УДК 004.2, 004.31 DOI 10.23683/2311-3103-2018-8-48-58

Н.А. Лукин

IP-ЯДРО ДЛЯ ФУНКЦИОНАЛЬНО-ОРИЕНТИРОВАННОГО ПРОЦЕССОРА С АРХИТЕКТУРОЙ VLIW-RISC

Анализ алгоритмов систем управления реального времени показывает, что для обеспечения требуемой производительности встроенных вычислительных средств необходимо обеспечивать минимальное время реализации заранее выбранных базовых алгоритмических процедур. СБИС-реализация специализированных архитектур встроенных процессоров позволяет за счет аппаратной реализации таких процедур существенно минимизировать время их выполнения. Главными целями синтеза IP-ядра функционально-ориентированного процессора в этом случае являются максимально-возможное быстродействие при реализации базовой процедуры и минимум накладных затрат на организацию работы программ. Архитектура IP-ядра, основанная на комбинации RISC с командами класса VLIW, позволяет достичь указанных целей при реализации скалярного произведения векторов, принятого в качестве базовой процедуры. Разработано высокопроизводительное 64-разрядное процессорное ядро, которое выполняет любую операцию системы команд не более чем за один такт, обеспечивая минимальное время вычисления скалярного произведения. Это обуславливает максимально быструю реализацию алгоритмов навигации для систем управления ракетно-космической техникой и летательными аппаратами. В статье рассматриваются результаты разработки технологическое программное обеспечение, предназначенное для разработки и отработки штатных программ функционально-ориентированного процессора. В состав этого программного обеспечения входят специализированный язык ассемблера DAPLANG, созданный на базе макрогенератора ML-1, а также специальная библиотека взаимодействия ФОП с программной средой скриптового языка Lua. Приводятся результаты экспериментальных исследований ФОП с целью верификации архитектуры и системы команд.

Функционально-ориентированные процессоры; IP-ядро; системы реального времени.

N.A. Lookin

IP-CORE FOR A FUNCTIONAL-ORIENTED PROCESSOR WITH VLIW-RISC ARCHITECTURE

The analysis of algorithms of real-time control systems shows that in order to ensure the required performance of embedded computer systems, it is necessary to ensure the minimum time for processing the specific algorithmic procedures. The VLSI implementation of special-purpose architectures of embedded processors allows, due to the hardware implementation of such procedures, to significantly minimize the processing time. The main goals of the synthesis of IP-core of the functional-oriented processor in this case are the maximum possible performance of processing of the basic procedure and a minimum of overhead costs for the programs functioning. The IP-core, based on a combination of RISC architecture and VLIW-class instructions, makes it possible to achieve these goals when computing the scalar product of vectors, adopted as the basic procedure. A high-performance 64-bit processor core has been developed, which performs any operation of the instruction set in no more than one clock cycle, ensuring minimal computing time for the scalar product. This leads to the fastest possible implementation of navigation algorithms for control systems for rocket and space technology and aircraft. The article discusses the results of the development of technological software designed for the design and verification of application programs of a function-oriented processor. The structure of this software includes a specialized assembler language DAPLANG, created on the basis of the ML-1 macro-generator, as well as a special library for interacting FOP with the Lua scripting language. The results of experimental studies of FOP with for verifying the architecture and system of commands are given.

Functional-oriented processor; IP-core; real-time systems.

Введение. Для обеспечения высокой эффективности реализации алгоритмов систем реального времени необходимо добиваться рациональных решений в части архитектур и встроенного (штатного) программного обеспечения. Это требует на этапе разработки архитектур соответствующих функционально-ориентированных

процессоров (ФОП) оптимизации вычислений по критерию сложности на основе учета взаимного отображения алгоритмов и архитектур [1–3]. Если учесть предполагаемую реализацию ФОП в виде сверхбольшой интегральной схемы (СБИС), то можно сформулировать проблему синтеза рациональной (или оптимальной) архитектуры СБИС ФОП в базисе схем из функциональных элементов [4, 5]. В статье рассматриваются некоторые задачи архитектурного проектирования СБИС ФОП и обсуждаются результаты разработки процессорного ядра ФОП, предназначенного для эффективной реализации алгоритмов навигации для систем реального времени.

Особенности проектирования ФОП для систем реального времени. Даже для алгоритмов конкретных классов задач не существует научных и инженерных методик автоматизированного проектирования рациональных (тем более, оптимальных) архитектур ФОП [6]. Поэтому начальным этапом создания архитектуры любого ФОП, встроенного в систему реального времени, является **анализ алгоритмов** конкретных задач [7] с целью выявления вычислительных процедур, в наибольшей степени влияющих на время реализации алгоритмов в целом.

Следующим этапом проектирования ФОП служит **анализ взаимосвязей** выделенных **вычислительных процедур** алгоритмов конкретного класса **и структурно-логических характеристик** функциональных модулей, в результате которого строится формализм, учитывающий как алгоритмы, так и архитектуры, что является основой для их совместной оптимизации [4]. Это соотношения, связывающие между собой аппаратную и временную сложность вычислений в выбранном базисе ФМ. Чаще всего используются два вида оптимума [4]:

- ♦ минимум времени реализации алгоритма при заданной величине аппаратурных затрат;
- ♦ минимум аппаратурных затрат на реализацию алгоритма при заданной величине времени вычислений.

Данный этап проектирования заканчивается созданием **рациональной (или оптимальной) архитектуры ФОП**.

Архитектура любого вычислительного устройства вообще и ФОП, в частности – это результат отображения графа алгоритма на граф структуры процессора [6]. На данном этапе реализуется задача **покрытия спроектированной архитектуры ФОП элементным базисом СБИС**, и в результате создается логический проект ФОП. Специфика разработки ФОП заключена, в основном, в этапах создания его архитектуры, так как появляется возможность оптимизации на основе использования определяющих соотношений между сложностью вычислений и производительностью ФОП. Это создает предпосылки к построению прикладной теории архитектурного проектирования ФОП как класса вычислительных устройств и открывает путь к их автоматизированному (в ряде случаев – автоматическому) проектированию. Это особенно важно в случае разработки ФОП как СБИС.

Еще одна специфика разработки ФОП как компонента системы реального времени проявляется при создании его программного обеспечения. Ограниченный круг реализуемых алгоритмов, а также требование высокой (иногда сверхвысокой) производительности требуют минимизации всех возможных накладных расходов при решении задач в реальном времени [8]. Мы приходим к языку программирования уровня ассемблера, но учет специфики архитектуры требует реализации таких языковых конструкций, которые позволяют адаптировать язык в случае изменения архитектурных параметров. В итоге, базовым языком программирования ФОП следует считать ассемблер модульного типа, основанный на развитой библиотеке макросов. Модульный ассемблер может служить основой соответствующего языка спецификаций, для которого характерно то, что пользователь по большей части программирует алгоритмические функциональные блоки с помощью процедур библиотеки, нежели чем с помощью операторов общего типа. В этом смысле язык программирования тоже становится функционально-ориентированным.

Описанная выше в общем виде технология проектирования рациональных архитектур ФОП была конкретизирована при разработке СБИС ФОП для конкретной предметной области – автономная бесплатформенная навигация для объектов ракетно-космической техники и летательных аппаратов.

Архитектура функционально-ориентированного процессора ФОП-64. Анализ основных алгоритмов вычисления параметров ориентации показывает, что базовыми вычислительными операциями являются операции векторно-матричной алгебры [9]. Наиболее часто используется операция, которая в обобщенном виде представляет суммирование декартовых произведений конечных множеств [10]:

$$f = \sum_{m=1}^{M}\left[\left\{a_{(i_1,i_2,\ldots,i_N)}\right\} \times \left\{b_{(j_1,j_2,\ldots,j_N)}\right\} \times \ldots \times \left\{z_{(k_1,k_2,\ldots,k_N)}\right\}\right]_m, \quad (1)$$

где N – мощность конечных множеств, M – число множеств. Эта операция обобщает скалярное произведение векторов, которое в случае 3-х мерного пространства приобретает вид:

$$f = \sum_{m=1}^{3}(a_m b_m). \quad (2)$$

Операция умножения практически полностью определяет точность реализации алгоритмов, что обуславливает выполнения умножения с обязательным получением *(rA + rB)*-разрядного произведения, где *rA, rB* – разрядности сомножителей *A* и *B* соответственно.

На основании анализа возможных вариантов реализации процедуры (2) в базисе схем из функциональных элементов [9] были получены верхние оценки аппаратной и временной сложности вычислений и произведена оптимизация полученных схем. В результате, была предложена функциональная схема вычислительного (процессорного) ядра ФОП в базисе функциональных элементов СБИС, при этом основой служили вентили типа 2И-НЕ.

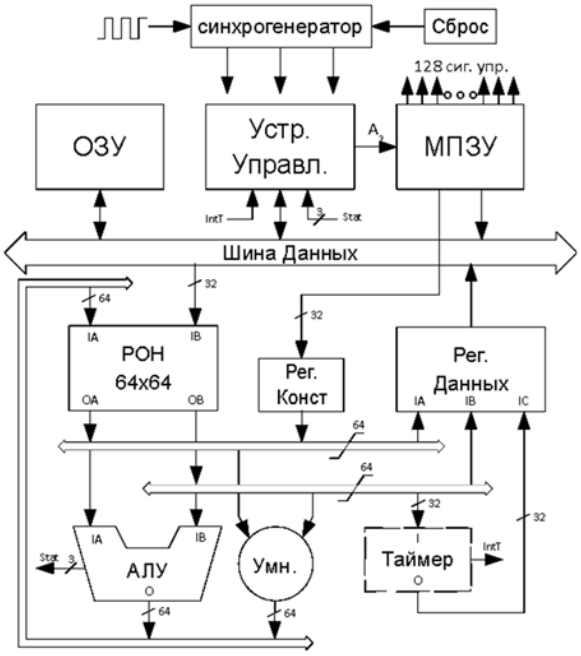

Рис. 1. Структура ФОП-64

Обозначения на рис. 1:

ОЗУ – оперативное запоминающее устройство; Устр. Управл. – устройство управления; МПЗУ – микропрограммное запоминающее устройство; РОН – регистры общего назначения; Рег. Конст. – регистр констант; Рег. Данных – регистр данных; АЛУ – арифметико-логическое устройство; Умн. – умножитель.

На базе полученных оценок сложности и синтезированной функциональной схемы ядра была разработана его структурная схема. Структурная схема ФОП-64 приведена на рис. 1.

Основой организации процессорного ядра является одновременное управление работой всех его модулей на основе сверхдлинного командного слова (VLIW), состоящего из независимых функциональных полей. Длина командного слова составляет 128 бит. Это позволяет программисту управлять работой всех функциональных модулей архитектуры, добиваясь их одновременного функционирования во время исполнения команды. Все операции ядра выполняются за 1 такт, обеспечивая максимально-возможную производительность обработки данных. Архитектура ФОП-64 принадлежит к классу RISC+VLIW [11].

На ранних этапах работ было произведено сравнение производительности аналогичной архитектуры 32-разрядного ядра с существующими в то время процессорными ядра бортовых ЦВМ. Было показано, что для реализации широкого класса алгоритмов навигации разработанная архитектура ФОП обеспечивает меньшее число тактов по сравнению с гарвардским, а также векторным типами архитектур [12, 13].

Аппаратная реализация ФОП-64. Создание штатного варианта ФОП-64 предполагается состоящим из двух этапов:

♦ I этап. Разработка прототипа СБИС ФОП, позволяющего, с одной стороны, проводить обработку данных в реальном времени, а с другой – по максимуму учесть предполагаемую микроэлектронную реализацию.

♦ II этап. Разработка СБИС ФОП. В ходе выполнения настоящей работы был разработан прототип процессорного ядра СБИС ФОП на основе серийно изготавливаемой отладочной платы Terasic DE2-115. Наличие на плате развитой периферии, которая позволяет стыковать прототип ФОП как с инерциальными датчиками, так и центральной навигационной системой или также с ее прототипом, т.е. с ПЭВМ. На рис. 2 представлена блок-схема отладочной платы Terasic DE2-115 [14] на базе FPGA Altera Cyclone IV.

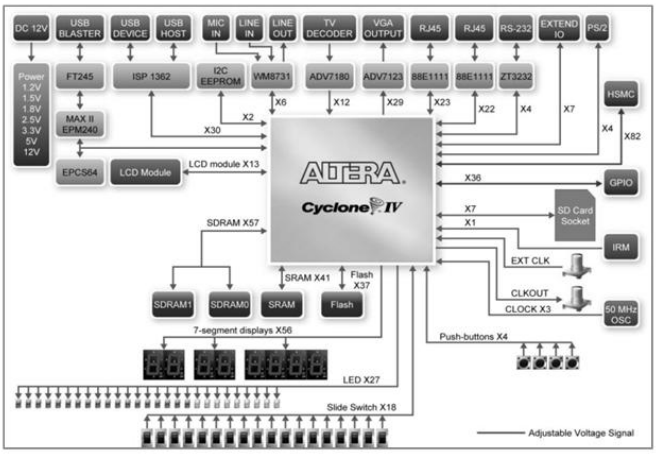

Рис. 2. Блок-схема отладочной платы Terasic DE2-115

Программное обеспечение ФОП-64. Полный комплекс программного обеспечения процессора состоит из трех основных составляющих:

- Технологическое ПО (ТПО). Оно встроено в состав компьютерной рабочей станции, предназначенной для разработки и отработки алгоритмов и программ на ФОП.
- Системное ПО. Это ПО встроено в ФОП и представляет собой множество программных подсистем. Основными являются подсистема компиляции с входного языка разработчика прикладных программ в совокупность исполняемых кодов и подсистема управления вычислительным процессом.
- Штатное ПО. Это ПО встроено в состав ФОП и размещается в его памяти программ. Оно непосредственно реализует алгоритмы целевых систем.

В настоящее время разработаны следующие основные программы ТПО прототипа ФОП:

- программный эмулятор архитектуры ФОП;
- язык программирования нижнего уровня DAPLANG 1.0.

Эмулятор ФОП представляет собой динамически подключаемую библиотеку в формате Microsoft Windows DLL [15]. В библиотеке реализована потактовая модель процессора и функции его запуска, останова, записи/чтения памяти и МПЗУ, чтения внутренних регистров. Отладочное окружение представляет собой исполняемую программу, которая реализует функции интерфейса библиотеки собственно эмулятора. Отладочная программа позволяет производить загрузку программы в МПЗУ процессора, чтение/запись файлов в память процессора, а также вводить данные в память вручную.

Язык программирования DAPLANG 1.0. Для разработки программ на ядре ФОП-64 был разработан язык программирования DAPLANG 1.0. Этот язык принадлежит к классу ассемблеров и разработан на базе макрогенератора ML/1 [9]. Возможности эмулятора и языка DAPLANG 1.0 позволяют разрабатывать любые программы на модели и физическом прототипе ФОП.

Система отработки программ ФОП-64. Для обеспечения эффективной отработки штатных программ ФОП разработана специализированная программная система, являющаяся частью ТПО. Эта система представляет собой специальную библиотеку взаимодействия эмулятора ФОП и программной среды скриптового языка Lua [18], интерпретатор которого является свободно распространяемым, с открытыми исходными текстами на языке Си. Он широко используется для создания программного обеспечения из-за удобства встраивания, скорости исполнения кода и лёгкости обучения. На базе интерпретатора языка Lua, разработанной библиотеки взаимодействия с эмулятором и свободно распространяемой оболочки управления проектами Lua Development Tools на базе Eclipse [19] создана программная среда для запуска и отладки программ пользователя.

Структура ТПО приведена на рис. 3.

Разработанное ТПО было использовано в ходе экспериментальной отработки структуры и штатных программ ФОП-64.

Экспериментальная отработка архитектуры и программного обеспечения ФОП-64. ФОП-64 – это специализированный процессор в составе системы реального времени. Поэтому верификацию архитектуры и программного обеспечения необходимо проводить, прежде всего, исходя из необходимости реализации алгоритмов предметной области, для которой предназначена система.

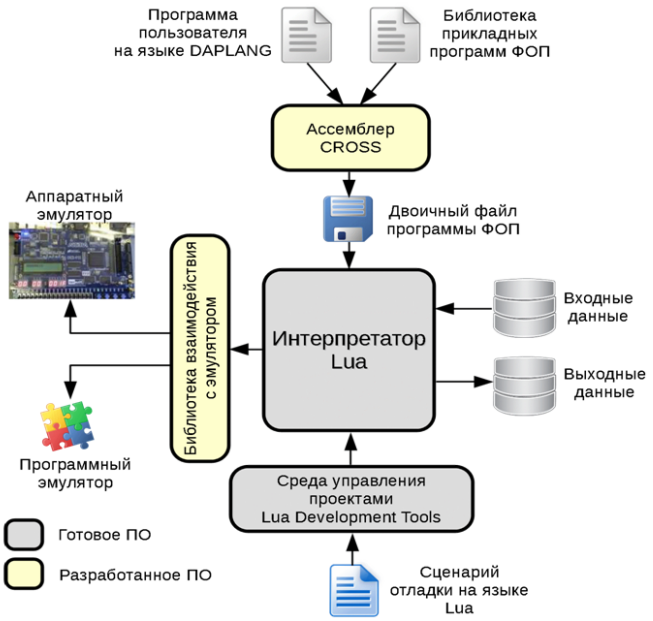

Рис. 3. Структура ТПО ФОП-64

В нашем случае такой областью являлась бесплатформенная навигация, а наиболее вычислительно сложными служили алгоритмы вычисления параметров вращения и линейного перемещения твердого тела в трехмерном пространстве. Базовой технологией верификации была совокупность прогонов штатных программ, реализующих алгоритмы навигации, при этом множество возможных вариантов прогонов состояло из двух основных групп:

- Прогоны, которые соответствуют режимам работы ФОП в составе СУ изделий с экстремальными значениями механических воздействий (угловые скорости и линейные ускорения).
- Прогоны программ с экстремальными значениями переменных или их сочетаний. Основная цель для данной группы прогонов – верификация корректности работы ПО и архитектуры ФОП.

Основными задачами, решаемыми с помощью прогонов реальных алгоритмов, являлись измерение и оценка суммарной погрешности и среднего времени реализации алгоритмов. Собственно верификация проводилась на базе простого стенда, состоящего из стандартного компьютера (PC), прототипа ядра СБИС ФОП-64 на основе упомянутой FPGA Altera и специально разработанного устройства измерения реального времени (УИВ), стабильность работы которого на десятичный порядок выше по отношению к внутреннему генератору ядра ФОП-64. Основные режимы стенда приведены в табл. 1.

Каждый прогон представлял собой программную реализацию следующих алгоритмов:

- вычисления вектора ориентации;
- вычисление кватерниона поворота от инерциальной системы координат (ИСК) к связанной (ССК);
- матрицы перехода от ССК к ИСК;
- вычисление приращения кажущейся скорости в ИСК;
- определение координат и скорости объекта в ИСК.

Таблица 1

Режимы функционирования стенда ФОП-64

№ п\п	Наименование режима	Назначение режима	Описание работы стенда
1	Начальная установка	Приведение аппаратуры и ПО стенда в фиксированное состояние	1^0 РС: выдача команды на приведение ФОП в фиксированное состояние. 2^0 ФОП: переход к программе "Начальная установка".
2	Обнуление ЗУ	Приведение ОЗУ и РОН ФОП в состояние "0"	1^0 РС: выдача команды "Обнуление ЗУ" 2^0 ФОП: переход к программе "Обнуление ЗУ".
3	Обработка данных	Прогон i-ой программы ФОП	1^0 РС: выдача команды "Начать i-ую программу" 2^0 ФОП: Инициализация i-ой программы обработки данных. 3^0 Выдача в УИВ команды начала формирования временного интервала. 4^0 Обработка данных по i-ой программе 5^0 Выдача в УИВ команды окончания формирования временного интервала. 6^0 Выдача в РС признака окончания обработки данных.
4	Запись данных в ФОП	Запись данных в ОЗУ ФОП по команде РС	1^0 РС: выдача команды "ЗП" 2^0 РС $\leftrightarrow$ ФОП: запись массива данных из РС в ФОП
6	Чтение данных из ФОП в РС	Чтение данных из ФОП в РС по команде РС	1^0 РС: выдача команды "ЧТ" 2^0 РС $\leftrightarrow$ ФОП: чтение массива данных из ФОП в РС

Перечень алгоритмов охватывает собой все необходимые этапы решения задачи бесплатформенного блока ориентации, а также значительную часть задачи инерциальной навигации, в этом смысле прогоны являлись представительными.

Кроме прогонов штатных программ, реализующих упомянутые выше алгоритмы навигации, было произведено сравнение ФОП-64 и серийной БЦВМ "Малахит-7" разработки НПО автоматики, которая является базовой для СУ ракеты-носителя "Союз-2 ТМ" [20]. Данная разработка относится к периоду начала 2010-ых годов, процессор БЦВМ реализует 32-разрядные вычисления и имеет гарвардскую архитектуру. В ходе проведения прогонов программ были получены оценки необходимого числа тактов на реализацию алгоритмов для ФОП-64 и "Малахит-7", при этом за один такт работы процессоров в обоих случаях был принят интервал времени выполнения одной операции сложения. В табл. 2 приведены результаты сравнительных экспериментов с ФОП-64 и БЦВМ "Малахит-7".

Таблица 2

Реализация математических функций и алгоритмов навигации в БЦВМ «Малахит-7» и ФОП-64

Функции			**Параметры реализации**	
Наименование функции	Процессор	Диапазон изменения аргумента	Длина подпрограмм, команд	Время, тактов
$Y = Sin(X)$	ФОП-64	$X \in [-\pi/2;\ \pi/2]$	56	56
	Малахит-7	$X \in [-1;\ 1]$	110	84
$Y = 1/\sqrt{X}$	ФОП-64	$X \in [0.5; 1]$	28	50
	Малахит-7	$X \in [0.5; 1]$	125	562-586
Алгоритмы бесплатформенной системы ориентации			**Параметры реализации**	
Наименование алгоритма	Процессор		Время реализации, тактов	Суммарная абсолютная погрешность
5-ти шаговый алгоритм БИСО (разработки НПОА)	Малахит-7		6462	$4 \cdot 10^{-6}$
	ФОП-64		854-878	$1{,}2 \cdot 10^{-11}$*

Эксперименты показали существенное превосходство архитектуры ФОП-64 над "Малахит-7" как по времени вычисления, так и по суммарной абсолютной погрешности реализации алгоритмов. Основными причинами этого явились аппаратная реализация 64-разрядных вычислений, однотактное выполнение практически всех арифметических операций в ФОП-64, аппаратная поддержка режимов подкачки операндов из ОЗУ в РОНы, что обуславливает выполнение подпрограмм вычисления функций над переменными в сверхоперативной памяти без обращения к ОЗУ.

В ходе проведения экспериментов было произведено программирование алгоритмов БИСО для высокоманевренных малогабаритных летательных аппаратов. Эти алгоритмы более сложны, так как определение параметров ориентации необходимо производить в условиях экстремальных механических воздействий (линейное ускорение до 200g и угловые скорости до 600 ^{0}/сек). В каждом цикле выполнялись все алгоритмы определения угловых и линейных параметров движения летательного аппарата, а также определения его скорости и координат в ИСК. Это потребовало не более 3000 тактов работы ФОП. Расчеты времени реализации алгоритмов БИСО в предположении разработки описываемого ядра в составе СБИС с учетом использования технологии КМОП с нормами 90 нм показали, что длительность цикла съема информации с инерциальных датчиков и, соответственно, время реализации алгоритмов с помощью ФОП не превысит 120–150 мксек. Это существенно меньше достигнутого в современных отечественных и зарубежных навигационных системах, поэтому решения, заложенные в ФОП, обладают перспективой с точки зрения реализации алгоритмов БИНС для широкого класса высокоманевренных ЛА.

Заключение. Коллективом исследователей и разработчиков Группы микропроцессорных архитектур ИМАШ УрО РАН совместно с сотрудниками Института радиоэлектроники и информационных технологий Уральского федерального университета (ИРИТ-РТФ УрФУ) была проведена разработка ФОП третьего поколения, предназначенного для эффективной реализации алгоритмов бесплатформен-

ных навигационных систем. Требования жесткого реального времени определили облик архитектуры ФОП БИНС – разрядность вычислений должна составлять не менее 64 бит, а максимально-допустимый период считывания инерциальных параметров с инерциального измерительного блока не должен превышать 500 мксек. Это были главные параметры, которыми задавались разработчики при проектировании архитектуры ФОП.

Главным результатом исследований явилась архитектура ФОП класса RISC с управлением обработки данных с помощью командного слова большой разрядности – VLIW (128 бит). Вычислительное ядро данной архитектуры представляет собой 64-битные АЛУ и аппаратный умножитель, реализующий однотактное умножение со знаками с выдачей 128-битного произведения, а также 4-х портовый регистровый файл емкостью 64 регистра по 64 бит каждый.

Результаты проведенных работ позволяют сделать вывод том, что ФОП с разработанной архитектурой может являться бортовым вычислителем для реализации алгоритмов бесплатформенной навигации в перспективных СУ летательных аппаратов и ракетно-космических комплексов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Vuduc R., Czechowski K.* Toward a Theory of Algorithm-Architecture Co-design // High Performance Computing for Computational Science - VECPAR 2012. VECPAR 2012. Lecture Notes in Computer Science. Vol 7851. – Springer, Berlin, Heidelberg, 2012. – P. 4-8.
2. *Czechowski K., & Vuduc R.* A Theoretical Framework for Algorithm-Architecture Co-design // IEEE 27th International Symposium on Parallel and Distributed Processing. – 2013. – P. 791-802.
3. *Zhu Y., Samajdar A., Mattina M., & Whatmough P.* Euphrates: Algorithm-SoC Co-Design for Low-Power Mobile Continuous Vision // ACM/IEEE 45th Annual International Symposium on. Computer Architecture (ISCA). – 2018. – P. 547-560.
4. *Лукин Н.А.* Системное проектирование функционально-ориентированных процессоров для бортовых корреляционно-экстремальных навигационных систем // Вестник Самарского государственного аэрокосмического университета им. академика С.П. Королёва. – 2009. – № 4 (20). – С. 218-236.
5. *Edwards M.D.* A generic hardware architecture to support the system level synthesis of digital systems // Microprocessing and Microprogramming. – 1994. – No. 40 (4). – P. 225-240.
6. *Henessy J.L.* VLSI Processor Architecture // IEEE Transactions on Computers. – December 1984. – Vol. c-33, No. 12. – P. 1221-1246.
7. *Скиена* С. Алгоритмы. Руководство по разработке. – 2-е изд.: пер. с англ. – СПб.: БХВ-Петербург. 2011. – 720 с.
8. *Kolar D., Cerny S.* Evolution of software for embedded systems in processor expert // Proceedings. 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems. – 2004. – P. 419-422.
9. *Лукин Н.А.* Функционально-ориентированные процессоры для реализации алгоритмов БИНС // Гироскопия и навигация. – 2001. – № 2. – С. 40-56.
10. *Lovelly T.M., Bryan D., Cheng K., Kreynin R., George A., Gordon-Ross A., Mounce G.* A framework to analyze processor architectures for next-generation on-board space computing // Proceedings of 2014 IEEE Aerospace Conference, 1-8 March 2014. – P. 1-8.
11. *Кузнецов О.П.* Дискретная математика для инженера. – 3-е изд. – СПб.: Изд-во "Лань", 2004. – 400 с.
12. *Lin T.J., Chao C.M., Liu C.H., Hsiao P.C., Chen S.K., Lin L.C., Liu C.W., Jen C.W.* A unified processor architecture for RISC & VLIW DSP // Proceedings of the 15th ACM Great Lakes Symposium on VLSI 2005, Chicago, Illinois, USA, April 17-19, 2005. – P. 50-55.
13. *Лукин Н.А., Водичева Л.В., Пономарев И.Г.* Принципы оптимального проектирования БИНС: функционально-стоимостной анализ реализации алгоритмов // Гироскопия и навигация. – 2005. – № 4 (51). – С. 14-22.
14. *Lookin N.A., Vodicheva L.V., Ponomarev I.G.* A Miniature Precise SINS for High Maneuvering Moving Vehicles: Cost-Efficiency Analysis of Algorithm Realization // 11thSaint-Petersburg International Conference on Integrated Nav-igation Systems, St.-Petersburg, May 23-25, 2005. – P. 243-245.

15. https://www.intel.com/content/www/us/en/programmable/solutions/partners/partner-profile/terasic-inc-/board/altera-de2-115-development-and-education-board.html.
16. https://docs.microsoft.com/en-us/windows/desktop/dlls/dynamic-link-libraries.
17. *Tanenbaum A.S.* A General-Purpose Macro Processor as a Poor Man's Compiler-Compiler // IEEE Transactions on Software Engineering. SE-2 (2). – 1976. — P. 121-125.
18. *Лукин Н.А.* Эволюция специализированного параллелизма и систолические вычисления в системах реального времени // Известия ВУЗов. Физика. – 2016. – № 8/2. – P. 65-67.
19. *Ierusalimschy R.* Programming in Lua. Lua.Org (August 1, 2016).
20. https://wiki.eclipse.org/LDT/User_Area/User_Guides/User_Guide_1.3.
21. *Бельский Л.Н., Дерюгин С.Ф., Смирнов В.П., Трапезников М.Б., Шалимов Л.Н.* Разработка бортовых цифровых вычислительных средств ФГУП «НПО автоматики им. академика Н.А. Семихатова» // История отечественной электронной вычислительной техники. – М.: ООО Изд. дом "Столичная энциклопедия", 2017. – С. 375-385.

REFERENCES

1. *Vuduc R., Czechowski K.* Toward a Theory of Algorithm-Architecture Co-design, *High Performance Computing for Computational Science - VECPAR 2012. VECPAR 2012. Lecture Notes in Computer Science*. Vol 7851. Springer, Berlin, Heidelberg, 2012, pp. 4-8.
2. *Czechowski K., & Vuduc R.* A Theoretical Framework for Algorithm-Architecture Co-design, *IEEE 27th International Symposium on Parallel and Distributed Processing*, 2013, pp. 791-802.
3. *Zhu Y., Samajdar A., Mattina M., & Whatmough P.* Euphrates: Algorithm-SoC Co-Design for Low-Power Mobile Continuous Vision, *ACM/IEEE 45th Annual International Symposium on. Computer Architecture (ISCA)*, 2018, pp. 547-560.
4. *Lukin N.A.* Sistemnoe proektirovanie funktsional'no-orientirovannykh protsessorov dlya bortovykh korrelyatsionno-ekstremal'nykh navigatsionnykh sistem [System design of function-oriented processors for on-board correlation-extreme navigation systems], *Vestnik Samarskogo gosudarstvennogo aerokosmicheskogo universiteta im. akademika S.P. Koroleva* [Bulletin of Samara State Aerospace University named after academician S.P. Korolev], 2009, No. 4 (20), pp. 218-236.
5. *Edwards M.D.* A generic hardware architecture to support the system level synthesis of digital systems, *Microprocessing and Microprogramming*, 1994, No. 40 (4), pp. 225-240.
6. *Henessy J.L.* VLSI Processor Architecture, *IEEE Transactions on Computers*, December 1984, Vol. c-33, No. 12, pp. 1221-1246.
7. *Skiena S.* Algoritmy. Rukovodstvo po razrabotke [Algorithms. Development guide]. 2nd ed.: transl. from Engl. Saint Petersburg: BKHV-Peterburg, 2011, 720 p.
8. *Kolar D., Cerny S.* Evolution of software for embedded systems in processor expert, *Proceedings. 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems*, 2004, pp. 419-422.
9. *Lukin N.A.* Funktsional'no-orientirovannye protsessory dlya realizatsii algoritmov BINS [Functionally-oriented processors for the implementation of SINS algorithms], *Giroskopiya i navigatsiya* [Gyroscopy and Navigation], 2001, No. 2, pp. 40-56.
10. *Lovelly T.M., Bryan D., Cheng K., Kreynin R., George A., Gordon-Ross A., Mounce G.* A framework to analyze processor architectures for next-generation on-board space computing, *Proceedings of 2014 IEEE Aerospace Conference, 1-8 March 2014*, pp. 1-8.
11. *Kuznetsov O.P.* Diskretnaya matematika dlya inzhenera [Discrete Mathematics for the Engineer]. 3rd ed. Saint Petersburg: Izd-vo "Lan'", 2004, 400 p.
12. *Lin T.J., Chao C.M., Liu C.H., Hsiao P.C., Chen S.K., Lin L.C., Liu C.W., Jen C.W.* A unified processor architecture for RISC & VLIW DSP, *Proceedings of the 15th ACM Great Lakes Symposium on VLSI 2005, Chicago, Illinois, USA, April 17-19, 2005*, pp. 50-55.
13. *Lukin N.A., Vodicheva L.V., Ponomarev I.G.* Printsipy optimal'nogo proektirovaniya BINS: funktsional'no-stoimostnoy analiz realizatsii algoritmov [Principles of optimal SINS design: functional cost analysis of the implementation of algorithms], *Giroskopiya i na-vigatsiya* [Gyroscopy and Navigation], 2005, No. 4 (51), pp. 14-22.
14. *Lookin N.A., Vodicheva L.V., Ponomarev I.G.* A Miniature Precise SINS for High Maneuvering Moving Vehicles: Cost-Efficiency Analysis of Algorithm Realization, *11thSaint-Petersburg International Conference on Integrated Nav-igation Systems, St.-Petersburg, May 23-25, 2005*, pp. 243-245.

15. Available at: https://www.intel.com/content/www/us/en/programmable/solutions/partners/partner-profile/terasic-inc-/board/altera-de2-115-development-and-education-board.html.
16. Available at: https://docs.microsoft.com/en-us/windows/desktop/dlls/dynamic-link-libraries.
17. *Tanenbaum A.S.* A General-Purpose Macro Processor as a Poor Man's Compiler-Compiler, *IEEE Transactions on Software Engineering.* SE-2 (2), 1976, pp. 121-125.
18. *Lukin N.A.* Evolyutsiya spetsializirovannogo parallelizma i sistolicheskie vychisleniya v sistemakh real'nogo vremeni [The evolution of specialized parallelism and systolic calculations in real-time systems], *Izvestiya VUZov. Fizika* [University news. Physics], 2016, No. 8/2, pp. 65-67.
19. *Ierusalimschy R.* Programming in Lua. Lua.Org (August 1, 2016).
20. Available at: https://wiki.eclipse.org/LDT/User_Area/User_Guides/User_Guide_1.3.
21. *Bel'skiy L.N., Deryugin S.F., Smirnov V.P., Trapeznikov M.B., Shalimov L.N.* Razrabotka bortovykh tsifrovykh vychislitel'nykh sredstv FGUP «NPO avtomatiki im. akademika N.A. Semikhatova» [Development of onboard digital computing facilities of FSUE "NPO Avtomatiki im. academician N.A. Semikhatova"], *Istoriya otechestvennoy elektronnoy vychislitel'noy tekhniki* [History of domestic electronic computing]. Moscow: OOO Izdatel'skiy dom "Stolichnaya entsiklopediya", 2017, pp. 375-385.

Статью рекомендовал к опубликованию д.т.н., профессор Л.Г. Доросинский.

Лукин Николай Алексеевич – Институт машиноведения Уральского отделения РАН; e-mail: nicklookin@mail.ru; 620075, г. Екатеринбург, ул. Мамина-Сибиряка, 137, кв. 126; тел.: 89655001842; с.н.с.; к.т.н.

Lookin Nickolay Alexeevich – Institute of Engineering Science of Urals Branch of RAS; e-mail: nicklookin@mail.ru; 137, Mamin-Sibiryak street, ap. 126, Yekaterinburg, 620075, Russia; phone: 89655001842; Senior Researcher; cand. of eng. sc.