УДК 004.2+744.4

МОДЕЛИРОВАНИЕ МЕТОДОВ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ ПРОЦЕССОРА С АРХИТЕКТУРОЙ RISC-V

Сторожев И.Е., Певцов Е.Ф.

Центр проектирования интегральных схем, устройств наноэлектроники и микросистем,
МИРЭА – Российский технологический университет, Москва, Российская Федерация,
storozhev9003@gmail.com

Аннотация: В статье представлена микроархитектура процессора с архитектурой RISC-V, поддерживающая ограниченное количество инструкций из набора команд RV32IM. Проведены разработка и функциональная верификация прототипа на языке описания аппаратуры SystemVerilog, а также статический временной анализ для оценки тактовой частоты в САПР Xilinx Vivado. Применены методы повышения производительности процессора: конвейеризация тракта данных, пересылка результатов (bypass), статический предсказатель условных переходов, синхронный умножитель, сумматор с ускоренным переносом.

Ключевые слова: процессор, RISC-V, микроархитектура, конвейеризация, синхронный умножитель, сумматор с ускоренным переносом.

I. ВВЕДЕНИЕ

Процессоры играют ключевую роль в работоспособности компьютеров. Так как с каждым годом требования по эффективности процессоров растут, разработчики применяют различные методы и техники по повышению производительности своих решений. Архитектура RISC-V – современный открытый стандарт на архитектуру процессора [1,2,3], который широко распространяется не только во всем мире, но и, в частности, в России. Многие крупнейшие компании по производству отечественной цифровой электроники используют данную архитектуру для своих разработок.

II. РАЗРАБОТКА RTL КОДА ПРОЦЕССОРА С ПРИМЕНЕНИЕМ МЕТОДОВ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ

2.1 Конвейерная микроархитектура

Для апробации методов по повышению производительности сначала реализован однотактный процессор, содержащий в себе следующие базовые блоки для выполнения инструкций RV32IM: 32 архитектурных регистра, арифметико-логическое устройство, устройство умножения и деления разделенные память инструкций и данных, счетчик команд. Далее в этот проект была встроена конвейерная микроархитектура, которая является мощным средством повышения производительности, так как добавляет в микроархитектуру временной параллелизм [4]. Реализованная в данном проекте методика конвейеризации подразумевает добавления в тракт данных не архитектурных временных регистров (рис. 1).

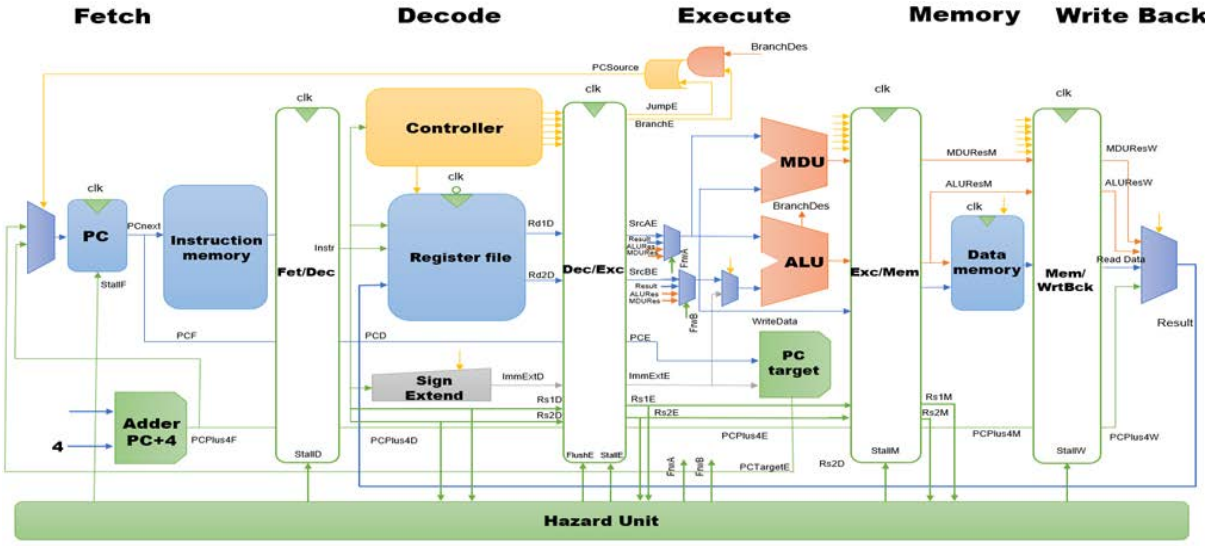

Рисунок 1. Структурная схема конвейерной микроархитектуры процессора

В результате примененной конвейеризации тракт данных разделен на 5 стадий: Выборка (Fetch), Декодирование (Decode), Выполнение (Execute), Обращение к памяти (Memory), Запись в регистры (Write Back) [5]. Известно, что в конвейерной микроархитектуре возникают три вида конфликтов: по управлению, по данным и операционные. Для предотвращения конфликтов по управлению и данным в микроархитектуру встроен блок предотвращения ошибок (Hazard Unit) [6], который содержит следующий функционал: статический предсказатель условных переходов [7], пересылка результатов, остановка и очистка конвейера.

2.3 Повышение производительности операционных устройств

Самой распространенной операцией, выполняемой процессором, является операция сложения. Сумма двух одно битных операндов происходит через простую схему сумматора с полным переносом, но если числа имеют больше разрядов, например 32, как в архитектуре RV32IM, то схема будет содержать 32 полных сумматора, в которых входные и выходные переносы соединены между разрядами, что сильно замедляет вычисление результирующего переноса, так как нужно ждать пока рассчитаются переносы для остальных разрядов. Для повышения производительности в данном проекте в процессор встроен сумматор с ускоренным переносом (carry-lookhead, CLA) [8], так что 32-битный сумматор разделен на восемь 4-разрядных сумматоров, с сигналами генерации (G) и распространения (P), которые требуются для определения переноса n-того разряда сумматора по следующей формуле:

$$C_n = A_n B_n + (A_n + B_n) C_{n-1} = G_n + P_n C_{n-1}, \quad (1)$$

где A_n и B_n являются n-тыми разрядами операндов.

На рисунке 2 представлена RTL схема 32-битного сумматора с ускоренным переносом. В красной области показана синтезированная логика сигналов генерации и распространения, а в фиолетовой области 4 полных сумматора. В каждом из 8 блоков выходной перенос рассчитывается по формуле (1) и переходит в следующий блок.

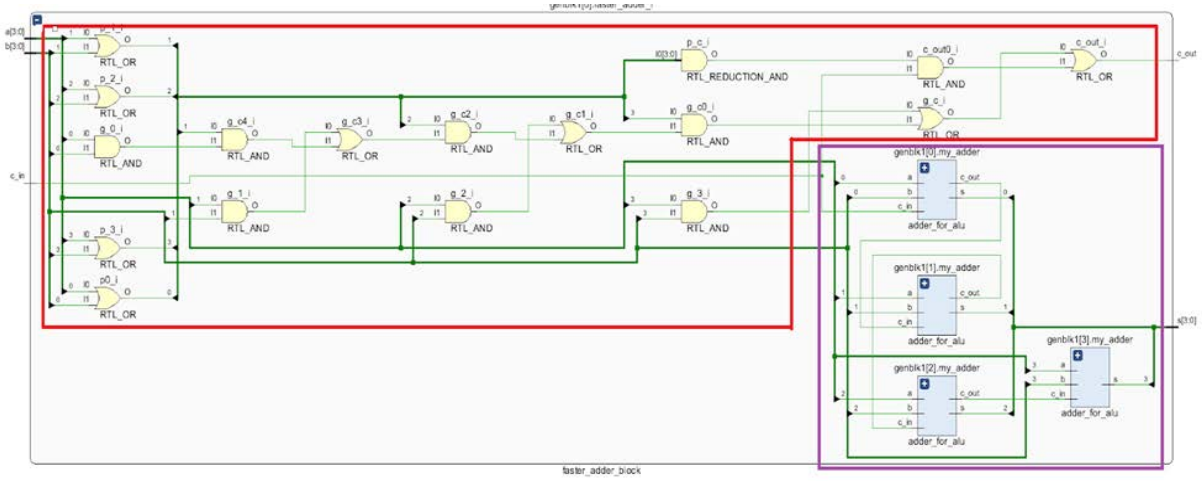

Рисунок 2. RTL блок сумматора с ускоренным переносом

Главный фактор снижения производительности – операция умножения в среди операционных устройств, так как она является самой медленной и относительно часто встречающейся в отличии от деления. В однотактной реализации умножения производилось с помощью комбинаторной логики, что сильно увеличивало критический путь, такие решения не применяются в современных процессорных системах.

В данном проекте реализован синхронный умножитель, который содержит в себе вспомогательные регистры, применение которых позволило выполнять операцию умножения за 32 такта, что соответственно существенно увеличивает тактовую частоту процессора [9].

Алгоритм умножения представляет из себя конечный автомат (КА) с двумя состояниями S0 и S1. В состоянии S0 КА ожидает сигнал run, который инициирует умножение, а именно присвоит значения операндов в регистры, обнулит счетчик, регистр готовности и результирующий регистр. Далее КА переходит в состояние S1, в котором выполняется основной алгоритм. Сначала происходит проверка младшего разряда второго регистра операнда, если он равен единице, то результирующий регистр присвоит значение суммы первого регистра операнда и своего значения, иначе он не изменится, после этого значения регистров операндов сдвигаются на один разряд влево и вправо соответственно, также одновременно с этим счетчик увеличивает свое значение. Далее, если счетчик достиг 32 операция заканчивается и регистр rdone сигнализирует о том, что данные готовы.

RTL-представление блока синхронного умножителя приведено на рис.3.

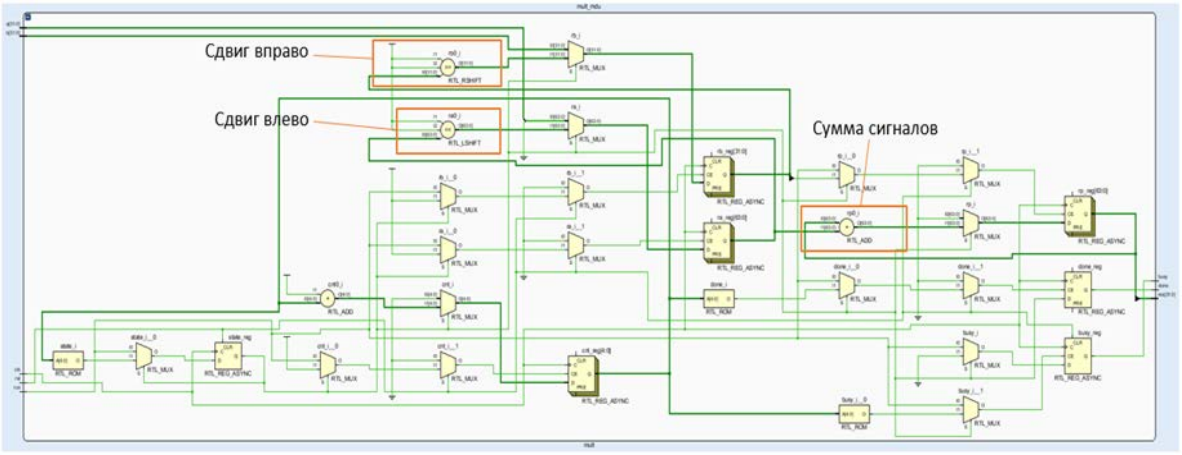

Рисунок 3. RTL блок синхронного умножителя

2.4 Функциональная верификация

Для спроектированного прототипа проведена функциональная верификация, показывающая корректность работоспособность микроархитектуры и примененных методов. В частности для демонстрации работы в память процессора была загружена программа, выполняющая вычисление факториала 7 (рис. 4).

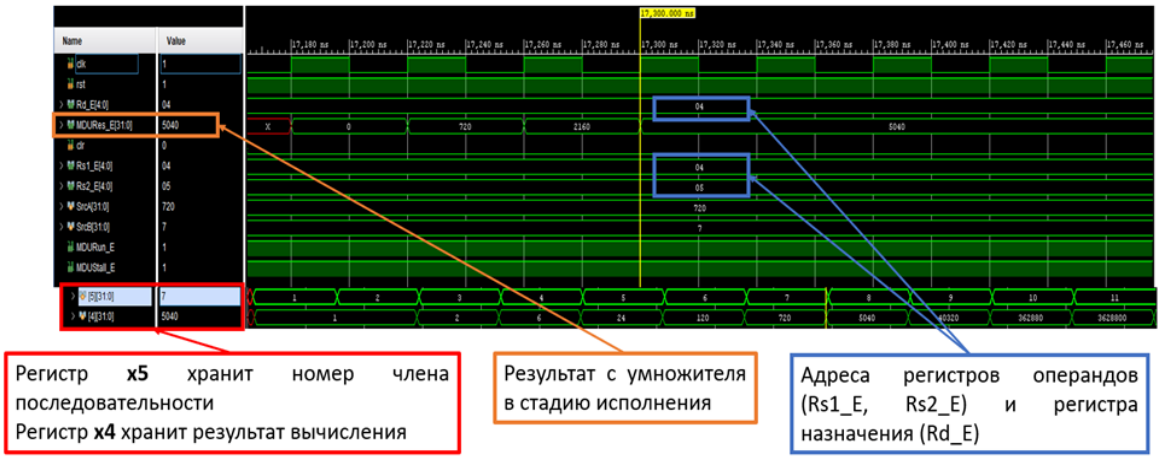

Рисунок 4. Тестовый сценарий вычисления факториала числа 7

Из временной диаграммы видно, что результаты записываются в нужные регистры x5 и x4 каждый такт, что свидетельствует о правильности работы пересылки результата. В процессе выполнения не появляются задержки из-за работы предсказателя. По результатам факториал 7 равен 5040, что является верным значением.

2.5 Статический временной анализ

Для анализа и оценки производительности проводился статический временной анализ (STA) в САПР Xilinx Vivado относительно тактовой частоты 100 МГц для ПЛИС Arty A7 (табл. 1).

По результатам анализа для реализаций с применение методов повышения производительности критический путь составил -0.332 нс. Для однотактной микроархитектуры критический путь составил -8.74 нс.

Таблица 1. Результаты статического временного анализа

Параметры процессора	Критический путь, нс	Тактовая частота, МГц
Без конвейеризации	-8.74	53
После конвейеризации	-0.332	96

III. ЗАКЛЮЧЕНИЕ

Разработан прототип процессора с архитектурой RISC-V на языке описания аппаратуры SystemVerilog, поддерживающий ограниченное количество инструкций из наборов команд RV32IM. Апробированы следующие методы повышения производительности: конвейеризация, включающая в себе блок предотвращения конфликтов, сумматор с ускоренным переносом, синхронные умножитель и делитель. Данные методы и техники делают микроархитектуру высокопроизводительной, что было подтверждено с помощью статического временного анализа, проведенного в САПР Xilinx Vivado. В сравнении с однотактной реализацией, улучшенный процессор имеет тактовую частоту выше на 81%.

БЛАГОДАРНОСТЬ

Работа выполнена при поддержке Министерства науки и высшего образования РФ (Государственное задание для университетов№ ФГФЗ-2023-0005).) и с применением оборудования Центра коллективного пользования РТУ МИРЭА (соглашение от 01.09.2021 № 075–15-2021-689, уникальный идентификационный номер 2296.61321X0010).

ЛИТЕРАТУРА

[1] Aneesh Ravvendran, Vinayak Baramu Patil, David Selvakumar A RISC-V Instruction Set Processor-Microarchitecture Design and Analysis // International Conference on VLSI Systems, Architectures, Technology and Applications. 2016.

[2] David Patterson, Andrew Waterman The RISC-V Reader: An Open Architecture Atlas. University of California, Berkeley: 2017.

[3] Gonzalez A., Korpan B., Zhao J. Replicating and Mitigating Spectre Attacks on a Open Source RISC-V Microarchitecture // CARRV. 2019. №19.

[4] David A. Patterson, John L. Hennessy Computer Organization and Design. Cambridge: Elsevier, 2018.

[5] Bora S., Paily R. A High-Performance Core Micro-Architecture Based on RISC-V ISA for Low Power Applications // IEEE Transactions on Circuits and Systems. 2021. №6.

[6] Gokulan T, Akshay Muraleedharan, Kuruvilla Varghese Design of a 32-bit, dual pipeline superscalar RISC-V processor on FPGA // Euromicro Conference on Digital System Design. 2020.

[7] C. Arul Rathi, G. Rajakumar Design and Development of an Efficient Branch Predictor for an In-order RISC-V Processor // Journal Of Nano- And Electronic Physics. 2020. №5.

[8] Харрис Д.М., Харрис С.Л. Цифровая схемотехника и архитектура компьютера: RISC-V. Elsevier, 2021.

[9] Don Kurian D. Single Cycle RISC-V Micro Architecture Processor and its FPGA Prototype // Seventh International Symposium on Embedded Computing and System Design. 2017.

METHODS TO IMPROVE THE PERFORMANCE OF RISC-V ARCHITECTURE PROCESSOR

I. Storozhev, E. Pevtsov

Centre for Design of Integrated Circuits, Nanoelectronics and Microsystems Devices,
MIREA – Russian Technological University, Moscow, Russian Federation, storozhev9003@gmail.com

Abstract: The paper presents a microarchitecture of a processor with RISC-V architecture supporting a limited number of instructions from the RV32IM instruction set. Development and functional verification of the prototype in the hardware description language SystemVerilog, as well as static timing analysis for clock frequency estimation in Xilinx Vivado CAD. Methods to improve processor performance were applied: data path pipelining, result forwarding (bypass), static conditional transition predictor, synchronous multiplier, adder with accelerated carry.

Keywords: processor, RISC-V, microarchitecture, pipelining, synchronous multiplier, accelerated carry adder.