕ $t_{имп.и-не}$ по уровню 0.7 В, длительности импульсов помех на выходе И $t_{имп.и}$ по уровню 0.3 В, а также задержки нарастания импульсов $t_{зд.имп.и}$.

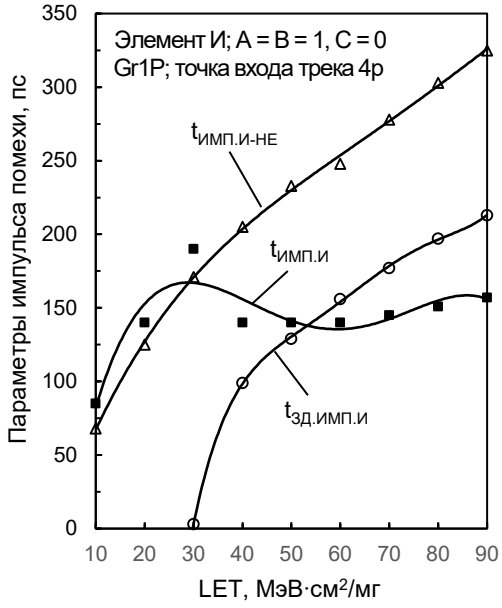


Рис. 10. Временные параметры импульсов помех в группе транзисторов Gr1P элемента И в зависимости от линейного переноса энергии частицей на трек при точке входа трека 4p.

В случае элемента И, напряжение на его выходе изменяется (рис. 8) практически зеркально с учетом полярности напряжений по отношению элемента ИЛИ, когда заряд в элементе ИЛИ собирается (рис. 5) без «плато» перед фронтом импульса помехи для трека с точкой входа 5n.

Длительности импульсов помех на выходе И с повторяют небольшой добавкой (рис. 10) длительности импульсов на узле И-НЕ в интервале LET = 10–30 МэВ·см^2/мг. При LET = 40 МэВ·см^2/мг и более длительности импульсов помех на выходе И снижаются до значений $t_{имп.и}$ = 140–150 пс.

Длительность задержки снижения напряжения импульса помехи до уровня 0.3 В на выходе И $t_{зд.имп.и}$ является практически линейной функцией значений LET, начиная с 40 МэВ·см^2/мг. Зависимости длительностей импульсов на узле И-НЕ $t_{имп.и-не}$ и задержки импульса помехи на выходе И $t_{зд.имп.и}$ (рис. 10) практически одинаковы во времени с разницей между их значениями по времени 100–110 пс.

5.2. Особенности образования помех

Группы Gr4N элемента ИЛИ и Gr1P элемента И содержат по три транзистора, включая два транзистора с общими стоками и транзистор инвертора элемента, затвор которого соединен с общими стоками двух других транзисторов. Завершение образования задержки импульса помехи и его фронт для всех LET в диапазоне 30–90 МэВ·см^2/мг происходит в элементе ИЛИ при напряжении -0.055±0.025 В на затворе NMOП транзистора инвертора элемента ИЛИ и при 0.96±0.02 В на затворе PMOП транзистора инвертора элемента И, когда эти транзисторы инверторов заперты.

В запертом режиме инверторов при напряжениях между стоком и истоком близком к нулю фронт импульса помехи формируется зарядом емкости выходного узла током открытого PMOП транзистора инвертора до напряжения 0.7 В на выходе ИЛИ, а в элементе И зарядом емкости выходного узла током открытого NMOП транзистора инвертора до напряжения 0.3 В на выходе И. Одновременное увеличение напряжения на узле ИЛИ-НЕ при окончании сбора заряда NMOП транзисторами узла ИЛИ-НЕ переключает инвертор, что и завершает формирование импульса помехи на выходе ИЛИ. В элементе И этот процесс происходит зеркально при другой полярности импульса помехи за счет разряда емкости узла И-НЕ, что переключает инвертор и завершает формирование импульса помехи на выходе И.

6. Заключение

Представленный анализ особенностей элементов полезен при проектировании КМОП микропроцессорных систем космического назначения. Импульс помехи на выходе логического элемента становится ложным сигналом, когда его амплитуда превышает порог переключения следующего элемента в цепочке. В результате проведенного исследования, обоснованного достоверными результатами моделирования, установлено, что увеличение задержки образования импульсов помехи на выходах логических элементов ИЛИ и И при сборе заряда

с трека группами из трех транзисторов, выполненных в минимальном объеме кремния, ограниченном мелкой траншейной изоляцией, приводит к уменьшению длительности импульсов помех при треках с линейным переносом энергии свыше 30–40 МэВ·см^2/мг.

Работа выполнена в рамках Госзадания, проект № 0065-2019-0008 «Архитектурные и схемотехнические методы снижения энергопотребления и повышения сбоеустойчивости микропроцессоров и коммуникационных контроллеров высокопроизводительных ЭВМ».

Modeling of CMOS elements resistance to interference when exposed to single particles using TCAD

Yu.V. Katunin, V.Ya. Stenin

Abstract. The results are presented for modeling of noise pulses inside logic elements of a majority gate when collecting charge from tracks of single particles in a wide range of linear energy transfer by a particle to the tracks of the range 10–90 MeV·cm^2/mg. The study was performed using 3D TCAD physical models of CMOS transistors of 65 nm bulk technology with shallow trench isolation of transistor groups. Most importantly, what happens when the charge collection with the track passing through the group NMOS transistors of OR element, as well as in case through the group of the PMOS transistors AND element, is to hold these transistors groups in the mode when all transistors are simultaneously collect charges, in particular while in reverse bias mode. When all transistors of the group are simultaneously collect charge, it forms a delays of noise pulses produced at the outputs of the elements, which reduces the duration of these pulses. The noise pulse duration is reduced to an average level of 250–350 ps for the input track points in the group of NMOS transistors of the element OR on the range 30–90 MeV·cm^2/mg. For input track points in the group of PMOS transistors of the element AND, the noise pulse duration at the output AND is reduced to the level of 140–150 ps on the range 40–90 MeV·cm^2/mg.

Keywords: noise pulse, logical element, simulation, majority gate, single nuclear particle, topology, particle track, charge collection

Литература

1. P.E. Dodd, M.R. Shaneyfelt, J.A. Felix, J.R. Shwank. Production and propagation of single-event transients in high-speed digital logic ICs. «IEEE Transactions on Nuclear Science», v. 51 (2004), no. 6, 3278–3284.

2. P.E. Dodd, L.W. Messengill. Basic mechanisms and modeling of single-event upset in digital microelectronics. «IEEE Transactions on Nuclear Science», v. 50 (2003), no. 3, 583–602.

3. N.M. Atkinson, A.F. Witulski, W.T. Holman, J.R. Ahlbin, B.L. Bhuva, L.W. Massengill. Layout technique for single-event transient mitigation via pulse quenching. «IEEE Transactions on Nuclear Science», v. 58 (2011), no. 3, 885–890.

4. Yu.V. Katunin, V.Ya. Stenin. Modeling of single ionizing particles impact on logic elements of a CMOS triple majority gate. «Russian Microelectronics», v. 49 (2020), no. 3, 214–223.

5. R. Garg, S.P. Khatri. Analysis and design of resilient VLSI circuits: mitigating soft errors and process variations. New York: Springer, 2010. 194–205.

6. Soft errors in Modern Electronic Systems / M. Nicolaidis, Ed. New York: Springer, 2011. 27–54.

Отладка графической подсистемы СнК с использованием аппаратного комплекса для прототипирования Palladium Z1

А.Ю. Богданов[1]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, aubogdan@cs.niisi.ras.ru;

Аннотация. Рассматривается задача прототипирования графической подсистемы СнК. Даётся описание используемого стенда для отладки. Приводятся полученные результаты работы.

Ключевые слова: прототип, СнК, Palladium Z1

1. Введение

В настоящее время развитие микроэлектронной индустрии идёт по пути интеграции различный подсистем на одном кристалле. Системы на кристаллах (СнК) активно применяются в различных сферах человеческой деятельности: автоматизация, авиа- и железнодорожная промышленность, цифровая экономика, искусственный интеллект и т.д. Одним из ключевых показателей, помимо производительности, является наличие графической подсистемы [1].

В ФГУ ФНЦ НИИСИ РАН (далее по тексту НИИСИ РАН) разрабатывается СнК, имеющая следующие функции работы графической подсистемы:

- Поддержка аппаратного ускорения трёхмерной и двухмерной графики;
- Поддержка интерфейса DVI;
- Поддержка интерфейса DisplayPort:
  - Версия спецификации 1.1;
  - Масштабирование линий передачи 1x, 2x, 4x.

Для отладки данной графической подсистемы необходима система прототипирования, соответствующая следующим требованиям:

- Наличие большого объёма логических элементов, достаточных для прототипирования трёхмерной графики (минимум 80 млн. логических элементов);
- Возможность удалённой работы с прототипом;
- Возможность совместной отладки ПО и аппаратуры;
- Возможность использования стенда как на начальной стадии проекта, так и на протяжении всего маршрута проектирования;
- Доступ ко всем сигналам проекта во время работы;
- Наличие минимального набора внешних интерфейсов для работы с прототипом:
  1. UART - инициализация, управление;
  2. Ethernet – загрузка образа ОС и исполняемых программ.

В настоящее время для целей прототипирования используются как платы собственной разработки, так и готовые отладочные платы от сторонних производителей (Terasic, Hitechglobal и др.). В большинстве случаев в данных решениях применяется плата с одной ПЛИС, объём вычислительных ресурсов которой не достаточен для размещения блока трёхмерной графики.

Для прототипирования проектов требующих разбиение на несколько ПЛИС в НИИСИ РАН применяется платформа Protium S1 от компании Cadence. Она представляет собой набор 2-х независимых плат с 4-мя ПЛИС Virtex-7 каждая. Специализированное ПО данной системы позволяет в автоматическом режиме разбивать проект на части для загрузки в отдельные ПЛИС. Во время работы данной платформы для отладки доступно только часть заранее указанных сигналов на этапе компиляции проекта [2], что не удовлетворяет вышеприведённым требованиям к отладочному стенду для графической подсистемы.

В НИИСИ РАН в маршруте проектирования СБИС активно применяется аппаратный комплекс для прототипирования Palladium Z1. С его помощью успешно отлажена система, состоящая из 8-ми микропроцессоров, объединённых в единую сеть по интерфейсу RapidIO - воспроизведены зависания, получаемые в реальной системе.

На основе полученного опыта было принято решение построить прототип графической подсистемы с использованием данного аппаратно-программного комплекса.

2. Платформа Palladium Z1

Платформа Palladium Z1 [3] компании Cadence является системой эмуляции класса ЦОД. Общий вид комплекса представлен на рис 1. Она ускоряет проведение верификации СнК, подсистем и СФ-блоков, а также проведение валидации на уровне системы.

Рис. 1. Платформа Palladium Z1

Данный комплекс можно использовать на различных этапах разработки СБИС, начиная с анализа архитектуры, проверки логической модели блока и заканчивая интеграцией системного уровня с программным обеспечением для полной проверки системы. Основой аппаратного комплекса является матрица специализированных процессоров и специализированное ПО Integrated Compile Engine [4].

Ключевые особенности платформы Palladium Z1:

- Доступ ко всем сигналам проекта во время моделирования;
- Поддержка неограниченного количества сигналов тактовой частоты в проекте;
- Возможность сохранять/загружать состояние работы прототипа;
- Максимальная частота работы прототипа до 4-х МГц;

Эти особенности позволяют проводить совместную отладку ПО и аппаратуры, воспроизводить ситуации получаемые в реальных системах.

В составе используемой в НИИСИ РАН платформы доступно 24 логических домена (до 96М вентилей), максимальное количество одновременно работающих пользователей зависит от количества логических доменов и размеров запускаемых проектов.

3. Отладочный стенд графической подсистемы СнК

В составе графической подсистемы используются два контроллера вывода на экран, ядра 2D и 3D графики (рис. 2).

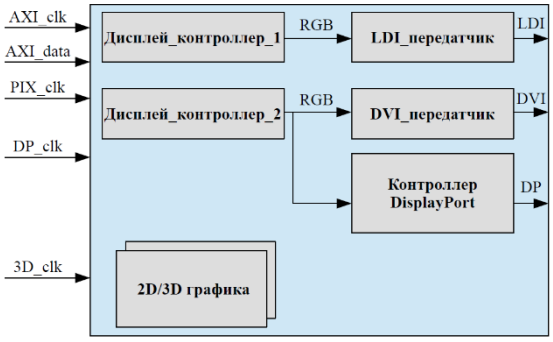


Рис. 2. Состав графической подсистемы СнК

Контроллер_вывода_на_экран_2 предназначен для вывода видеосигнала формата RGB 24-бита через *DVI_передатчик* на DVI интерфейс и через *контроллер_DisplayPort* на соответствующий интерфейс. *Контроллер_вывода_на_экран_1* используется для вывода видеосигнала формата RGB 24-бита через *LDI_передатчик* на LDI интерфейс. Передача данных и запись/чтение регистров контроллеров осуществляется через AXI интерфейс. *PIX_clk, 3D_clk, DP_clk* являются опорными частотами для соответствующих блоков контроллеров интерфейса.

Для проверки графической подсистемы используется отладочный стенд (рис.3), в состав которого входит следующее оборудование:

- Аппаратно-программный комплекс для прототипирования Palladium Z1;
- Video SpeedBridge (VSB).

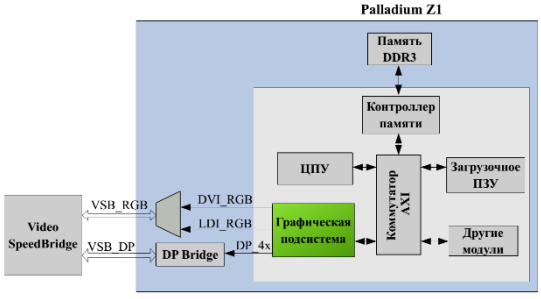


Рис. 3. Отладочный стенд графической подсистемы в Palladium Z1

В системе Palladium Z1 реализован прототип микропроцессора, который запущен в режиме внутрисхемной эмуляции (In-Circuit Emulation mode). В качестве DDR3 памяти используется

эмулируемая модель от компании Cadence, предназначенная для работы с Palladium. Она соответствует планке памяти mt16jtf51264az_udimm объемом 4 Гбайта от компании Micron. Так как система Palladium Z1 не эмулирует PHY уровень интерфейсов, то выводится видеосигнал изображения, до его преобразования на физический уровень *DVI_RGB*, *LDI_RGB*.

Выбор формата выводимого изображения RGB или DisplayPort осуществляется на этапе компиляции проекта.

VSB осуществляет видео захват передаваемого изображения в реальном времени и позволяет записывать его на диск для дальнейшего анализа [5]. Для вывода изображения в формате Displayport на VSB используется специализированный мост *DP_Bridge* (заменяет собой PHY уровень), который преобразует полученное изображение после кодирования 8/10 бит в формат VSB.

Для взаимодействия с системой используется “виртуальный” UART интерфейс, реализованный с помощью языка Tcl и инструментальных средств среды Palladium Z1: ввод команд осуществляется через командную строку, вывод информации осуществляется непосредственно в файл.

4. Полученные результаты

В ходе работы удалось достичь следующих результатов: загружена ОС Linux, отлажены драйверы для поддержки блоков 2D/3D графики, интерфейса DisplayPort. На рис. 4 представлено полученное изображение формата 1920x1080 интерфейса DisplayPort на VSB.

Для удобства дальнейшей отладки ПО для блоков двухмерной и трёхмерной графики (скорость работы прототипа) был запущен аналогичный проект на платформе Protium S1.

Для проверки работы интерфейса DisplayPort с мониторами различных производителей был запущен аналогичный проект на ПЛИС Arria 10 GX. На рис.5 показано изображение, полученное на данном стенде на интерфейсе DisplayPort.

Рис. 5. Полученное изображение интерфейса DisplayPort на Arria 10 GX

Сравнительная характеристика запуска данного проекта в различных платформах прототипирования приведена в табл.1.

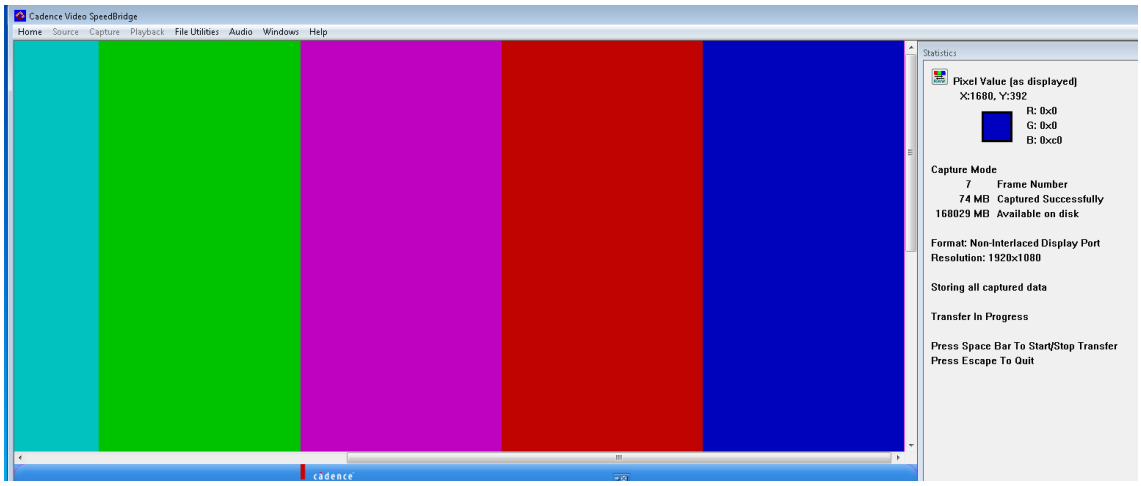


Рис. 4. Полученное изображение формата 1920x1080 интерфейса DisplayPort на VSB

Таблица 1. Сравнение запуска проекта на платформах Palladium/Protium/Arria 10 GX

Платформа прототипирования	Состав проекта	Время сборки	Скорость работы	Возможность отладки
Palladium	3D/2D DVI Displayport	10 мин	1.2 МГц	Полная - доступ ко всем сигналам проекта во время работы (не требуется пересборка проекта)
Protium S1	3D/2D DVI	4 – 6 часов	8 МГц	Частичная – требуется пересборка проекта для вывода новых сигналов для отладки
Arria 10 GX	Displayport	4 часа	60 МГц	

Как видно из приведённых данных в табл 1., систему Palladium Z1 можно использовать на протяжении всех стадий проекта. Прототипы на ПЛИС, обладающие более высокой скоростью работы удобно использовать для наладки необходимого ПО. В случае обнаружения аппаратной ошибки в прототипах на ПЛИС проект можно перенести в систему Palladium Z1 для дальнейшего воспроизведения ошибки и анализа возникающей ситуации. Время переноса проекта между представленными прототипами занимает около 10 минут при условии предварительной подготовки проектов для данных стендов.

5. Заключение

Для отладки графической подсистемы СнК был использован аппаратно-программный комплекс для прототипирования Palladium Z1. Он позволил на ранних стадиях проекта успешно запустить графическую подсистему СнК, выполнить первоначальную наладку аппаратуры и ПО.

К достоинствам комплекса относится скорость сборки проекта, доступ ко всем сигналам проекта во время работы, объём доступных логических ресурсов.

К недостаткам комплекса можно отнести относительно невысокую скорость моделирования, что является несущественным по сравнению с предоставляемыми возможности для отладки проектов.

Публикация выполнена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН по теме №0065-2019-0004

Debugging the SoC graphics subsystem using the Palladim Z1 prototyping platform

A.Y. Bogdanov

Abstract. The problem of prototyping the graphic subsystem of the SoC is considered. The description of the debugging stand used is given. The results of the work are given.

Keywords: prototype, SoC, Palladium Z1

Литература

1. В.Н. Куцевол, А.Н. Мешков, Е.М. Николаева, С.В. Черных. Верификация графической подсистемы, интегрированной в процессор. Вопросы радиоэлектроники, серия “Электронная вычислительная техника”, выпуск 3, 2014.

2. А.Ю. Богданов. Опыт применения платформы прототипирования на ПЛИС “Protium” для ве-

рификации микроппроцессоров.”Труды НИИСИ РАН”, Том 7 (2017), № 2.
3. URL: https://www.cadence.com/en_US/home/tools/system-design-and-verification/ acceleration-and-emulation/palladium-z1.
4. VXE User Guide. Product Version 16.5. Cadence Design Systems. August 2017.
5. Video-Audio SpeedBridge User Guide. Cadence Design Systems. July 2017.

Параметры камеры просмотра видео 360 градусов

М.В. Михайлюк[1], Д.В. Омельченко[2], Д.А. Кононов[3], Д.М. Логинов[4]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, mix@niisi.ras.ru;

[2]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, mix@niisi.ras.ru;

[3]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, mix@niisi.ras.ru;

[4]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, mix.niisi.ras.ru

Аннотация. В работе рассматривается выбор параметров виртуальной камеры для просмотра видео 360 градусов, созданного с помощью кубической проекции. Проведены оценки искажения изображения при поворотах камеры для разных углов ее горизонтального поля зрения. Предлагаются алгоритмы реализации поворотов камеры в процессе просмотра видео 360 градусов.

Ключевые слова: виртуальное окружение, реальное время, видео 360 градусов, кубическая проекция, параметры виртуальной камеры, повороты виртуальной камеры

1. Введение

В общем случае система виртуального окружения (СВО) [1-4] включает подсистему управления динамическими объектами, подсистему расчета динамики этих объектов и подсистему визуализации. Управление динамическими объектами осуществляется либо пользователем (оператором) СВО с помощью устройств или пультов управления, голосовыми или жестовыми командами, используя экзоскелет или системы отслеживания (трекинга) движений. По результатам этих управляющих воздействий подсистема управления формирует управляющие сигналы на исполнительные устройства (двигатели) динамических объектов и передает их в подсистему динамики. Эта подсистема вычисляет результат воздействий путем вычисления новых позиций и ориентаций динамических объектов виртуальной сцены и передает вычисленные координаты и углы ориентации в подсистему визуализации, которая синтезирует изображения части сцены, наблюдаемой через виртуальную камеру. Для обеспечения масштаба реального времени и создания видимости непрерывного процесса, цикл формирования каждого кадра изображе-ния не должен превышать 40 мсек.

Кроме просмотра изображений на экране монитора у пользователя может возникнуть желание или необходимость как можно более адекватно зафиксировать наблюдаемые им в СВО процессы. Например, для передачи другому лицу эффекта погружения в виртуальную среду, для фиксации некоторого происходящего в ней интересного события, для последующего анализа проходящего процесса или эксперимента, для последующего анализа действий оператора, для будущих презентаций и т.д.

Обычно для этих целей используется запись видео из камеры наблюдателя. Однако просмотр такого видео не позволяет увидеть, а что же происходило сбоку или сзади камеры просмотра. Во многих случаях это может оказаться важным и интересным. Для этих целей хорошо подходит видео 360 градусов (V360). Его идея состоит в том, чтобы в видеофайл записывать не только изображение из камеры наблюдателя, но и изображения вокруг него, используя несколько камер. Существуют разные подходы к созданию и просмотру V360. Здесь мы рассмотрим создание видео 360 градусов с помощью кубической проекции [5].

2. Постановка задачи

Технология создания и просмотра V360, основанного на кубической проекции состоит из следующих этапов.

1. Наблюдая сцену в системе виртуального окружения (СВО) через виртуальную камеру (обозначим ее через $C_{наб}$), наблюдатель нажимает клавишу начала записи V360. В этот момент в сцене в точке расположения камеры $C_{наб}$ создается и включается новый объект – камера 360 градусов (С360), состоящая из главной камеры C_0 и пяти дополнительных камер C_i (i = 1,...,5). Все камеры имеют перспективную проекцию, вертикальный угол обзора (FOV), равный 90 градусов, и отношение ширины к высоте формируемого изображения (aspect), равное единице. Главная камера направлена туда же, куда направлена камера наблюдателя, а осталь-

ные пять камер направлены вдоль осей локальной системы координат камеры C_0 под углами 90 градусов друг к другу. Таким образом, при перемещении и повороте камеры наблюдателя $C_{наб}$ камера C360 перемещается и поворачивается вместе с ней. Камера $C_{наб}$ может перемещаться наблюдателем с помощью клавиатуры, джойстика, пульта управления и т.д. Она также может двигаться по заданной траектории, которая может вычисляться (как, например, орбита Международной космической станции) или являться копией траектории движения управляемого объекта (например, привязана к управляемому роботу), задаваться анимационным треком и др.

Далее, с частотой 25 кадров в секунду из камер C_i формируются изображения виртуальной сцены, которые записываются в двумерные RGB-текстуры некоторого фиксированного размера (например, 1024x1024 пикселов). Эти текстуры объединяются в одну, результат сжимается и записывается в видеофайл в качестве очередного кадра (это можно сделать, например, с помощью набора библиотек FFmpeg [6]). Имя видеофайла можно формировать автоматически, используя некоторый фиксированный префикс, а также дату и время записи.

2. Просмотр видео 360 градусов производится с помощью специально разработанного приложения. В нем можно выбрать нужный файл, затем создать виртуальную сцену для просмотра V360 и с частотой 25 кадров в секунду считывать и обрабатывать (проигрывать) его кадры. Сцена V360 представляет собой единичный куб, расположенный в центре мировой системы координат и ориентированный вдоль ее осей. Обработка кадра состоит в его распаковывании, выделении шести текстур и наложении этих текстур изнутри на грани куба. Для просмотра в центре куба создается виртуальная камера C_v с некоторым горизонтальным углом зрения φ и некоторым aspect > 1. На экране монитора производится визуализация изображения, видимого из этой камеры. Во время проигрывания видео интерфейс приложения позволяет пользователю поворачивать камеру C_v вокруг ее локальных осей координат, тем самым рассматривая окружающую обстановку из текущей точки проигрывания.

Параметрами камеры Cv являются угол φ горизонтального раствора (поля зрения), aspect, расстояния n = OO' и f = OO" до ближней и дальней плоскостей отсечения (см. рис. 1). При повороте камеры боковые ребра (например, OA) пирамиды видимости не должны пересекать грани куба (иначе, часть изображения на кубе будет отсечена ближней плоскостью камеры). Поэтому расстояние до ближней плоскости отсечения можно взять равным (см. [7])

$$n = \frac{1}{2\sqrt{tg^2(\varphi/2)(1+aspect^{-2})+1}} - \varepsilon$$

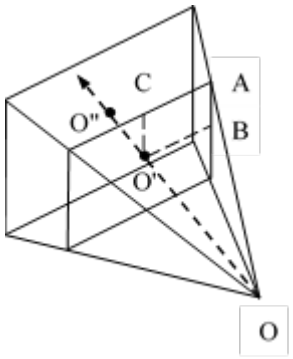


Рис. 1. Выбор ближней и дальней плоскостей отсечения

С другой стороны, дальняя плоскость должна располагаться дальше, чем вершины единичного куба, поэтому расстояние до нее берем чуть больше половины длины диагонали куба, т.е. $f = \sqrt{3}/2 + \varepsilon$. Значение aspect можно выбрать произвольным, потребуем лишь, чтобы оно превышало 1. Естественно, чтобы не было искажений изображения, область вывода (ViewPort) надо задавать с таким же aspect. За счет перспективной проекции формируемое изображение будет содержать искажения. Рассмотрим зависимость степени искажения изображения от величины угла φ зрения камеры.

3. Степень искажения V360

Рассмотрим некоторый горизонтальный отрезок M на изображении, наложенном на грань куба. Расположим камеру C_v так, чтобы этот отрезок находился возле ее центральной линии (см. рис. 2а). Обозначим угол, под которым виден этот отрезок в камере, через α. Пусть он проецируется на картинную плоскость в отрезок AC. Повернем теперь камеру по часовой стрелке на угол $\psi - \alpha$ (см. рис. 2б) – теперь тот же отрезок будет проецироваться на отрезок A'C' (заметим, что угол α, под которым виден отрезок, не изменится, так как не изменится расстояние до отрезка от положения камеры). Сравним длины отрезков AC и A'C', а вернее найдем отношение этих длин

$$k(\psi,\alpha) = \frac{A'C'}{AC}.$$

Так как $AC = n \cdot tg(\alpha)$, а

$A'C' = n \cdot tg(\psi) - n \cdot tg(\psi - \alpha)$, а

$$tg(\psi-\alpha)=\frac{tg(\psi)-tg(\alpha)}{1+tg(\psi)tg(\alpha)},\ \text{то}$$

$$k(\psi,\alpha)=\frac{A'C'}{AC}=\frac{n\cdot tg(\psi)-n\cdot tg(\psi-\alpha)}{n\cdot tg(\alpha)}=$$

$$=\frac{tg(\psi)-\dfrac{tg(\psi)-tg(\alpha)}{1+tg(\psi)tg(\alpha)}}{tg(\alpha)}.$$

Делая простые преобразования, получаем

$$k(\psi,\alpha)=\frac{1+tg^2(\psi)}{1+tg(\psi)tg(\alpha)}. \qquad (1)$$

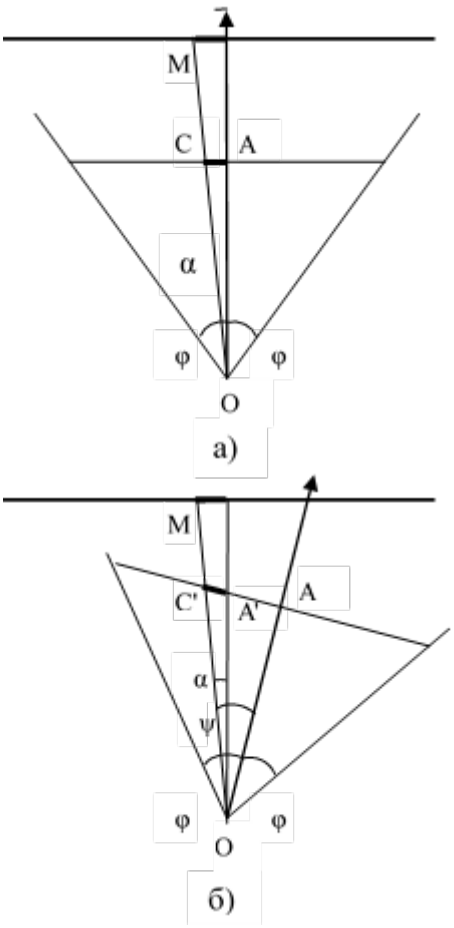


Рис. 2. Проекция объекта в различные части картинной плоскости

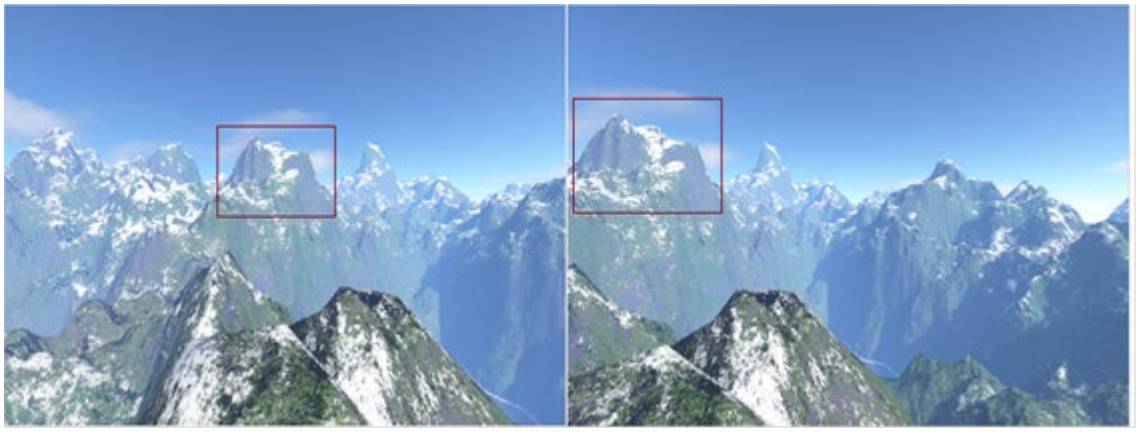
Легко видеть, что функция $k(\psi,\alpha)$ при любом фиксированном ψ является монотонно убывающей по α. Разделив числитель и знаменатель в формуле (1) на $tg(\psi)$, можно увидеть, что $k(\psi,\alpha)$ монотонно возрастает по ψ. Так как горизонтальный угол раствора камеры равен φ, то $\psi \le \varphi/2$ и максимальное значение $k(\psi,\alpha)$ будет при $\psi=\varphi/2$. Для каждого значения α можно вычислить величины $k(\psi,\alpha)$ для различных углов ψ. В таблице 1 показаны эти величины для значений $\alpha=0^O$, 5^O *и* 10^O и углов ψ с шагом 10 (2-й столбец таблицы). Естественно, при $\alpha=0^O$ берется предел отношения (1). В первом столбце приведены значения горизонтального угла φ зрения камеры. В таблице специально выделены строки для камер с углом зрения 46°, 50° и 54°, т.к. именно такие углы приблизительно соответствуют человеческому глазу и используются в фотоаппаратах и кинокамерах с так называемыми «нормальными» объективами. Из таблицы видно, что в каждом столбце с возрастанием φ и ψ коэффициент растяжения изображения $k(\psi,\alpha)$ также увеличивается. При этом, чем больше α, тем скорость роста меньше. Это значит, что большие предметы будут искажаться меньше маленьких. Тем не менее, при углах горизонтального раствора камеры, меньших 40°, искажения небольшие и могут быть малозаметны пользователю. При углах, больших 60° искажения сильно заметны. Поэтому для просмотра V360 следует выбирать камеры с раствором в районе 50 градусов.

Таблица 1. Зависимость искажения изображения от угла камеры

φ	ψ	$k(\psi,0)$	$k(\psi,5)$	$k(\psi,10)$
0	0	1	1	1
20	10	1,03	1,02	1
40	20	1,13	1,10	1,06
46	**23**	**1,18**	**1,14**	**1,1**
50	**25**	**1,22**	**1,17**	**1,13**
54	**27**	**1,26**	**1,21**	**1,16**
60	30	1,33	1,27	1,21
80	40	1,70	1,59	1,48
100	50	2,42	2,19	2
120	60	4	3,47	3,06
140	70	8,55	6,89	5,76
160	80	33,16	22,17	16,58
180	90			

На рис. 3 показаны скриншоты кубической проекции, сделанные камерой с горизонтальным раствором 90 градусов.

Обведенная прямоугольником сопка в центре камеры (левая картинка) и она же на краю камеры (правая картинка) различаются в размерах довольно сильно. На таких же видах, но для камеры с горизонтальным раствором 46 градусов (см. рис. 4) размеры этих сопок практически неразличимы.



а) объект в центре камеры б) объект на краю камеры

Рис. 3. Камера с горизонтальным углом зрения 90 градусов

а) объект в центре камеры б) объект на краю камеры

Рис. 4. Камера с горизонтальным углом зрения 46 градусов

4. Повороты камеры

Важным вопросом является выбор системы координат, вокруг осей которой пользователь (зритель) при просмотре может поворачивать камеру и на какие углы. Если выбрать мировую систему координат, то после нескольких поворотов пользователь, скорее всего, потеряет ориентацию, т.е. перестанет понимать, куда направлены оси этой системы и что надо сделать, чтобы направить камеру в нужную ему точку. Более удобной является локальная система координат камеры, т.к. пользователь всегда знает, где у камеры "верх", "низ", "право" и "лево". Поворот вокруг оси X будет соответствовать наклону головы вверх/вниз, а вокруг Y - ее повороту вправо/влево. Поворот вокруг оси Z соответствует наклонам головы в сторону и необходимость таких поворотов зависит от контекста задач в СВО. Для многих СВО такая операция представляется излишней.

Возможны и другие выборы системы координат. Например, в плейере 360 градусов видеохостинга Youtube текущая ориентация камеры определяется относительно начальной ее ориентации двумя углами: произвольным углом γ поворота вокруг начальной оси Y камеры и углом $\eta \in [-\pi/2, \pi/2]$ поворота вокруг нового положения оси X. Углы γ и η вычисляются и обрезаются до нужных диапазонов в каждом кадре после их изменения зрителем с помощью интерфейса управления (например, клавиатуры, мыши или джойстика). Более точно, обрезать надо только угол η следующим образом: если $\eta < -\pi/2$, то η заменяется на $-\pi/2$, а если $\eta > \pi/2$, то η заменяется на $\pi/2$. Такое управление камерой удобно в СВО, в которых определен "верх" и нет необходимости "кувыркаться". Для реализации этого интерфейса достаточно в каждом кадре отследить нажатие пользователя на клавишу (как событие в операционной системе), вычислить новые углы $\gamma = \gamma \pm \delta$ и $\eta = \eta \pm \delta$, (где δ – добавочный угол поворота), обрезать η до нужного диапа-

зона и повернуть камеру вокруг оси Y на угол γ, а затем вокруг оси X на угол η. Однако при реализации этого интерфейса с помощью графической библиотеки OpenGL вместо поворотов камеры проще выполнять повороты окружающего единичного куба с изображением. При этом в силу двойственности куб надо поворачивать на такие же углы, но в другую сторону. Таким образом, в каждом кадре устанавливается единичная модельно-видовая матрица и осуществляется поворот системы координат камеры на углы $-\gamma$ и $-\eta$ вокруг векторов, соответственно, Y и X. На рис. 5 показана система координат куба, совпадающая с системой координат камеры после установки единичной модельно-видовой матрицы. Таким образом, камера направлена в точку P (в противоположную сторону оси Z). Необходимо повернуть куб так, чтобы точка P" перешла в точку P. Первый поворот вокруг оси Y = (0.0f, 1.0f, 0.0f) осуществляется с помощью оператора glRotatef($-\gamma$, 0.0f, 1.0f, 0.0f). Он переводит точку P" в точку P', при этом ось X куба переходит в положение X'. Матрица этого поворота будет иметь вид

$$R_{Y,-\gamma}=\begin{pmatrix}\cos\gamma & 0 & -\sin\gamma\\ 0 & 1 & 0\\ \sin\gamma & 0 & \cos\gamma\end{pmatrix}.$$

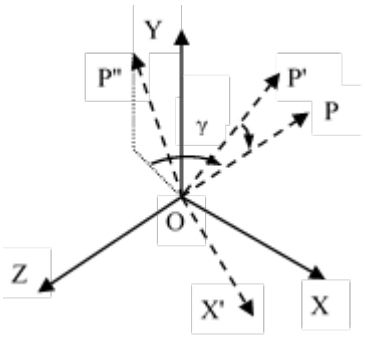


Рис. 5. Повороты системы координат

Теперь, для того, чтобы опустить точку P' до точки P, надо осуществить поворот вокруг оси X (а не вокруг оси X', т.к. $OP'\perp X$) на угол $-\eta$. Теперь ось X' имеет координаты (1.0f, 0.0f, 0.0f), а координаты оси X можно вычислить по формуле $X=R_{Y,\gamma}X'$, где $R_{Y,\gamma}$ – матрица поворота вокруг оси Y на угол γ. Расписывая это равенство, получаем

$$X=\begin{pmatrix}cos\gamma & 0 & sin\gamma\\ 0 & 1 & 0\\ -sin\gamma & 0 & cos\gamma\end{pmatrix}\begin{pmatrix}1\\0\\0\end{pmatrix}=\begin{pmatrix}cos\gamma\\0\\-sin\gamma\end{pmatrix}.$$

Таким образом, второй оператор поворота будет иметь вид

glRotatef($-\eta, Cos\gamma, 0.0f, -Sin\gamma$).

Матрица поворота вокруг произвольного вектора (x, y, z) на некоторый угол α имеет вид

$$R=\begin{pmatrix}x^2(1-c)+c & xy(1-c)-zs & xz(1-c)+ys\\ yx(1-c)+zs & y^2(1-c)+c & yz(1-c)-xs\\ xz(1-c)-ys & yz(1-c)+xs & z^2(1-c)+c\end{pmatrix},$$

где $c=\cos\alpha$, $s=\sin\alpha$. Используя это, получим матрицу для второго поворота

$$R_{X,-\eta}=\begin{pmatrix}c_\gamma^2(1-c_\eta)+c_\eta & -s_\gamma s_\eta & -c_\gamma s_\gamma(1-c_\eta)\\ s_\gamma s_\eta & c_\eta & c_\gamma s_\eta\\ -c_\gamma s_\gamma(1-c_\eta) & -c_\gamma s_\eta & s_\gamma^2(1-c_\eta)+c_\eta\end{pmatrix},$$

где $c_\gamma=\cos\gamma$, $s_\gamma=\sin\gamma$, $c_\eta=\cos\eta$, $s_\eta=\sin\eta$.

Общее преобразование будет задаваться произведением матриц поворота $R=R_{Y,-\gamma}R_{X,-\eta}$. Интересно, что такой же результат можно получить последовательным выполнением операторов

glRotatef($-\eta, 1.0f, 0.0f, 0.0f$) и

glRotatef($-\gamma, 0.0f, 1.0f, 0.0f$)

с матрицами соответственно

$$R_{X,-\eta}=\begin{pmatrix}1 & 0 & 0\\ 0 & \cos\gamma & \sin\gamma\\ 0 & -\sin\gamma & \cos\gamma\end{pmatrix}$$ и

$$R_{Y_1,-\gamma}=\begin{pmatrix}\cos\gamma & 0 & -\sin\gamma\\ 0 & 1 & 0\\ \sin\gamma & 0 & \cos\gamma\end{pmatrix}$$

(здесь второй поворот делается вокруг оси Y1 – нового положения оси Y). Хотя объяснить это затруднительно, но можно проверить равенство матриц преобразований

$$R=R_{Y_1,-\gamma}R_{X,-\eta}=R_{X,-\eta}R_{Y,-\gamma}.$$

Одновременное нажатие двух клавиш, поворот джойстика вокруг двух осей или перемещение мыши по диагонали легко отслеживается с помощью генерации событий в операционной системе. Поэтому в каждом кадре по относительному перемещению средства управления в каждом направлении можно вычислить необходимые углы поворота и применить вышеописанный алгоритм.

В СВО, имитирующих космическое пространство, все направления равнозначны, поэтому в видео, созданном в них с помощью кубической проекции, произвольные повороты вполне допустимы и даже удобны. В таких СВО можно предложить осуществлять повороты

камеры вокруг текущих положений осей ее локальной системы координат. При этом, если управляющий орган позволяет менять одновременно несколько поворотов вокруг осей камеры, то нужно зафиксировать порядок применения этих поворотов.

Установка камеры в OpenGL (а вернее, в ее расширении) осуществляется с помощью функции gluLookAt(x0, y0, z0, rx, ry, rz, ux, uy, uz), где (x0, y0, z0) - координаты позиции камеры, (r_x, r_y, r_z) – координаты произвольной точки, куда направлена камера, а (ux, uy, uz) - вектор, указывающий "верх" камеры. Наша задача состоит в том, чтобы вычислить параметры этой функции в каждом кадре просмотра видео. Так как камера всегда находится в центре куба (в этой же точке находится начало мировой системы координат), то (x_0, y_0, z_0) всегда совпадает с (0, 0, 0). Рассмотрим поворот камеры вокруг оси X на угол χ. Как видно из рис. 6, единичный вектор OP переходит в единичный вектор OR, где точка R определяет новое направление взгляда камеры и имеет координаты (0, $\sin\chi$, $-\cos\chi$). Верх камеры указывает вектор Y', но так как угол χ мал, то можно взять вектор Y=(0, 1, 0).

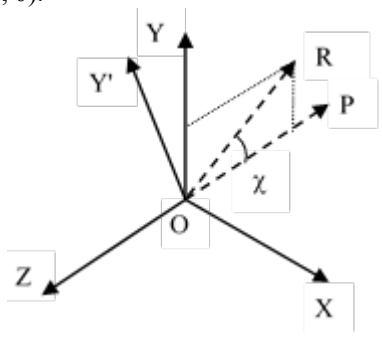


Рис. 6. Повороты камеры вокруг оси X

При повороте камеры вокруг оси Y (см. рис. 7) ее взгляд будет направлен в точку R = (–$\sin\chi$, 0, $-\cos\chi$), а верх будет показывать вектор Y = (0, 1, 0).

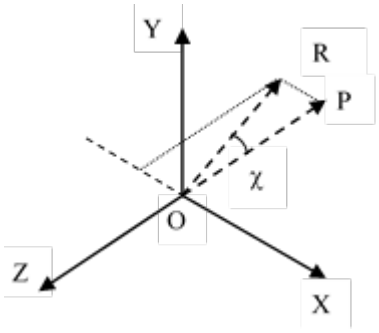


Рис. 7. Повороты камеры вокруг оси Y

При повороте камеры вокруг оси Z (см. рис. 8) направление камеры не изменится, т.е. R = –Z = (0, 0, –1). Верх камеры будет указывать вектор OU = (–$\sin\chi$, $\cos\chi$, 0).

Симуляцию приближения и удаления камеры (zoom) можно выполнять с помощью изменения угла зрения камеры при неизменных размерах области вывода изображения на экране компьютера. Камера с меньшим углом захватывает небольшую часть окружающей обстановки, поэтому при отображении в область вывода изображение будет растянуто до его размеров и объекты будут увеличены в размерах, а, значит, казаться ближе. Аналогично, при увеличении угла зрения камеры все объекты сцены будут казаться удаленными. Кроме того, при больших углах будут искажения размеров объектов изображения, описанные выше.

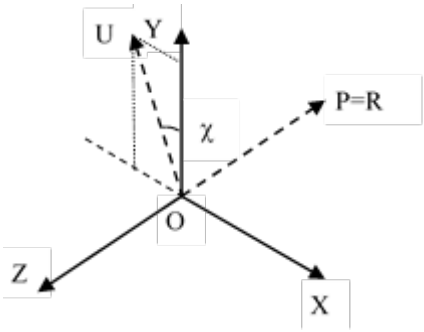


Рис. 8. Повороты камеры вокруг оси Z

5. Заключение

В данной работе рассмотрены вопросы выбора раствора камеры для просмотра видео 360 градусов, полученного с помощью кубической проекции. Показано, что при углах, больших 50 градусов, искажения изображения (растяжения его участков) при повороте камеры уже заметны для глаза. Например, при угле камеры просмотра 90º длины одного и того же участка в центре и на периферии камеры отличаются более, чем в полтора раза. Кроме того, рассмотрены алгоритмы поворотов камеры при просмотре видео 360 градусов. Эти алгоритмы реализованы в разработанном в ФГУ ФНЦ НИИСИ РАН плейере для просмотра видео 360 градусов. Проведенная апробация показала их адекватность и возможность использования в системах виртуального окружения.

Публикация выполнена в рамках государственного задания по проведению фундаментальных научных исследований (ГП 14) по теме (проекту) “34.9. Системы виртуального окружения: технологии, методы и алгоритмы математического моделирования и визуализации” (0065-2019-0012).

Camera parameters to watch 360-degree video

Mikhail Mikhaylyuk, Denis Omelchenko, Dmitry Kononov, Dmitry Loginov

Abstract. In the paper the virtual camera parameters for watching 360-degree video produced with a cubemap projection are considered. The image distortion estimates for camera rotations given different horizontal FOV angles are provided. The algorithms of virtual camera rotations during the watching 360-degree video are proposed.

Keywords: virtual environment, real time, 360-degree video, cubemap projection, virtual camera parameters, virtual camera rotations

Литература

1. Михайлюк М.В., Брагин В.И. Технологии виртуальной реальности в имитационно-тренажерных комплексах подготовки космонавтов. Пилотируемые полеты в космос, ФГБУ «НИИ ЦПК имени Ю.А.Гагарина», № 2(7), 2013 стр. 82-93.

2. Масалкин А.И., Торгашев М.А. Опыт использования систем имитации визуальной обстановки в тренажерах пилотируемых космических аппаратов. // Пилотируемые полеты в космос, Звездный городок. – 2015. – № 2. – С. 36-42.

3. М.В. Михайлюк, А.В.Мальцев, П.Ю.Тимохин, Е.В.Страшнов, Б.И. Крючков, В.М. Усов. Системы виртуального окружения для прототипирования на моделирующих стендах использования космических роботов в пилотируемых полетах. Пилотируемые полеты в космос, 2020, № 2 (35), стр. 61-75.

4. Платформа Unity https://unity.com/ru/solutions (дата обращения: 06.07.2020).

5. Chen Z., Wang X., Zhou Y., Zou L., Jiang J. Content-Aware Cubemap Projection for Panoramic Image via Deep Q-Learning. MultiMedia Modeling. MMM 2020. Lecture Notes in Computer Science, 2020, vol. 11962, pp. 304-315. DOI: 10.1007/978-3-030-37734-2_25.

6. FFmpeg. A complete, cross-platform solution to record, convert and stream audio and video. Available at: https://ffmpeg.org/, accessed 18.03.2020.

7. Timokhin P.Y., Mikhaylyuk M.V. Effective technology to visualize virtual environment using 360-degree video based on cubemap projection. CEUR Workshop Proceedings: Proc. Int. Conf. on Computing for Physics and Technology (CPT2020), Pushhino, 2020.

Командный режим управления виртуальным двуногим шагающим роботом

Е.В. Страшнов[1], И.Н. Мироненко[2], Л.А. Финагин[3]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, strashnov_evg@mail.ru;

[2] ФГУ ФНЦ НИИСИ РАН, Москва, Россия, mironenko_in@mail.ru;

[3]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, antifin@mail.ru

Аннотация. В работе рассматриваются решения для командного управления двуногим шагающим роботом в системах виртуального окружения. Для этого была создана виртуальная модель, в которой параметры динамических объектов и датчиков задаются посредством разработанного конструктора, расширяющего возможности системы 3ds Max. Предлагаемые решения для управления шагающим роботом основаны на применении технологии виртуальных пультов, в рамках которой движение в сочленениях робота задается путем изменения состояния элементов виртуального пульта. Апробация командного режима управления двуногим шагающим роботом выполнялась в комплексе системы виртуального окружения, созданном в ФГУ ФНЦ НИИСИ РАН.

Ключевые слова: двуногий шагающий робот, командный режим управления, виртуальный пульт, система виртуального окружения

1. Введение

Новые достижения в области машинного зрения, искусственного интеллекта и теории автоматического управления привели к стремительному развитию робототехники в последние годы. Активно создаются манипуляционные и мобильные роботы, которые нашли широкое применение в промышленности, медицине, космонавтике, строительстве, сельском хозяйстве и т.д. Отдельным направлением исследований является экстремальная робототехника [1], в рамках которой создаются роботы, предназначенные для выполнения работ в условиях опасных или неблагоприятных для здоровья человека. Необходимость таких работ возникает при ликвидации последствий техногенных катастроф, тушении пожаров, освоении космоса и т. д. Особое внимание в этой области уделяется способу передвижения роботов внутри помещения и при наличии завалов. Большинство колесных и гусеничных роботов неспособны свободно перемещаться в таких условиях. Поэтому повышенный интерес представляют шагающие роботы – машины, которые оставляют дискретный след на поверхности перемещения [2]. Частным случаем является двуногий шагающий робот, преимущество которого перед остальными шагающими устройствами заключается в том, что он способен работать в условиях, наиболее адаптированных для человека.

Управление двуногим шагающим роботом является сложной задачей и может осуществляться дистанционно с помощью человека-оператора или в автономном режиме, согласно заложенной программе. Отработку выполнения шагающим роботом различных задач более удобно проводить в виртуальной среде с применением виртуальной модели робота. Это позволит в дальнейшем уменьшить риск поломки робота, протестировать алгоритмы управления и сформировать правильное экспертное мнение о применимости шагающего робота для решения задач в экстремальных условиях. В связи с этим применение систем виртуального окружения для управления двуногими шагающими роботами является важной и актуальной задачей.

В данной работе предлагаются методы и подходы управления двуногими шагающими роботами в командном режиме с помощью виртуального пульта. Предлагаемое решение включает в себя разработку виртуальной модели шагающего робота, которая соответствует своему реальному прототипу антропоморфного робота Skybot F-850 [3]. Для этого был разработан конст¬руктор в системе трехмерного моделирования 3ds Max, с помощью которого модель шагающего робота создается в виде набора объектов, образующих систему шарнирно связанных тел. Такой подход позволяет задавать массо-инерционные характеристики звеньев робота, параметры исполнительных устройств (например, электроприводов) и датчиков, с помощью которых реализуется обратная связь в виртуальной среде. В свою очередь виртуальный пульт управления шагающим роботом создается в редакторе пультов и функциональных схем управления [4, 5]. Его элементами являют-

ся кнопки, джойстики, тумблеры и т.д., с которыми пользователь может взаимодействовать. Рассмотрим предлагаемые решения более подробно.

2. Трехмерная модель двуногого шагающего робота

Виртуальная модель двуногого шагающего робота (см. Рис. 1) создана в системе трехмерного моделирования 3ds Max и содержит порядка 200 тысяч полигонов. Конструкция модели обладает свойством антропоморфности, то есть имеет схожее с человеком строение и аналогичные особенности. В этой модели две руки робота служат для манипуляций с объектами, а две ноги являются педипуляторами и предназначены для перемещения робота. На голове робота установлена виртуальная камера, которая позволяет оператору осуществлять мониторинг за процессом выполнения роботом операции через шлем виртуальной реальности.

Также рассматриваемая модель включает в себя специальные объекты, предназначенные для моделирования динамики и управления роботом. Для этого на языке MaxScript был разработан конструктор плагинов, которые дополняют стандартный набор объектов системы 3ds Max. На Рис. 1 показана панель инструментов для создания специальных объектов. В модели шагающего робота задействованы следующие объекты: центры масс, шарниры, электродвигатели, аппроксимирующие контейнеры, виртуальные датчики и объект для расчета инверсной кинематики. Для каждого такого объекта задается свой набор параметров. Например, для объекта «Шарнир» (см. Рис. 1) основными параметрами являются его степени свободы, ограничения на относительное движение в шарнире, сила или момент трения и т.д.

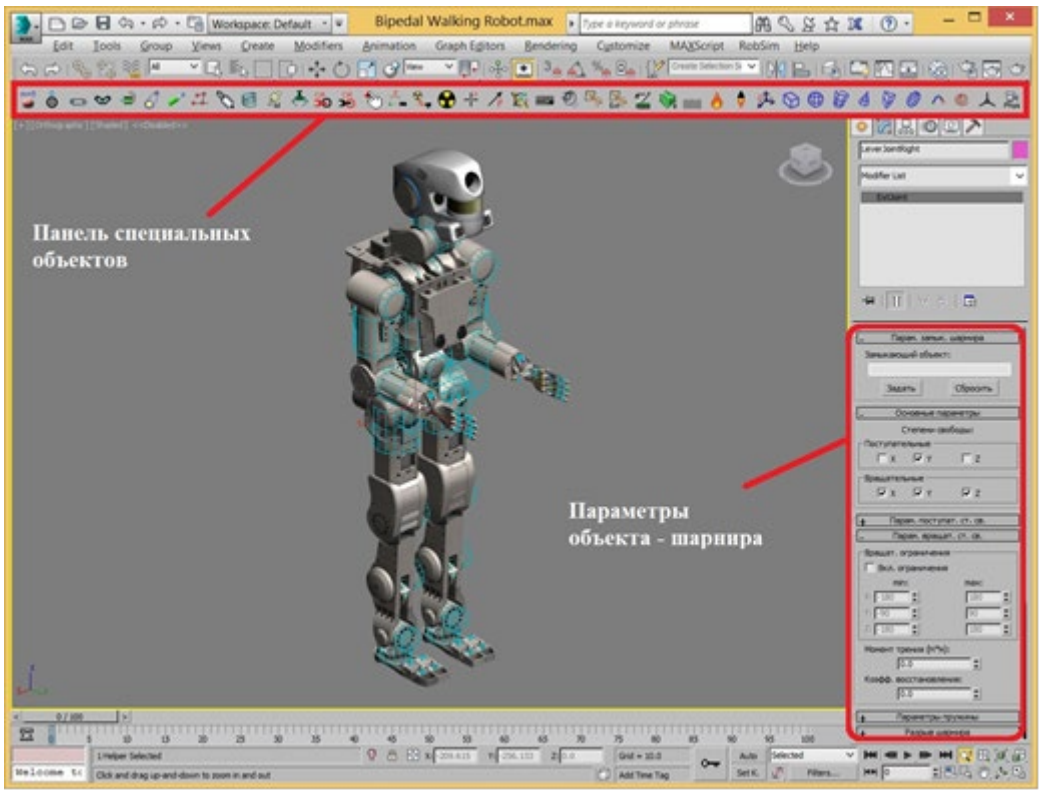


Рис. 1. Трехмерная модель шагающего робота в системе 3ds Max

Модель робота представляет собой иерархию, в которой звенья соединены с помощью шарниров и электродвигателей. На Рис. 2 показаны заданные ограничения, накладываемые на углы поворотов в сочленениях робота. Звено в модели определяется путем присоединения к нему объекта «Центр масс», параметрами которого являются масса и главные моменты инерции. Объект «Электродвигатель» является исполнительным устройством робота и в данной модели рассматривается двух типов: с одной вращательной и с одной поступательной степенью свободы. В качестве параметров используются паспортные параметры двигателя и редук-

тора, которые используются при расчете управляемого момента или силы [6]. С помощью объекта «Шарнир» задается кинематическая пара соединения двух звеньев. Захват робота представляет собой механизм «Белградская рука» [7], в котором общее движение пальца осуществляется только за счет одного электропривода.

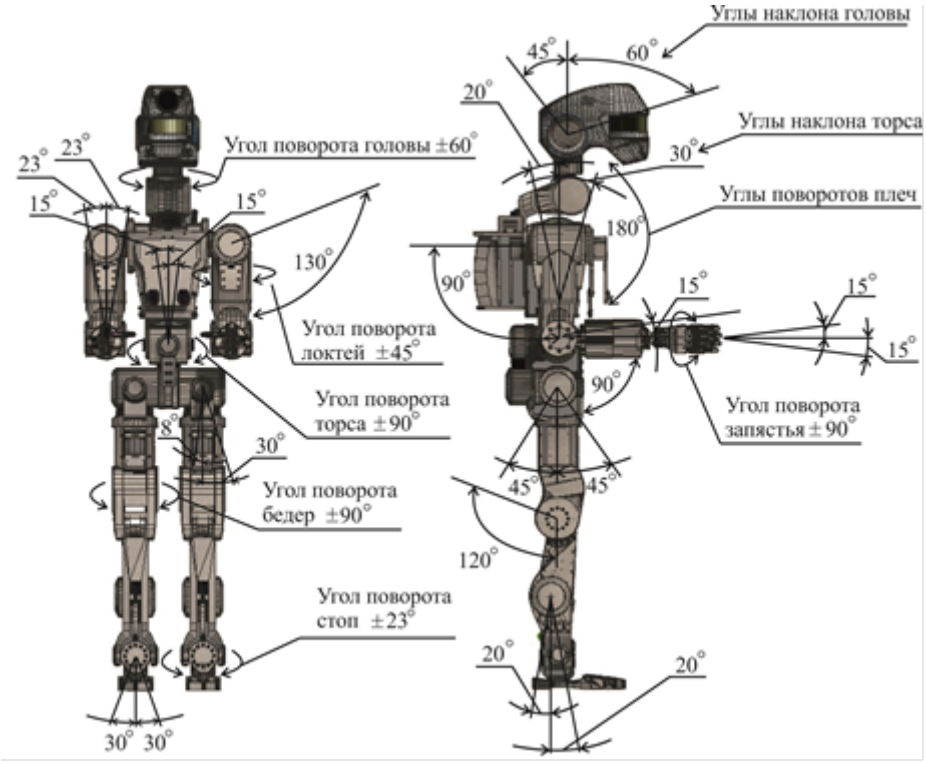


Рис. 2. Структура модели шагающего робота

Аппроксимирующие контейнеры (параллелепипеды, сферы, цилиндры и т.п.) окружают геометрию звеньев робота и используются для определения и разрешения коллизий робота с объектами виртуального окружения.

С помощью виртуальных датчиков, установленных на модели робота, реализуется обратная связь в виртуальной среде. В данной модели были задействованы датчики положения, дальномеры, сило-моментного очувствления, угловых скоростей, гироскопы и поворота двигателей. Датчики положения находятся на концах рук и ног робота и позволяют вычислить положения звеньев относительно некоторой фиксированной системы координат. Дальномеры измеряют расстояния до ближайшего объекта и расположены на стопах робота. Такие датчики предназначены для реализации адаптивного перемещения робота в зависимости от сложности поверхности и наличия препятствий. Датчики сило-моментного очувствления вычисляют силы и моменты, которые действуют на робота при его взаимодействии с виртуальным окружением. В свою очередь, датчики угловой скорости и гироскопы (датчики ориентации) измеряют угловую скорость и ориентацию звеньев робота, соответственно. Вычисления производятся относительно локальной системы координат звена, где установлен датчик. Наконец, датчик поворота двигателя выбирается опционально в параметрах двигателя и измеряет его угол поворота.

Объект для расчета инверсной кинематики служит для управления движением ног шагающего робота. На выходе этого объекта вычисляются углы поворотов двигателей, которые необходимо обеспечить для реализации заданного движения робота.

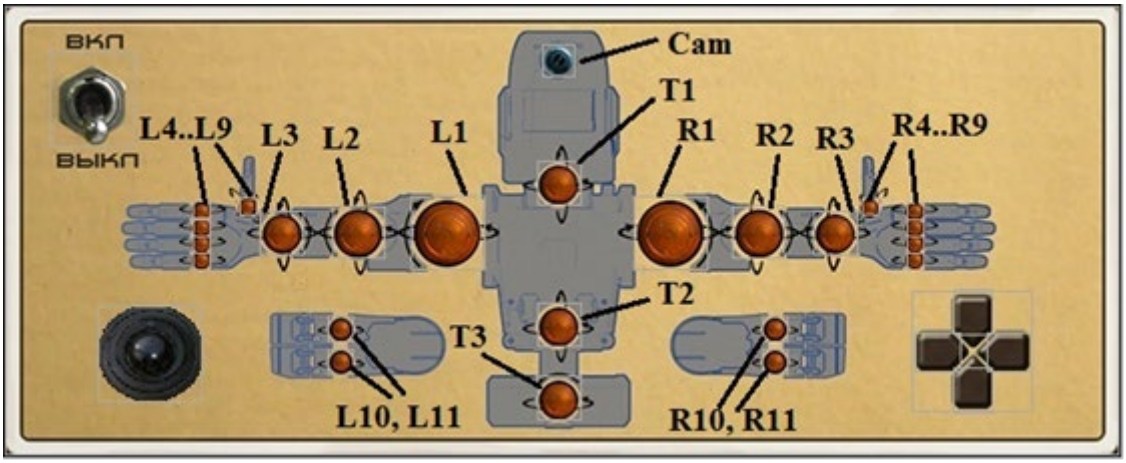


Рис. 3. Виртуальный пульт управления шагающим роботом

3. Виртуальный пульт управления

В данной работе предлагается подход, в котором управление шагающим роботом осуществляется с помощью виртуального пульта. Для этого в специальном редакторе был создан виртуальный пульт (см. Рис. 3), который представляет собой двумерное схематичное изображение шагающего робота с визуальными элементами управления типа тумблера, кнопок и джойстика. С этими элементами пользователь может взаимодействовать путем нажатия и перемещения компьютерной мыши. Каждому элементу пульта соответствует свой набор изображений для различных состояний, что позволяет создать анимацию работы соответствующего органа управления. Тумблер или двухпозиционный переключатель имеет два состояния (активное и пассивное), каждому из которых соответствует свое изображение. Кнопка включает в себя несколько разновидностей в зависимости от поведения при ее нажатии. В качестве изображений используются состояния, когда кнопка разжата, нажата и подсвечивается. В свою очередь, джойстик создается из многослойного изображения, где каждый слой имеет собственное смещение, которое пропорционально перемещению указателя мыши. Такой подход позволяет создать анимацию работы трехмерного джойстика. Опишем функциональность созданного пульта более подробно. С помощью тумблера, который находится в левой верхней части пульта, осуществляется включение и выключение пульта. Кнопки на пульте соответствуют двигателям в сочленениях робота, где «Li» – кнопки для двигателей левой руки и пальцев левой ноги, «Ri» – для двигателей правой руки и пальцев правой ноги, «Tj» – для двигателей шеи и торса, $i = \overline{1,11}$, $j = \overline{1,3}$. Стрелками вокруг кнопок на пульте указаны доступные направления вращений в рассматриваемом сочленении. При нажатии на кнопку управление двигателем осуществляется путем отклонения от нейтрального положения 3-х степенного джойстика, который располагается в левой нижней части пульта. Исходя из требования эргономичности, интерфейс виртуального пульта реализован таким образом, что одна и та же кнопка отвечает за управление несколькими двигателями в зависимости от направления смещения джойстика. Например, в локте правой руки робота располагаются три двигателя. Тогда нажатием кнопки «R2» и смещении джойстика вдоль его оси X управление осуществляется первым двигателем, при смещении вдоль его оси Y – вторым двигателем, а при повороте джойстика – третьим двигателем. Нажатием на кнопку «Cam» пульта осуществляется переход к изображению, получаемому от виртуальной камеры, установленной на голове робота. Кроме того, в правой нижней части пульта находятся кнопки, предназначенные для перемещения робота. Эти кнопки задают направления движения робота, при котором движение стоп ног осуществляется вдоль заранее рассчитанных траекторий.

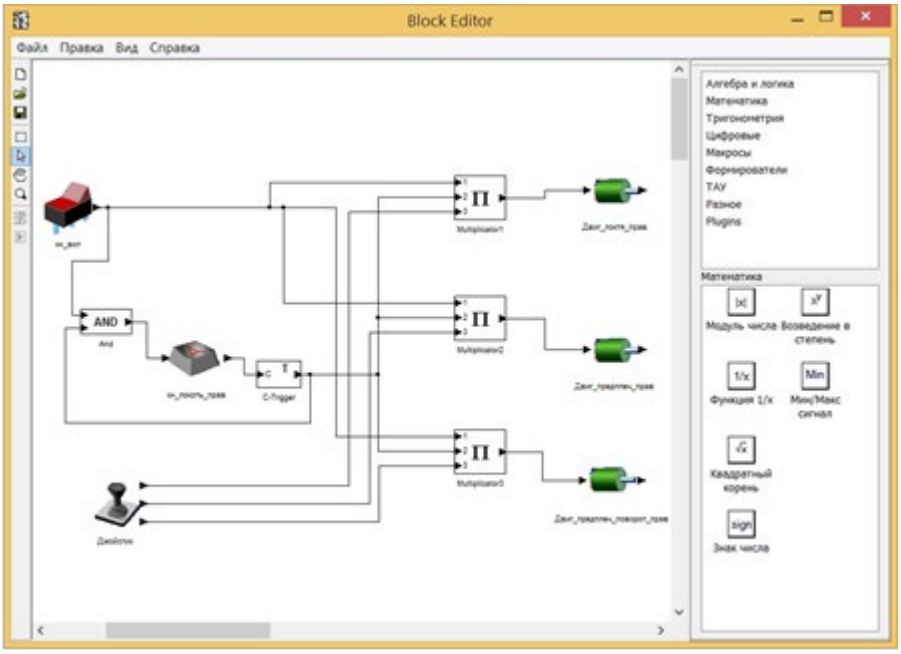


Рис. 4. Пример функциональной схемы в редакторе

4. Командный режим управления шагающим роботом

Созданный виртуальный пульт был применен для реализации командного режима управления двуногим шагающим роботом в программном комплексе системы виртуального окружения, разработанном в ФГУ ФНЦ НИИСИ РАН. Командный режим заключается в том, что оператор нажатием на кнопки выбирает один или несколько двигателей и с помощью джойстика осуществляет управление в выбранном сочленении. Работа пульта подчиняется в соответствие созданной функциональной схеме, в которой блоки элементов пульта связаны с блоками исполнительных устройств робота. При таком подходе воздействия оператора на элементы пульта (например, смещение джойстика) передаются в функциональную схему, которая выполняет расчет напряжений, подаваемых на электроприводы робота. На Рис. 4 показан фрагмент такой схемы, предназначенной для управления двигателями в локтевом суставе правой руки робота. Эта схема содержит блоки элементов виртуального пульта, электродвигателей и библиотеки редактора функциональных схем. Блок тумблера «кн_вкл» имеет один выход, который может принимать значение 0 или 1. Схема вычисляет напряжения только в том случае, если на выходе этого блока формируется 1. Блок кнопки «кн_локоть_прав» имеет один вход и один выход. При подаче 1 на вход этого блока выполняется включение подсветки кнопки. На выходе блока 1 формируется в том случае, если пользователь нажал на кнопку пульта. Блок «C-Trigger» хранит состояние нажатия кнопки. Значение на выходе меняется с 0 и 1 и обратно при подаче 1 на вход блока. Блок «Джойстик» имеет три выхода, которые соответствуют смещению джойстика в пределах от -1 до 1 вдоль осей X, Y и его повороте. Эти смещения задают нормированные напряжения, которые передают¬ся на входы блоков исполнительных устройств (электродвигателей), куда входят «Двиг_локтя_прав», «Двиг_предплеч_прав» и «Двиг_предплеч_поворот_прав».

Апробация командного режима управления шагающим роботом проводилась на примере выполнения роботом операции открытия двери виртуальной сцены интерьера помещения (см. Рис. 5). Путем нажатия на кнопки и смещения джойстика пульта был осуществлен захват и поворот ручки двери с помощью манипулятора (правой руки) робота. Затем при управлении двигателем плеча было выполнено открытие роботом двери. Проведенная апробация показала адекватность предложенных методов командного управления двуногими шагающими роботами для систем виртуального окружения.

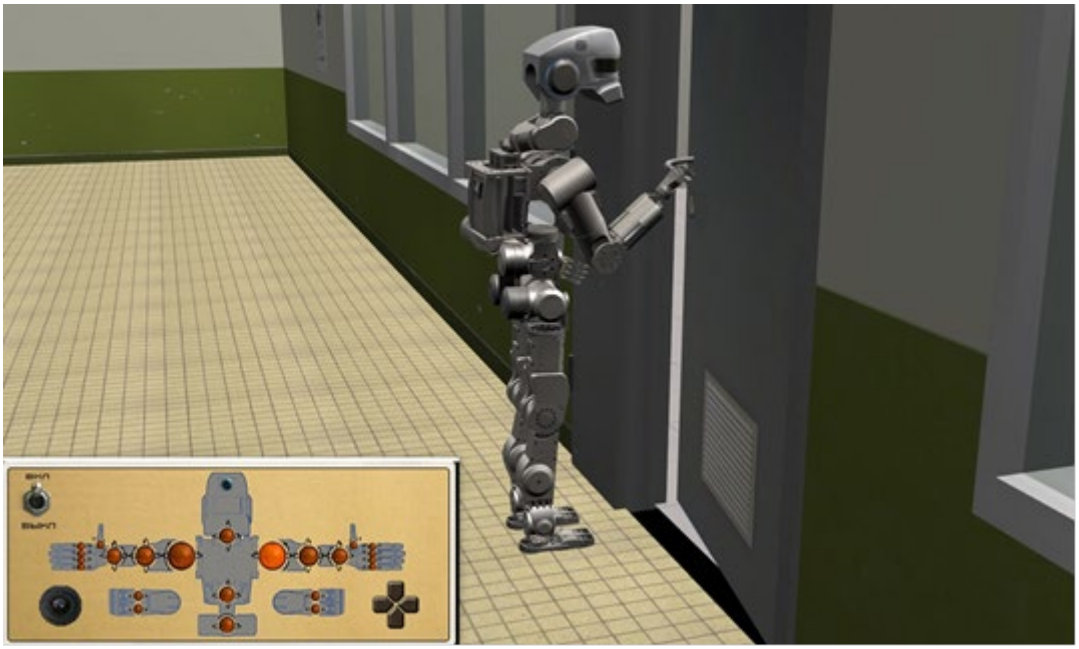


Рис. 5. Моделирование управления роботом в командном режиме

5. Заключение

В данной работе предложены методы и подходы, основанные на применении виртуальных пультов для управления виртуальной моделью двуногого шагающего робота в командном режиме. Предлагаемые решения расширяют возможности систем виртуального окружения и предоставляют удобное средство для отработки выполнения роботом различных технологи¬ческих операций. В дальнейшем предполагается использование созданной моде¬ли шагающего робота и виртуального пульта управления для тестирования моделирования движения робота с учетом его статического и динамического равновесия.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-07-00371.

Command mode for virtual bipedal walking robot control

Evgeny Strashnov, Irina Mironenko, Leonid Finagin

Abstract. The paper discusses solutions for command control of a bipedal walking robot in virtual environment systems. For this, a virtual model was created, in which the parameters of dynamic objects and sensors are set using a developed constructor that extends the capabilities of the 3ds Max. The proposed solutions for a walking robot control are based on the use of virtual remote control technology, in which the movement in the robot links is set by changing the state of the virtual remote controller elements. The approbation of the command mode for the bipedal walking robot control was carried out in the complex of the virtual environment system, created at the SRISA RAS.

Keywords: bipedal walking robot, command control mode, virtual remote controller, virtual environment system

Литература

1. Е.И. Юревич. Основы робототехники: учеб. пособие. 4-е изд., перераб. и доп. Спб., БХВ-Петербург, 2017.

2. В.Е. Павловский. О разработках шагающих машин. «Препринты ИПМ им. М.В. Келдыша»,

Москва, 2012.

3. И. Афанасьев. «Федор» летит на МКС. «Русский космос», 2019, № 9, 2-9.

4. М.В. Михайлюк. Двумерные виртуальные пульты управления в тренажерных комплексах. «Программная инженерия», 2014, № 5, 20-25.

5. М.В. Михайлюк, М.А. Торгашев. Визуальный редактор и модель расчета функциональных схем для имитационно-тренажерных комплексов. «Программные продукты и системы», 2014, № 4, 10-15.

6. Е.В. Страшнов, М.А. Торгашев. Моделирование динамики электроприводов виртуальных роботов в имитационно-тренажерных комплексах. «Мехатроника, автоматизация, управления», Т. 17 (2016), № 11, 762-768.

7. Э. Накано. Введение в робототехнику / перевод с японского под ред. А.И. Логинов, А.М. Филатов. М., Мир, 1988.

Повышение производительности векторного кода с помощью мониторинга плотности масок в векторных инструкциях

А.А. Рыбаков[1], А.Д. Чопорняк[2]

[1]МСЦ РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, rybakov@jscc.ru;

[2]МСЦ РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, adc@jscc.ru

Аннотация. Одним из ключевых факторов, влияющих на эффективность векторного кода, создаваемого с помощью AVX-512, является недостаточная плотность масок в векторных инструкциях. Это свидетельствует о том, что возможности векторных инструкций использованы не в полной мере, и во время выполнения кода обрабатывается только часть элементов векторов. Данная статья посвящена работе по созданию библиотеки многоверсионного кода, с помощью которой можно генерировать как векторный код, так и обычный код с эмуляцией векторных инструкций и возможностью мониторинга их отдельных характеристик, и в частности плотности масок. Результаты такого мониторинга можно использовать для повышения производительности векторного кода.

Ключевые слова: векторизация, повышение производительности, векторная маска, функции-интринсики, AVX-512.

1. Введение

Векторизация программного кода с использованием инструкций AVX-512 является низкоуровневой оптимизацией, с помощью которой можно добиться кратного увеличения производительности приложений [1]. Этому способствуют многочисленные особенности набора инструкций AVX-512 [2], и одной из основных особенностей является наличие масочных векторных инструкций, позволяющих применять операцию не ко всем элементам векторов-аргументов, а только к некоторым из них. Данная особенность является уникальной и позволяет применять векторизацию для сложного программного контекста, содержащего разветвленное управление, гнезда циклов и вызовы функций [3]. Однако использование масочных векторных инструкций может стать причиной снижения производительности кода, если плотность масок слишком низкая (то есть когда векторная инструкция обрабатывает не все элементы векторов, а только малую их часть, то смысл векторизации кода пропадает). Данный эффект снижения производительности особенно явно проявляется при использовании векторизации для циклов с нерегулярным числом итераций [4]. Для сглаживания негативных эффектов от снижения плотности масок в векторных инструкциях требуются возможности по мониторингу использования масок в процессе выполнения результирующего кода. При этом недостаточно просто численного показателя о средней плотности векторного кода или загрузке масок (в качестве примеров можно рассматривать показатели efficiency of vectorization, SIMD instructions per cycle, mask usage инструментов Intel VTune Amplifier и Intel Advisor [5,6]), требуется именно сбор статистики, по которой можно провести анализ плотности масок в векторных инструкциях в зависимости от внешних условий. Сбор такой статистики может быть обеспечен путем использования эмуляции векторных инструкций с параллельным накоплением произвольной информации, которая далее может быть проанализирована. Для реализации этой задачи выполняется разработка библиотеки создания многоверсионного кода, с помощью которой можно создавать как векторный код, использующий инструкции AVX-512, так и псевдовекторный код, в котором векторные инструкции эмулируются с помощью специальных классов, которые могут накапливать статистику времени выполнения.

Другой причиной создания библиотеки многоверсионного кода являются вопросы переносимости векторного кода. Во время оптимизации программ зачастую приходится сталкиваться с невозможностью компилятора автоматически векторизовать тот или иной фрагмент. Это может быть связано с разными причинами, в частности с недостатком у компилятора информации об отсутствии зависимостей между операциями. В этом случае на помощь приходят специальные функции-интринсики, которые могут быть использованы путем подключения библиотеки Intel <immintrin.h> [7]. Данные

функции-инстринсики являются обертками над векторными инструкциями и в процессе компиляции раскрываются в частности в инструкции AVX-512 (некоторые интринсики могут раскрываться в последовательности команд или даже в библиотечные вызовы). Конечно, если в коде программы используется такой подход, то программа сразу же становится непереносима на аппаратные платформы, в которых отсутствует поддержка тех инструкций, интринсики для которых были использованы.

Таким образом, создаваемая библиотека предназначена для генерации текста программы в трех режимах: режим генерации векторного кода для целевой аппаратной платформы с возможностью выбора допустимого набора векторных инструкций, режим эмуляции векторных инструкций для переносимости программы, режим сбора статистики для анализа производительности (что также выполняется с помощью эмуляции векторных инструкций). Схема получения результирующего исполняемого файла из исходного текста программы и библиотеки генерации многоверсионного кода показана на рис. 1.

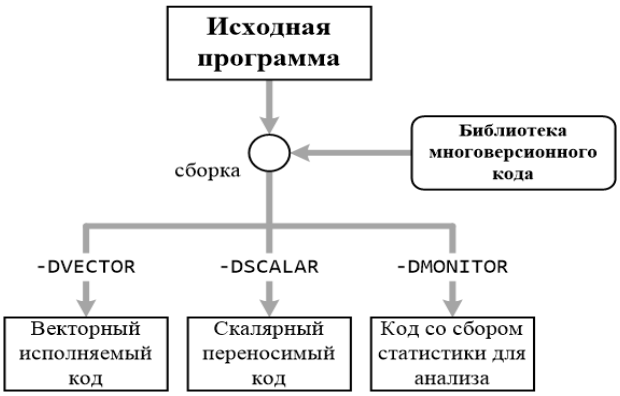


Рис. 1. Схема использования библиотеки многоверсионного кода

.

В данной статье на примере векторизации фрагмента кода из практической задачи рассмотрим влияние анализа плотности масок в векторных инструкциях на эффективность результирующего кода. В качестве примера будем использовать одну небольшую функцию из состава римановского решателя. Вопросы векторизации римановского решателя, в том числе и с помощью инструкций AVX-512, подробно изложены в работах [8,9].

2. Векторизация плоского цикла

Вначале кратко остановимся на понятии плоского цикла и подходе к его векторизации. Плоским циклом будем называть конструкцию следующего вида:

```
for (int i = 0; i < N; i++)
{
    <block(i)>
}
```

При этом тело цикла представляет собой блок вычислений со следующими свойствами. Во-первых, все обращения в память имеют вид x[i], то есть это обращения к некоторым массивам, при этом индекс массива совпадает с номером итерации плоского цикла, а также все массивы, встречающиеся внутри тела плоского цикла, не пересекаются по памяти. Данное требование к обращениям в память приводит к тому, что итерации плоского цикла становятся независимыми друг от друга, они могут выполняться в любом порядке, а значит и параллельно. Во-вторых, будем рассматривать только работу с вещественными числами одинарной точности (тип данных float). В один 512-битный векторный регистр помещается 16 таких элементов данных. Наконец, в-третьих, без ограничения общности будем считать, что количество итераций плоского цикла равно 16, в противном случае такой цикл всегда можно разбить на несколько более мелких циклов.

Такие плоские циклы представляют собой удобный контекст для векторизации, и в большинстве случаев они могут быть векторизованы с помощью векторных инструкций AVX-512 с помощью перевода тела цикла в предикатное представление и замены скалярных инструкций векторными аналогами, реализованными с помощью функций-интринсиков [10].

Многие практические вычислительные задачи состоят из выполнения однотипных вычислений, применяемых к разным наборам данных. Условно одну такую задачу можно описать как $F(a, b, c, ...) \rightarrow (x, y, z, ...)$, то есть к

некоторому набору входных параметров применяется функция F, выдающая в качестве результата набор выходных данных. Как уже говорилось, вызов данной функции F происходит многократно, при этом каждый раз используются различные наборы входных и выходных данных. Последовательность проведения вычислений может быть изменена, и вместо многих вызовов функции, работающей с набором параметров, можно рассмотреть другую функцию, работающую с массивами параметров в виде: G(a[], b[], c[], ...) → (x[], y[], z[], ...). Конечно внутри данной функции G должен быть организован цикл, на каждой итерации (с номером i) которого будет вызываться функция F со следующими наборами параметров: F(a[i], b[i], c[i], ...) → (x[i], y[i], z[i], ...). Нетрудно видеть, что такой цикл является плоским, а значит может быть легко векторизован.

Векторизация точного римановского решателя выполняется по описанной схеме. Детали реализации можно найти в [8], а в данной статье мы в качестве иллюстрации рассмотрим одну из функций, векторизация которой является наглядным примером важности анализа плотности масок векторных инструкций.

3. Мониторинг плотности масок

Итак, в качестве примера рассмотрим простую функцию prefun, входящую в состав римановского решателя (см. рис. 2). Данная функция содержит один оператор if-else, ветками которого являются блоки вычислений, содержащие простые арифметические операции, а также библиотечные функции pow и sqrt.

```
01 void prefun(float *f, float *fd,
02              float p, float dk, float pk, float ck)
03 {
04     if (p <= pk)
05     {
06         *f = G4 * ck * (pow(p/pk, G1) - 1.0);
07         *fd = (1.0 / (dk * ck)) * pow(p/pk, -G2);
08     }
09     else
10     {
11         float ak = G5 / dk;
12         float bk = G6 * pk;
13         float qrt = sqrt(ak / (bk + p));
14
15         *f = (p - pk) * qrt;
16         *fd = (1.0 - 0.5 * (p - pk) / (bk + p)) * qrt;
17     }
18 }
```

Рис. 2. Исходный вид функции prefun.

Таким образом, в функции содержится два линейных участка, которые должны выполняться под противоположными предикатами (p ≤ pk) и !(p ≤ pk). Внутри рассматриваемых линейных участков производятся арифметические вычисления, использующие входные параметры и глобальные константы (схематически граф потока управления функции prefun, а также графы зависимостей линейных участков данной функции изображены на рис. 3).

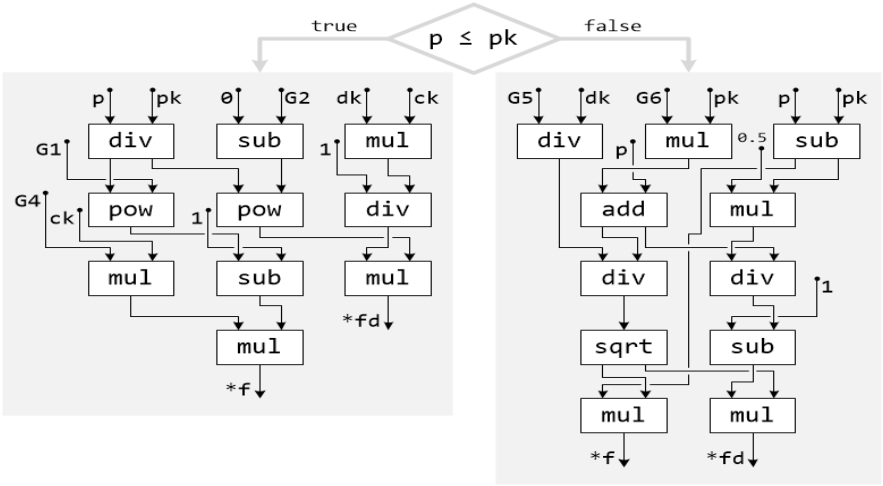


Рис. 3. Представление тела функции prefun в виде графа.

Для векторизации рассматриваемого программного контекста выполним объединение 16 вызовов функции prefun в один вызов функции prefun_16, которая вместо скалярных аргументов f, fd, p, dk, pk, ck оперирует с массивами соответствующих параметров fs, fds, ps, dks, pks, cks. При векторизации кода скалярные операции заменяются векторными аналогами, а скалярное условие заменяется на получение и использование векторной маски. При этом происходит слияние обеих ветвей исполнения под противоположными предикатами, векторные инструкции оперируют с векторами, содержащими по 16 элементов типа float, и с векторными масками, содержащими по 16 логических значений. На рис. 4 представлены две версии результирующего кода, полученные в двух разных режимах. Слева показан код, сгенерированный в режиме -DMONITOR, в котором 512-битные вектора и векторные маски представлены специальными библиотечными классами VectorF и Mask16, выполняющими операции над векторами с помощью эмуляции. Справа на рисунке показан результирующий код, сгенерированный в режиме -DVECTOR и использующий встроенные типы __m512 и __mmask16 и функции-интринсики для реализации векторных операций. Из рис. 4 можно заметить, что обе версии кода логически соответствуют друг другу.

```
01 void prefun_16(float *fs, float *fds,
02                 float *ps, float *dks, float *pks, float *cks)
03 {
04     VectorF f, fd, p, dk, pk, ck;
05     Mask16 cond, ncond;
06     VectorF z, g1, g2, g4, g5, g6, one, half;
07     VectorF t1, t2, t3, t4, t5;
08
09     // Инициализация необходимых векторов
10     ...
11
12     VectorF::CmpLE(nullptr, &p, &pk, &cond);
13     Mask16::Not(&cond, &ncond);
14
15     VectorF::Div(nullptr, &cond, &p, &pk, &t1);
16     VectorF::Pow(nullptr, &cond, &t1, &g1, &t2);
17     VectorF::Sub(nullptr, &cond, &t2, &one, &t3);
18     VectorF::Mul(nullptr, &cond, &g4, &ck, &t4);
19     VectorF::Mul(&f, &cond, &t4, &t3, &f);
20     VectorF::Sub(nullptr, &cond, &z, &g2, &t2);
21     VectorF::Pow(nullptr, &cond, &t1, &t2, &t3);
22     VectorF::Mul(nullptr, &cond, &dk, &ck, &t4);
23     VectorF::Div(nullptr, &cond, &one, &t4, &t5);
24     VectorF::Mul(&fd, &cond, &t5, &t3, &fd);
25
26     VectorF::Div(nullptr, &ncond, &g5, &dk, &t1);
27     VectorF::Mul(nullptr, &ncond, &g6, &pk, &t2);
28     VectorF::Add(nullptr, &ncond, &t2, &p, &t3);
29     VectorF::Div(nullptr, &ncond, &t1, &t3, &t4);
30     VectorF::Sqrt(nullptr, &ncond, &t4, &t4);
31     VectorF::Sub(nullptr, &ncond, &p, &pk, &t1);
32     VectorF::Mul(&f, &ncond, &t1, &t4, &f);
33     VectorF::Mul(nullptr, &ncond, &half, &t1, &t2);
34     VectorF::Div(nullptr, &ncond, &t2, &t3, &t1);
35     VectorF::Sub(nullptr, &ncond, &one, &t1, &t2);
36     VectorF::Mul(&fd, &ncond, &t2, &t4, &fd);
37
38     f.Store(fs);
39     fd.Store(fds);
40 }
```

```
01 void prefun_16(float *fs, float *fds,
02                 float *ps, float *dks, float *pks, float *cks)
03 {
04     __m512 f, fd, p, dk, pk, ck;
05     __mmask16 cond, ncond;
06     __m512 z, g1, g2, g4, g5, g6, one, half;
07     __m512 t1, t2, t3, t4, t5;
08
09     // Инициализация необходимых векторов
10     ...
11
12     cond = _mm512_cmp_ps_mask(p, pk, _MM_CMPINT_LE);
13     ncond = ~cond;
14
15     t1 = _mm512_mask_div_ps(z, cond, p, pk);
16     t2 = _mm512_mask_pow_ps(z, cond, t1, g1);
17     t3 = _mm512_mask_sub_ps(z, cond, t2, one);
18     t4 = _mm512_mask_mul_ps(z, cond, g4, ck);
19     f = _mm512_mask_mul_ps(f, cond, t4, t3);
20     t2 = _mm512_mask_sub_ps(z, cond, z, g2);
21     t3 = _mm512_mask_pow_ps(z, cont, t1, t2);
22     t4 = _mm512_mask_mul_ps(z, cond, dk, ck);
23     t5 = _mm512_mask_div_ps(z, cond, one, t4);
24     fd = _mm512_mask_mul_ps(fd, cond, t5, t3);
25
26     t1 = _mm512_mask_div_ps(z, ncond, g5, dk);
27     t2 = _mm512_mask_mul_ps(z, ncond, g6, pk);
28     t3 = _mm512_mask_add_ps(z, ncond, t2, p);
29     t4 = _mm512_mask_div_ps(z, ncond, t1, t3);
30     t4 = _mm512_mask_sqrt_ps(z, ncond, t4);
31     t1 = _mm512_mask_sub_ps(z, ncond, p, pk);
32     f = _mm512_mask_mul_ps(f, ncond, t1, t4);
33     t2 = _mm512_mask_mul_ps(z, ncond, half, t1);
34     t1 = _mm512_mask_div_ps(z, ncond, t2, t3);
35     t2 = _mm512_mask_sub_ps(z, ncond, one, t1);
36     fd = _mm512_mask_mul_ps(fd, ncond, t2, t4);
37
38     _mm512_store_ps(fs, f);
39     _mm512_store_ps(fds, fd);
40 }
```

Рис. 4. Генерация исполняемого кода в режимах -DMONITOR (слева) и -DVECTOR (справа).

Описанное представление функции prefun_16 в векторном виде приводит к ее ускорению в 5,8 раз на микропроцессорах с поддержкой AVX-512 по сравнению с невекторизованным оригиналом. Однако заметим, что в векторизованной версии содержится один общий линейный участок, являющийся объединением двух линейных участков, выполняющихся с использованием противоположных векторных масок cond и ncond. Это приводит к потере производительности в силу низкой плотности масок (в среднем каждая из них заполнена на 50%).

К сожалению данный факт является принципиально узким местом при использовании векторизации программного контекста с разветвленным управлением. Без использования дополнительной информации о профиле исполнения улучшить ситуацию не представляется возможным, и при слиянии разных ветвей исполнения производительность результирующего кода падает тем сильнее, чем выше вложенность условий [11].

4. Результаты мониторинга

Для анализа возможности повышения производительности получившегося кода в режиме эмуляции векторных инструкций внутри классов VectorF и Mask16 реализован функционал по сбору статистики плотности использованных масок (в данном случае анализировалась плотность маски cond). Вначале был осуществлен сбор статистики для случайных наборов входных данных, результат этой статистики приведен на рис. 5. На диаграмме видно распределение плотности маски, которое имеет нормальный характер. Это вполне ожидаемо, так как при случайных входных данных значение предиката ($p \leq pk$) также является случайным (0 или 1 с равной вероятностью), а плотность маски это количество всех ее истинных элементов, что можно трактовать как сумму ее единичных битов.

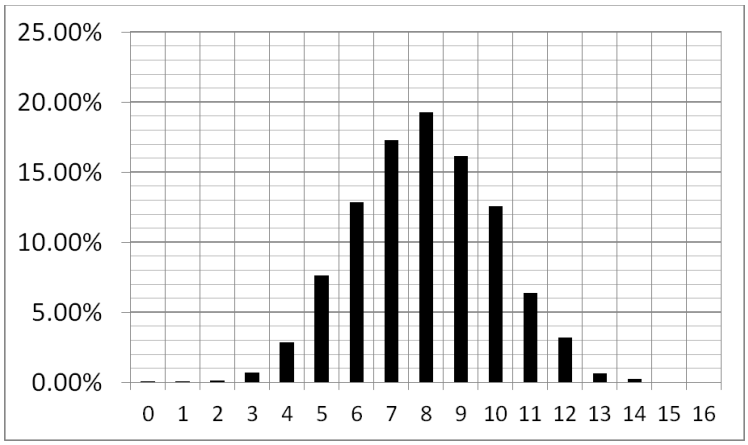


Рис. 5. Распределение плотности маски cond для случайно сгенерированных входных данных функции prefund

.

Однако наборы входных данных для физических задач имеют другую природу, и нельзя полагать, что эти наборы данных являются полностью независимыми и случайными. Если рассматривать задачу Римана о распаде произвольного разрыва, то в ней для каждого набора скалярных параметров рассматривается пара соседних ячеек расчетной сетки, для которых производится расчет газодинамических параметров на их общей границе [12]. При объединении нескольких вызовов функции римановского решателя в один вызов, обрабатывающий массивы входных параметров, с целью векторизации, наборы параметров, обрабатываемых на соседних итерациях векторизуемого плоского цикла, оказываются достаточно близкими. Можно условно считать, что каждый параметр внутри векторизуемого цикла изменяется достаточно медленно. Если рассматривать наш пример с функцией prefun_16, то можно сделать предположение, что значения массива ps изменяются медленно при изменении индекса i, аналогично значения массива pks изменяются медленно при изменении индекса i. Из этого можно сделать вывод, что также медленно изменяется и значение предиката ($ps[i] \leq pks[i]$). Но предикат это логическая величина, которая может принимать только значения 0 или 1. Это означает только то, что с большой вероятностью на продолжительных интервалах изменения индекса i значение предиката ($ps[i] \leq pks[i]$) остается константным. При векторизации кода предикаты объединяются в маски, таким образом, интервалы с константными предикатами будут формировать маски со значениями 0x0 или 0xFFFFFF (на рис. 6 представлена иллюстрация данного свойства задач физических расчетов). Конечно, если линейный участок в векторизованном коде часто попадает под маску со значением 0x0, то проверка маски на пустоту перед выполнением имеет смысл. Описанное свойство повышенной частоты нулевых и полных масок в векторизованном коде характерно для физических расчетных задач, таких как задачи газовой динамики, механики твердого тела, молекулярной динамики. И наоборот, это свойство не выполняется для дискретных задач, например задач сортировки, кодирования, задач комбинаторики и других.

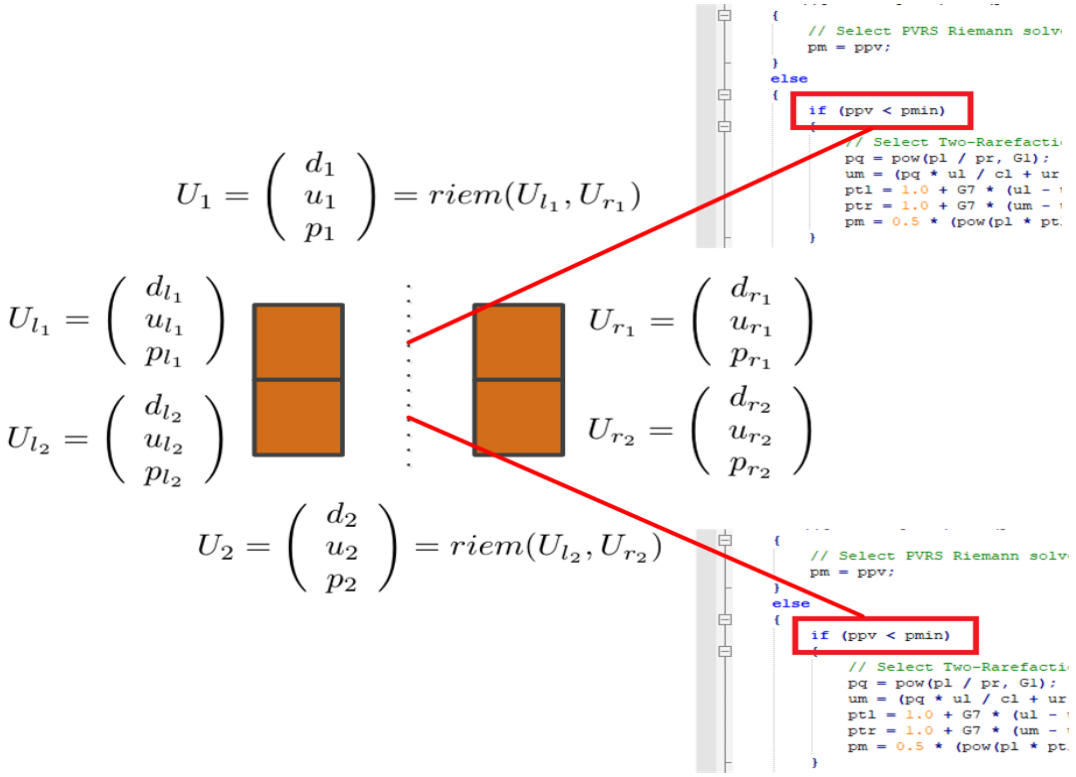


Рис. 6. Иллюстрация высокой вероятности совпадения предикатов для соседних итераций плоского цикла в физических расчетах.

Для рассматриваемой в данной статье функции prefun был осуществлен мониторинг плотности маски cond при использовании реальных входных данных, полученных при расчетах течения газа. Диаграмма с распределением плотности маски представлена на рис. 7, и данное распределение не имеет ничего общего с распределением, полученным на случайных входных данных. На диаграмме можно наблюдать почти 50% долю нулевых масок и превышающую 10% долю полных масок.

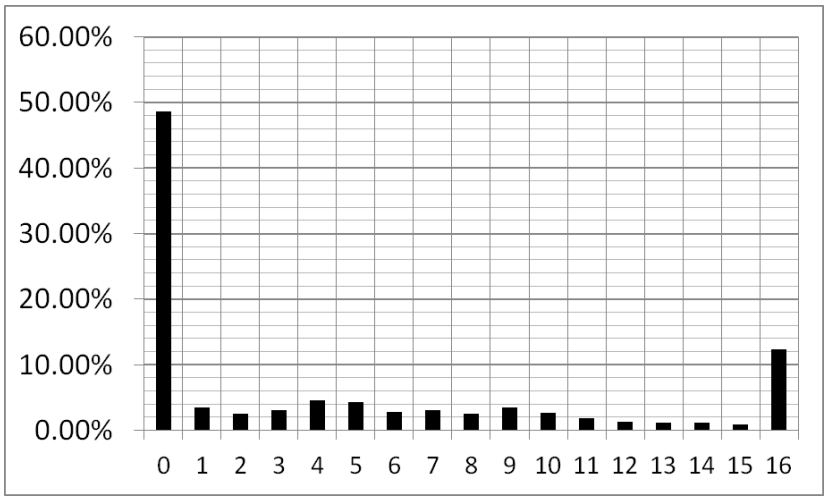


Рис. 7. Распределение плотности маски cond для реальных расчетных данных.

По результатам мониторинга соответствующие участки кода, выполняемые под масками cond и ncond, были помещены под условия if (cond) и if (ncond) соответственно, что позволило добиться дополнительного ускорения результирующего кода в 1,7 раз. Итоговое ускорение функции prefun в результате векторизации составило 9,9 раз.

Заметим, что рассматриваемый пример является тривиальным случаем, содержащим все-

го один условный оператор. Как правило, векторизуемые функции содержат множество условий. Добавление проверок для всех линейных участков в таких функциях приводит к деградации производительности, поэтому с помощью мониторинга плотности масок необходимо отыскивать те ветки, помещение под условие которых действительно имеет смысл.

5. Заключение

В рамках работы над проектом РФФИ № 20-07-00594 выполняется разработка библиотеки генерации многоверсионного кода, с помощью которой можно создавать как векторный код, предназначенный для целевой аппаратной платформы, так и код с эмуляцией векторных инструкций. В режиме эмуляции векторных инструкций реализованы возможности по сбору статистики времени выполнения программы, в частности собирается статистика по плотности векторных масок. Данная информация может быть использована для повышения производительности векторизуемых приложений.

В данной статье показано применение разрабатываемой библиотеки генерации многоверсионного кода на примере отдельной функции в составе римановского решателя и продемонстрирован положительный эффект от мониторинга плотности векторных масок, проявляющийся для расчетных физических задач.

Работа выполнена при поддержке гранта РФФИ № 20-07-00594.

Improving vector code performance by monitoring masks density in vector instructions

A.A. Rybakov, A.D. Chopornyak

Abstract. One of the key factors affecting the effectiveness of vector code created using the AVX-512 is the insufficient density of masks in vector instructions. This indicates that the capabilities of vector instructions are not fully used, and only part of the vector elements are processed during code execution. This article is devoted to the creation of a library of multi-version code, with which you can generate both vector code and regular code with emulation of vector instructions and the ability to monitor their individual characteristics, and in particular the density of the masks. The results of such monitoring can be used to improve the performance of vector code.

Keywords: vectorization, performance increase, vector mask, intrinsic functions, AVX-512.

Литература

1. Vectorization opportunities for improved performance with Intel AVX-512. https://techdecoded.intel.io/resources/vectorization-opportunities-for-improved-performance-with-intel-avx-512, дата обращения 18.06.2020.

2. Intel 64 and IA-32 architectures software developer's manual. Combined volumes: 1, 2A, 2B, 2C, 2D, 3A, 3B, 3C, 3D and 4. Intel Corporation, October 2019.

3. B.M. Shabanov, A.A. Rybakov, S.S. Shumilin. Vectorization of high-performance scientific calculations using AVX-512 instruction set. Lobachevskii Journal of Mathematics, 2019, Vol. 40, No. 5, p. 580–598.

4. А.А. Рыбаков, С.С. Шумилин. Исследование эффективности векторизации гнезд циклов с нерегулярным числом итераций. // Программные системы: Теория и алгоритмы, 2019, Т. 10, № 4 (43), с. 77–96.

5. Analyze vector instruction set with Intel VTune Amplifier. https://software.intel.com/content/www/us/en/develop/documentation/itac-vtune-mpi-openmp-tutorial-lin/top/analyze-vector-instruction-set-with-intel-vtune-amplifier.html, дата обращения 18.06.2020.

6. K. O'Leary. Intel VTune Amplifier and Intel Advisor. Optimization workshop, 2019, https://www.nccs.nasa.gov/sites/default/files/Optimization.pdf, дата обращения 18.06.2020.

7. Intel Intrinsics Guide. https://software.intel.com/sites/landingpage/IntrinsicsGuide, дата обращения 18.06.2020.

8. A.A. Rybakov, S.S. Shumilin. Vectorization of the Riemann solver using the AVX-512 instruction set. Program Systems: Theory and Applications, 2019, Vol. 10, № 3 (42), p. 41–58.

9. C.R. Ferreira, K.T. Mandli, M. Bader. Vectorization of Riemann solvers for the single- and multi-layer shallow water equations. Proceedings of the 2018 International Conference on High Performance

Computing and Simulation, HPCS 2018 (16-20 July 2018, Orleans, France), 2018, p. 415–422.

10. А.А. Рыбаков. Векторизация нахождения пересечения объемной и поверхностной сеток для микропроцессоров с поддержкой AVX-512. Труды НИИСИ РАН, 2019, Т. 9, № 5, с. 5–14.

11. А.А. Рыбаков, С.С. Шумилин. Векторизация сильно разветвленного управления с помощью инструкций AVX-512. Труды НИИСИ РАН, 2018, Т. 8, № 4, с. 114–126.

12. E.F. Toro. Riemann solvers and numerical methods for fluid dynamics: A practical introduction. 2nd edition, Springer, Berlin-Heidelberg, 1999, 645 p.

Подписано в печать 08.09.2020 г.
Формат 60x90/8
Печать цифровая. Печатных листов 6
Тираж 100 экз. Заказ № 683

Отпечатано в ФГУП «Издательство «Наука»
(Типография «Наука»)
121099, Москва, Шубинский пер., 6