DOI: 10.22184/1992-4178.2024.232.1.78.86

Эффективный подход в разработке управляющих автоматов микропроцессорных ядер

А. В. Строгонов, д. т. н.[1], О. Бордюжа, к. т. н.[2], А. И. Строгонов[3]

УДК 004.94 | ВАК 2.2.2

В России при участии НИУ МИЭТ организована Ассоциация вузов ЭКБ, решающих в том числе задачу подготовки специалистов для работы с архитектурой процессоров RISC-V, а также создана Школа синтеза цифровых схем на базе курса MIT в Сколково для быстрого освоения современных подходов к проектированию цифровых БИС. Архитектура RISC-V является дальнейшим развитием архитектуры MIPS, разработанной компанией MIPS Computer Systems. На архитектуре MIPS32 построен ряд российских процессоров, таких как Baikal-T от «Байкал Электроникс», «Мультикор» от АО НПЦ «ЭЛВИС» и КОМДИВ от НИИСИ РАН. Процессоры на базе RISC-V разрабатывают несколько российских компаний, в том числе Syntacore и CloudBEAR. В статье рассмотрен эффективный подход к разработке управляющего автомата микропроцессорного ядра с применением системы визуально-имитационного моделирования Matlab/Simulink и последующей генерацией VHDL-кода для разработки проекта в САПР Quartus II.

Предлагаемый читателям микропроцессор не относится к архитектуре MIPS, но как пример может быть полезен для наполнения курса лекций Школы синтеза цифровых схем. Простейший процессор с однотактной архитектурой из работы [1] состоит из двух взаимодействующих частей: тракт данных (память, регистры, АЛУ и мультиплексоры) и управляющий автомат (обычно представлен HDL-кодом), который получает текущую команду из тракта данных и в ответ сообщает ему, как именно выполнять эту команду. В частности, управляющий автомат генерирует адресные сигналы для мультиплексоров, сигналы разрешения работы для регистров и сигналы разрешения записи в память.

[1] Воронежский государственный технический университет, профессор кафедры твердотельной электроники, тел.: +7 910 247-14-70, andreistrogonov@mail.ru.

[2] Воронежский государственный лесотехнический университет им. Г. Ф. Морозова, факультет компьютерных наук и технологий, кафедра вычислительной техники и информационных систем, старший преподаватель.

[3] Воронежский государственный университет, факультет компьютерных наук, кафедра программирования и информационных технологий, ассистент.

Процесс разработки микропроцессорного ядра рассмотрим на примере двух проектов. **Проект 1** показывает применение системы визуально-имитационного моделирования Matlab / Simulink [2] с последующей генерацией VHDL-кода для разработки управляющего автомата в САПР Quartus II. **Проект 2** демонстрирует разработку микропроцессорного ядра непосредственно в САПР Quartus II с использованием редактора конечных автоматов State Machine Editor.

ОБЩИЕ СВЕДЕНИЯ ПО ПРОЕКТИРУЕМОМУ ПРОЦЕССОРУ

Воспользуемся системой команд гипотетического синхронного процессора, реализованного с помощью конечного автомата, с циклом работы в два такта [3]. В табл. 1 представлена система команд процессора с синхронной архитектурой. Процессор основан на использовании раздельных шин данных и команд.

Процессор ограничен двумя регистрами общего назначения (A и B), содержит указатель инструкций IP (счетчик команд) и регистр R для хранения адреса, с которого произошел вызов подпрограммы, поддерживает минимальный набор из 16 команд: команда пересылки «регистр-регистр» (XCHG), команды непосредственной загрузки (MOV), команда безусловного перехода к новому

адресу (JMP), команды перехода по условию (JMPZ), набор арифметико-логических операций (ADD, SUB, AND, OR, XOR, DEC), команда обращения к подпрограммам CALL и команда возврата из подпрограмм RET [3].

ПРОЕКТ 1

В работах [4, 5] было показано, как с применением системы визуально-имитационного моделирования Matlab / Simulink разработать микропроцессорное ядро для реализации в базисе ПЛИС. Основная идея работы [5] – это реализация управляющего автомата микропроцессорного ядра с использованием визуально-графического представления в приложении StateFlow системы Matlab / Simulink [6]. StateFlow является интерактивным инструментом разработки в области моделирования сложных, управляемых событиями систем. Он тесно интегрирован с Matlab и Simulink и основан на теории конечных автоматов [6].

Микропроцессор состоит из следующих блоков (рис. 1): ROM – ПЗУ команд (память программ); COP – блок выделения полей команды (дешифратор команд); ALU – 8-разрядное АЛУ (управляющий автомат); RON – блок регистров общего назначения (8-разрядные регистры A и B); RSN – блок регистров специального назначения, 8-разрядный регистр R (стек команд) для обеспечения выполнения команд обращения к подпрограммам (CALL) и возврата (RET) и 8-разрядный регистр IP (для хранения значений счетчика команд). Более подробное описание всех блоков можно найти в работах [4, 5]. На рис. 1 также представлена тестовая программа (прошивка), хранящаяся в ПЗУ. На рис. 2 продемонстрирован проект управляющего автомата микропроцессора, разработанный с помощью приложения StateFlow.

Далее с использованием Simulink HDL Coder системы Matlab / Simulink извлекается VHDL-код всех функциональных блоков микропроцессора (ПЗУ, дешифратора команд, регистров RON, RSN и ALU), а затем реализуется проект с асинхронным ПЗУ и с синхронным управляющим автоматом в базисе ПЛИС Cyclone V в САПР Quartus II.

Таблица 1. Система команд процессора с синхронной архитектурой [3]

Код операции	Мнемоника	Описание
0	NOP	Нет операции
01xxH	JMP	Безусловный переход по адресу, заданному младшим байтом команды
02xxH	JMPZ	Переход по адресу, заданному младшим байтом команды, если содержимое регистра A равно нулю
03xxH	CALL	Вызов подпрограммы по адресу, заданному младшим байтом команды
04xxH	MOV A, xx	Непосредственная загрузка в регистр A значения, заданного младшим байтом команды
05xxH	MOV B, xx	Непосредственная загрузка в регистр B значения, заданного младшим байтом команды
0600H	RET	Возврат из подпрограммы
0601H	MOV A, B	Загрузка в регистр A значения, содержащегося в регистре B
0602H	MOV B, A	Загрузка в регистр B значения, содержащегося в регистре A
0603H	XCHG A, B	Обмен местами значений в регистрах A и B
0604H	ADD A, B	Сложение значений в регистрах A и B, результат помещается в A
0605H	SUB A, B	Вычитание значений в регистрах A и B, результат помещается в A
0606H	AND A, B	Побитное логическое «И» значений в регистрах A и B, результат помещается в A
0607H	OR A, B	Побитное логическое «ИЛИ» значений в регистрах A и B, результат помещается в A
0608H	XOR A, B	Побитное логическое «Исключающее ИЛИ» значений в регистрах A и B, результат помещается в A
0609H	DEC A	Декремент регистра A (вычитание 1)

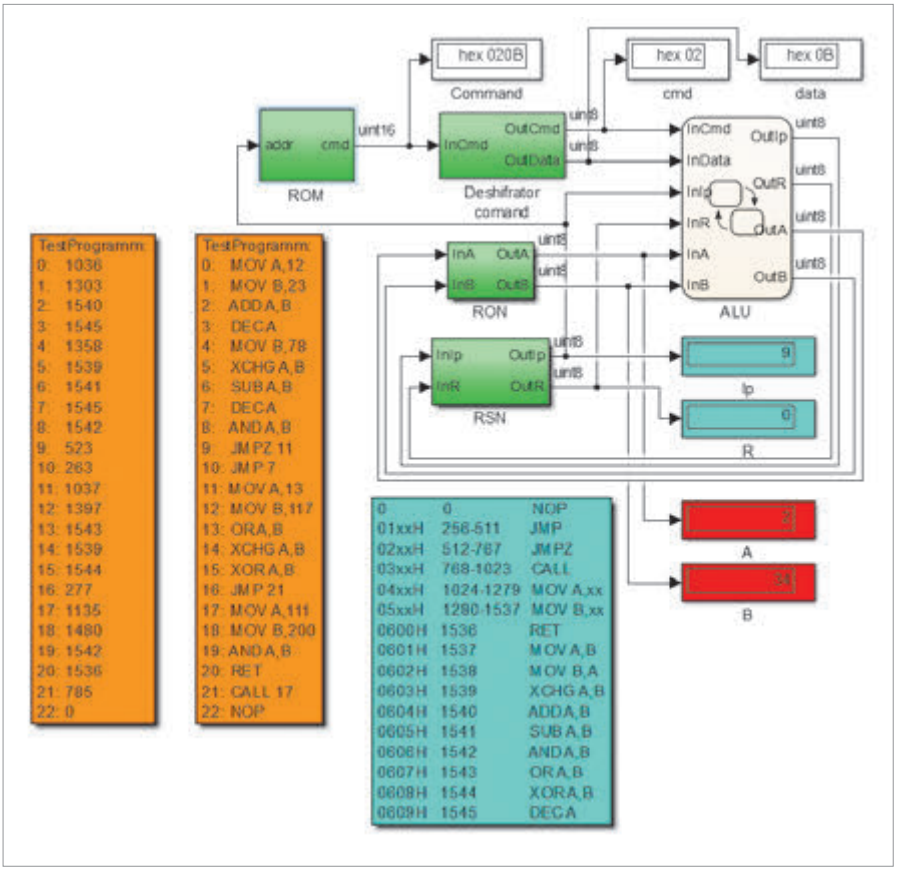


Рис. 1. Модель микропроцессорного ядра с управляющим автоматом в системе Matlab/Simulink, тестовая программа (мнемонические коды и их десятичное представление) и система команд

ПРОЕКТ 2

Была поставлена задача реализовать с помощью редактора конечных автоматов (State Machine Editor) в САПР Quartus II управляющий автомат микропроцессорного ядра, функционирование которого бы полностью соответствовало диаграмме рис. 2. ПЗУ и дешифратор команд реализованы VHDL-кодом в **Проекте 1**. Остальные блоки реализуются на логических элементах или с использованием мегафункций (сумматор/вычитатель, шинный мультиплексор). Для детального ознакомления **Проект 2** можно запросить у авторов работы.

Вследствие ограниченного функционала редактор конечных автоматов не позволяет, в отличие от приложения StateFlow системы Mathlab/Simulink, проводить над выходными сигналами состояний какие-либо операции, приводящие к изменению их значений, то есть они должны быть константами [7]. Например, действие увеличения содержимого счетчика команд на единицу OutIP=InIP+1 в состоянии INST после выполнения очередной команды не допустимо, то есть невозможно прибавить 1 к сигналу InIP, но возможно простое присваивание OutIP=InIP. Для преодоления этого ограничения были введены дополнительные сигналы ADD_IP (0 – нет увеличения содержимого счетчика команд, 1 – увеличить

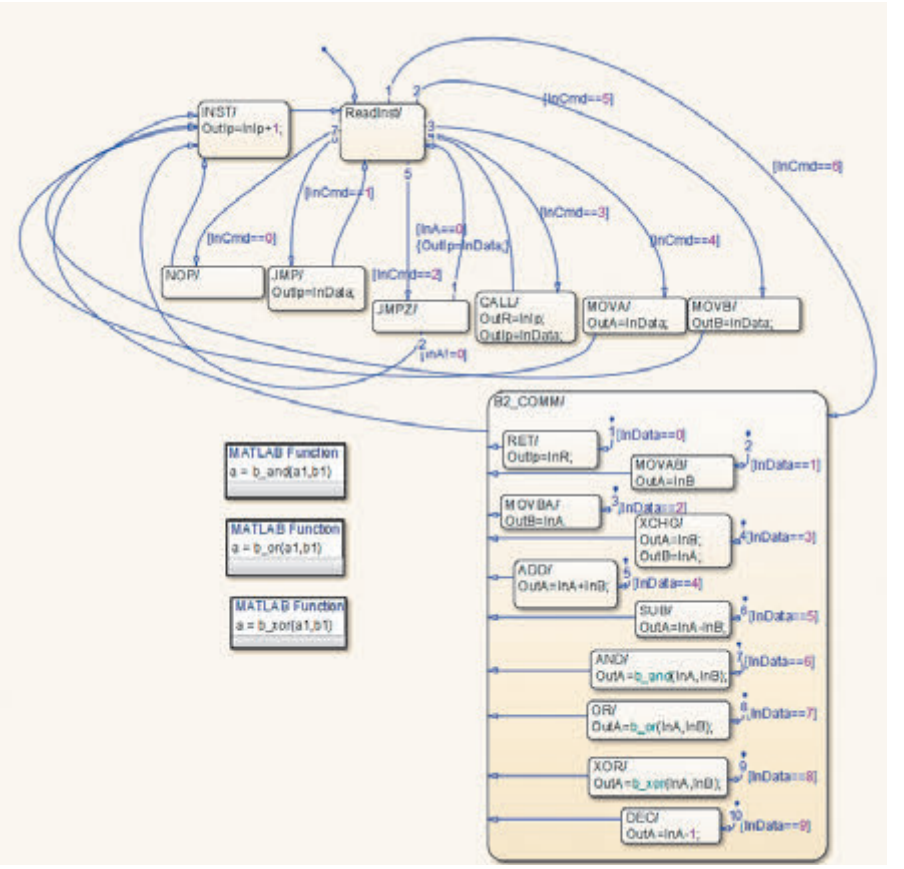


Рис. 2. Управляющий автомат (блок АЛУ), созданный с помощью приложения StateFlow

Таблица 2. Условия переходов по состояниям

Переход от состояния	К состоянию	Условия	Переход от состояния	К состоянию	Условия
ReadInst	NOP	InCmd==0	ReadInst	XCHG	InCmd==6&InData==3
ReadInst	JMP	InCmd==1	ReadInst	ADD	InCmd==6&InData==4
ReadInst	JMPZ	InCmd==2	ReadInst	SUB	InCmd==6&InData==5
ReadInst	CALL	InCmd==3	ReadInst	Bit_AND	InCmd==6&InData==6
ReadInst	MOVA	InCmd==4	ReadInst	Bit_OR	InCmd==6&InData==7
ReadInst	MOVB	InCmd==5	ReadInst	Bit_XOR	InCmd==6&InData==8
Inst	ReadInst	Безусловный	ReadInst	DecrementA	InCmd==6&InData==9
ReadInst	RET	InCmd==6&InData==0	ReadInst	Bit_XNOR	InCmd==6&InData==10
ReadInst	MOVAB	InCmd==6&InData==1	JMPZ	JMPZ_TEMP	InA==0
ReadInst	MOVBA	InCmd==6&InData==2	JMPZ	Inst	~(InA==0)

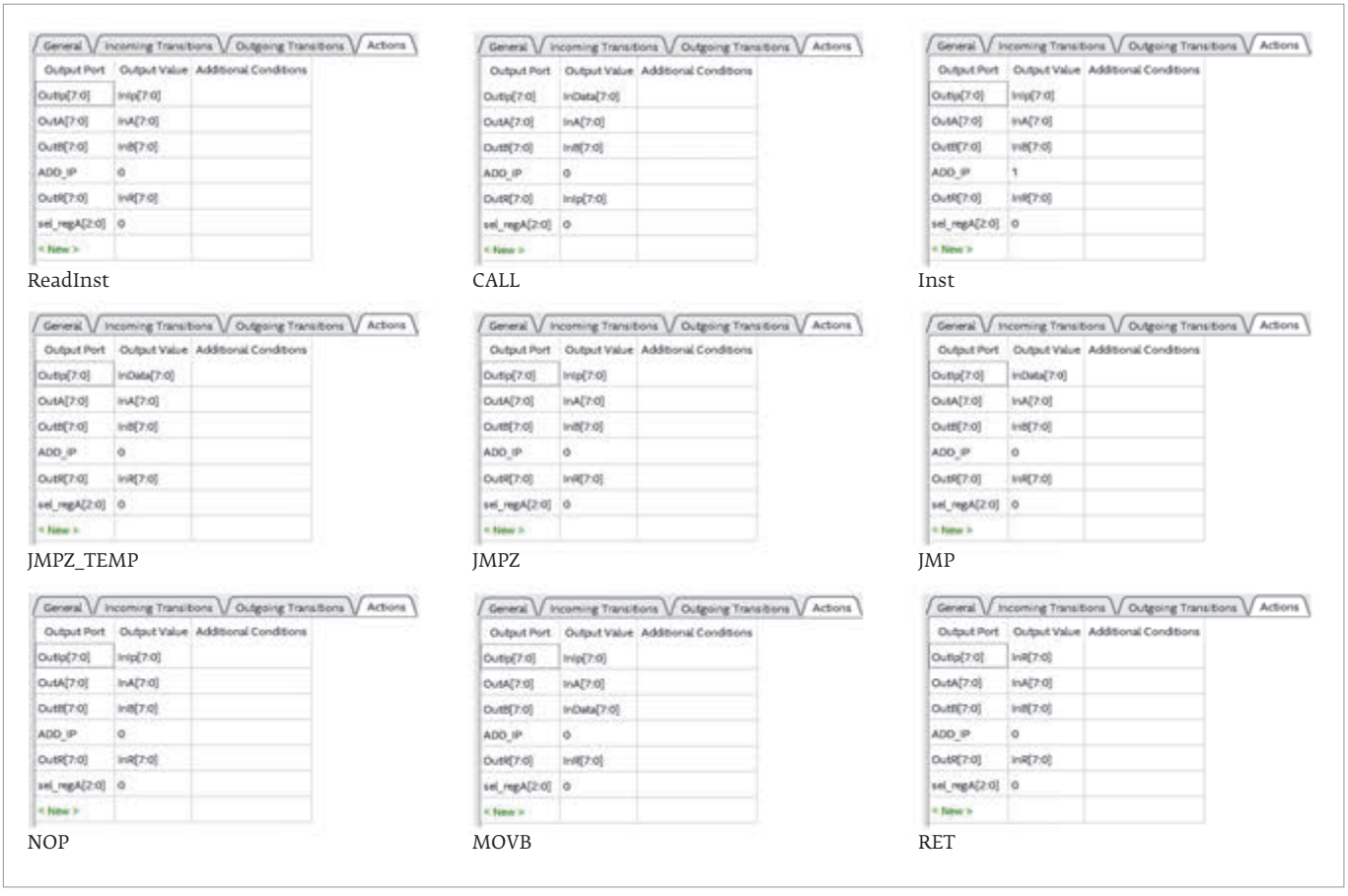


Рис. 3. Информационные потоки в состояниях ReadInst, CALL, Inst, JMPZ_TEMP, JMPZ, JMP, NOP, MOVB, RET

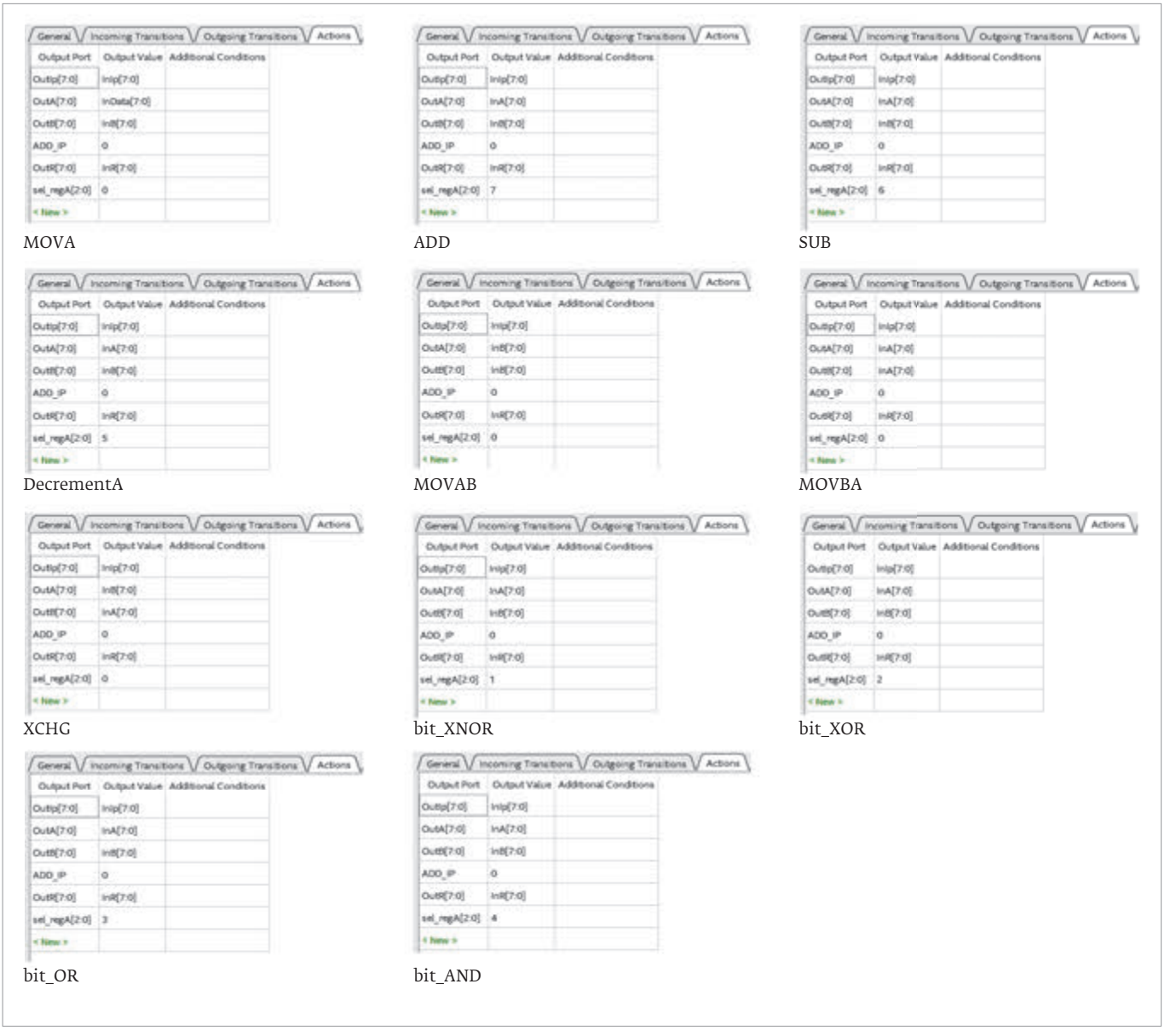


Рис. 4. Информационные потоки в состояниях MOVA, ADD, SUB, DecrementA, MOVAB, MOVBA, XCHG, bit_XNOR, bit_XOR, bit_OR, bit_AND

содержимое на 1) и sel_RegA[2..0]. Сигнал ADD_IP принимает значение логической 1 в одном единственном состоянии Inst. Сигнал sel_RegA[2..0] принимает восемь значений от 0 до 7.

ОПЕРАЦИИ С АККУМУЛЯТОРОМ (РЕГИСТР А)

В предложенном варианте реализации **Проекта 2** логико-арифметические операции над сигналами A и B (семь команд) и все остальные команды, связанные с регистром A, реализуются на функциональном блоке (назовем его распределенный аккумулятор), который представляет собой восемь регистров, подключенных к информационным входам шинного мультиплексора 8 в 1. Многоразрядный сигнал sel_RegA[2..0] является адресным сигналом шинного мультиплексора 8 в 1, который коммутирует один из его входов на вход InA[7..0] управляющего автомата. В состояниях, связанных с битовыми операциями bit_AND, bit_OR, bit_XOR, bit_XNOR, сигнал sel_RegA[2..0] принимает значения 4, 3, 2 и 1. Битовая команда XNOR (код 0610H) была введена дополнительно для возможного использования шинного мультиплексора 8 в 1. Она отсутствует в **Проекте 1** (см. табл. 1).

В состояниях, связанных с арифметическими операциями ADD, SUB, DecrementA, сигнал sel_RegA[2..0] принимает значения 7, 6 и 5. Во всех остальных состояниях – CALL, JMPZ_TEMP, JMPZ, JMP, NOP, RET, MOV A, MOV B,

MOVAB, MOVBA, XCHG и таких, как ReadInst (чтение команды), Inst (в том числе было введено вспомогательное состояние JMPZ_TEMP, отсутствующее в **Проекте 1**), 3-разрядный сигнал sel_RegA[2..0] принимает значение логического нуля. Условия переходов по состояниям показаны в табл. 2. На рис. 3 и 4 показаны информационные потоки во всех состояниях управляющего автомата.

Рассмотрим организацию регистра A, выполняющего роль распределенного аккумулятора в микропроцессоре. Логические (битовые) операции bit_AND, bit_OR, bit_XOR, bit_XNOR реализованы на вентилях. Арифметические операции ADD, SUB и DecrementA реализуются на мегафункции сложения / вычитания (LPM_ADD_SUB). Для арифметических операций требуется 8-разрядный сумматор (A+B), вычитатель (A-B) и декрементатор (A-1). К выходам блоков, осуществляющих логические и арифметические операции (можно рассматривать как АЛУ), подключено семь 8-разрядных регистров. А все остальные операции, не связанные с АЛУ, осуществляются на отдельном 8-разрядном регистре. Выходы всех восьми регистров подключены к шинному мультиплексору 8 в 1, адресным входом которого управляет автомат (sel_RegA[2..0]). Регистры B и R микропроцессора реализованы на отдельных 8-разрядных регистрах.

Операция увеличения содержимого счетчика команд на 1 (состояние Inst) реализуется с помощью сумматора, на один из входов которого подается логическая единица, и регистра для запоминания результата, подключенного к одному из входов мультиплексора 2 в 1. Если на адресном входе мультиплексора ADD_IP=1 (выходной сигнал управляющего автомата), то результат суммирования будет передан для дальнейшей обработки (для выбора адреса следующей команды), в противном случае при ADD_IP=0 должен быть произведен обход операции суммирования (например, для команды CALL).

На рис. 5 представлен граф-автомат микропроцессора на 20 состояний, созданный с помощью редактора конечных автоматов (State Machine Editor). На рис. 6 показаны выходные сигналы управляющего автомата (OUT_A, OUT_B, OUT_R, OUT_IP, ADD_IP и sel_regA[2..0]), а также поля команды inCmd и inData при функциональном моделировании работы микропроцессорного ядра. Видим, что разработанное микропроцессорное ядро корректно отрабатывает коды тестовой программы (см. рис. 1).

Разместим полученный VHDL-проект в ПЛИС Cyclone V 5CGXFC7C7F23CB, в табл. 3 показаны используемые ресурсы ПЛИС для трех проектов.

ЗАКЛЮЧЕНИЕ

В работе установлено, что приложение StateFlow системы Matlab / Simulink обладает большим функционалом по разработке управляющих автоматов

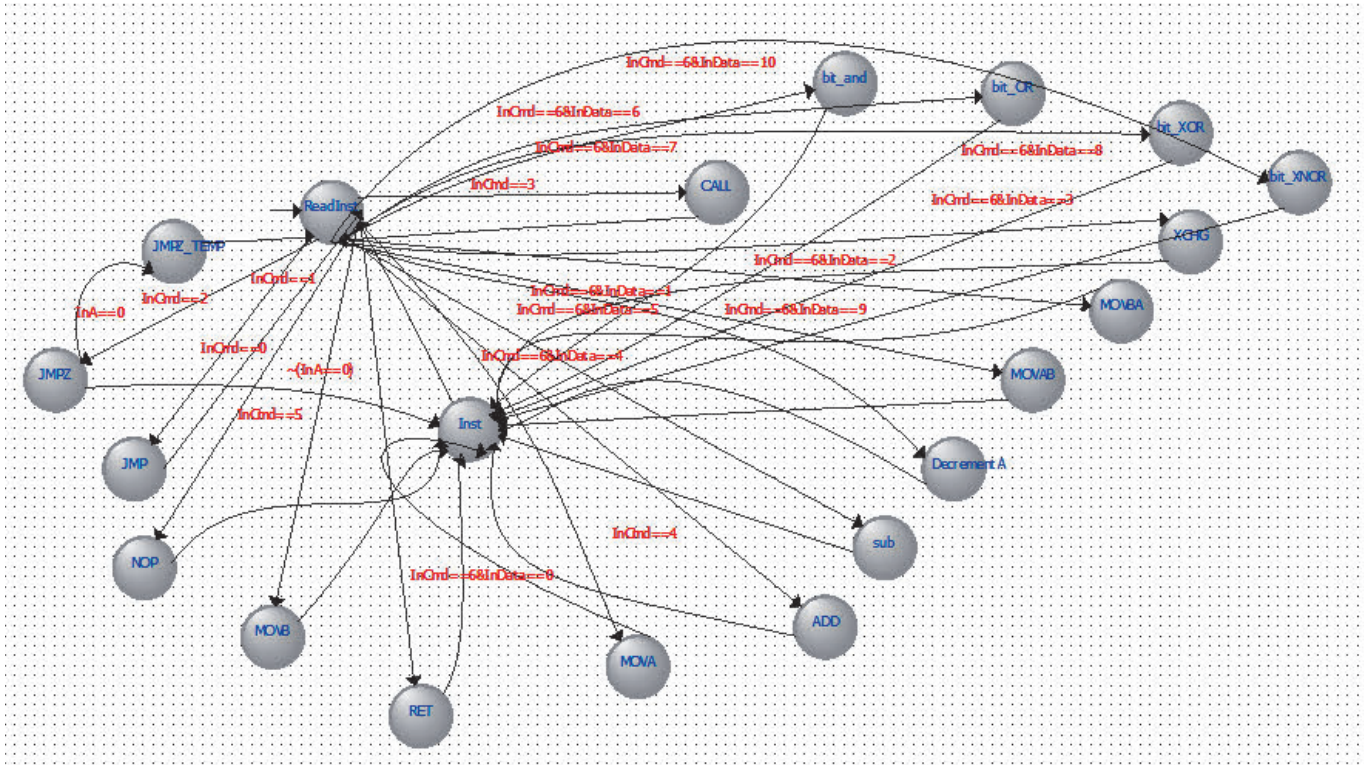


Рис. 5. Граф-автомат микропроцессора, созданный с помощью State Machine Editor САПР Quartus II

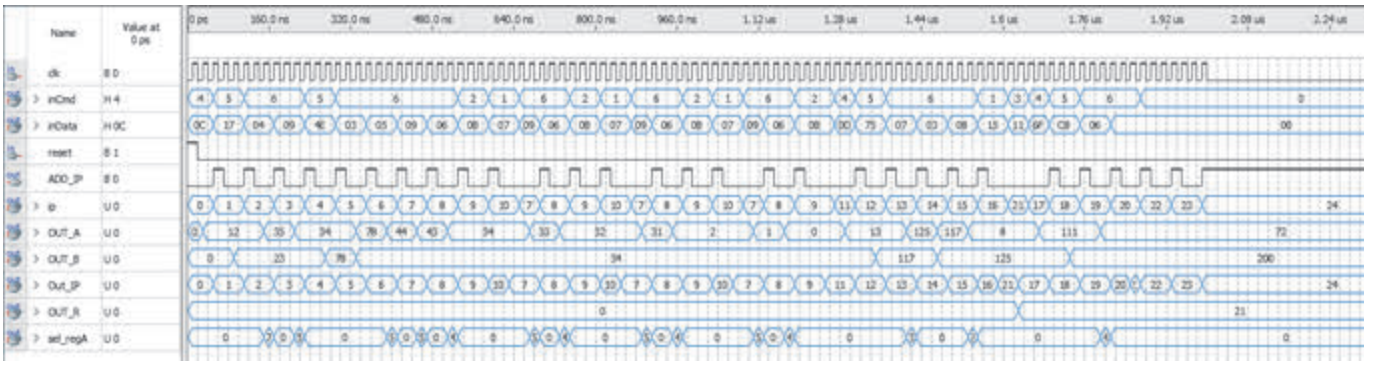


Рис. 6. Функциональное моделирование работы микропроцессорного ядра (Проект 2)

микропроцессорных ядер, чем редактор состояний State Machine Editor САПР Quartus II. Так в **Проекте 2** функциональный блок АЛУ из-за ограниченного функционала State Machine Editor пришлось разрабатывать в ручном режиме, путем введения дополнительных сигналов и вспомогательных состояний, что сказалось на незначительном повышении числа использования логических ресурсов ПЛИС. В целом удалось получить приемлемое быстродействие. Разработанный **Проект 2** раскрывает механизм обработки информационных потоков в микропроцессорном ядре и более детально проработан на функциональном уровне.

Приложение Simulink HDL coder для управляющего автомата микропроцессора, построенного с помощью StateFlow системы Matlab / Simulink, позволяет сгенерировать код языка VHDL синхронного автомата, при этом получаем оптимальное число используемых логических ресурсов ПЛИС и выигрыш по быстродействию по сравнению с **Проектом 2.**

Система визуально-имитационного моделирования Matlab / Simulink с приложениями StateFlow и Simulink HDL Coder может быть эффективно использована для ускорения процесса разработки моделей микропроцессорных ядер. Проект микропроцессора с асинхронным ПЗУ на языке VHDL может быть успешно реализован в ПЛИС Cyclone V 5CGXFC7C7F23CB, при этом общее число задействованных логических ресурсов составляет менее 1%.

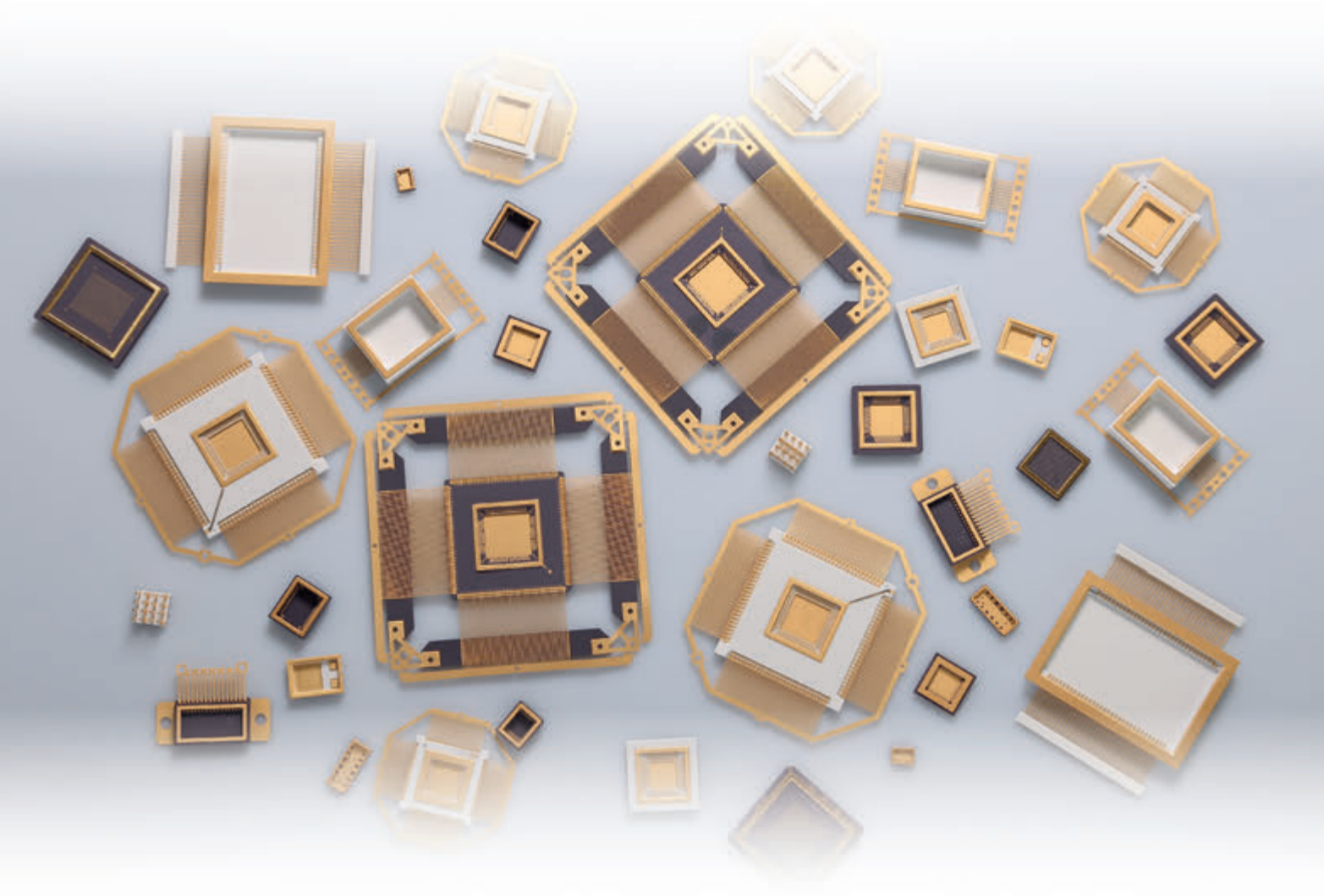
Таблица 3. Используемые ресурсы ПЛИС Cyclone V 5CGXFC7C7F23CB

Проект	Логические ресурсы			Максимальная тактовая частота синхросигнала, f_{maxCLK}, МГц (модель Slow 1100 мВ, 85 °С)
	Адаптивные логические модули (ALM)	Адаптивные таблицы перекодировок (ALUT), для реализации комбинационных функций	Выделенные триггеры логических элементов	
Управляющий автомат создан с помощью StateFlow (асинхронное ПЗУ)	92	135	37	146
Управляющий автомат создан с помощью State Machine Editor (асинхронное ПЗУ)	162	237	114	130
Микропроцессорное ядро полностью описано VHDL-кодом (взято из [4])	98	143	33	145

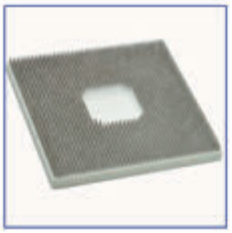
ЗАВОД
ПОЛУПРОВОДНИКОВЫХ
ПРИБОРОВ

ЙОШКАР-ОЛА, РЕСПУБЛИКА МАРИЙ ЭЛ

АКЦИОНЕРНОЕ ОБЩЕСТВО «ЗАВОД ПОЛУПРОВОДНИКОВЫХ ПРИБОРОВ»

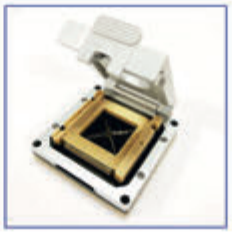


Выводные рамки



Металлокерамические корпуса

Нагревательные элементы



Контактные устройства

Графитовая оснастка

Оптоэлектронные корпуса

424003, Россия, Республика Марий Эл, г. Йошкар-Ола, ул. Суворова, 26
Тел.: +7-8362-45-70-09, 45-67-68.
info@zpp12.ru marketing@zpp12.ru

zpp12.ru

ЛИТЕРАТУРА

1. **Харрис Д. М., Харрис С. Л.** Цифровая схемотехника и архитектура компьютера / Пер. с англ. Imagination Technologies. М.: ДМК Пресс, 2018. 792 с.
2. **Строгонов А.** Проектирование конечных автоматов в приложении Stateflow системы Matlab/Simulink с последующей реализацией в базисе ПЛИС // ЭЛЕКТРОНИКА: Наука, Технология, Бизнес. 2023. № 3. С. 1–9.
3. **Тарасов И.** Проектирование конфигурируемых процессоров на базе ПЛИС. Часть 1 // Компоненты и технологии. 2006. № 2. С. 58–59.
4. **Строгонов А.** Проектирование учебного процессора для реализации в базисе ПЛИС // Компоненты и технологии. 2009. № 3. С. 118–121.
5. **Строгонов А., Буслов А.** Проектирование учебного процессора для реализации в базисе ПЛИС с использованием системы Matlab / Simulink // Компоненты и технологии, 2009. № 5. С. 114–118.
6. **Строгонов А., Буслов А.** Проектирование микропроцессорных ядер с использованием приложения StateFlow системы MATLAB / Simulink // Компоненты и технологии. 2010. № 1. С. 126–130.
7. **Van der Star P.** State machine editor & State machine wizard. Tutorial Quartus II. 29 April 2014 // https://ds.opdenbrouw.nl/quartus/state_machine_tutorial.pdf.

АКЦИОНЕРНОЕ ОБЩЕСТВО
ЭЛЕКОНД

РАЗРАБОТКА И ПРОИЗВОДСТВО КОНДЕНСАТОРОВ

оксидно-электролитические алюминиевые конденсаторы

К50-15, К50-17, К50-27, К50-29, К50-37, К50-68, К50-77, К50-80, К50-81, К50-83, К50-84, К50-85, К50-86, К50-87, К50-88, К50-89, К50-90, К50-91, К50-92, К50-93, К50-94, К50-95(чип), К50-96, К50-97(чип), К50-98, К50-99, К50-100, К50-101(чип), К50-102, К50-103, К50-104, К50-105, К50-106



объемно-пористые танталовые конденсаторы

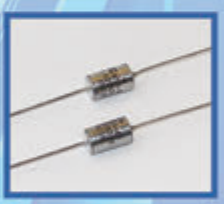
К52-1, К52-1М, К52-1БМ, К52-1Б, К52-9, К52-11, К52-17, К52-18, К52-19, К52-20, К52-21, К52-24, К52-26(чип), К52-27(чип), К52-28, К52-29, К52-30



оксидно-полупроводниковые танталовые конденсаторы

К53-1А, К53-7, К53-65(чип), К53-66, К53-68(чип), К53-69(чип), К53-71(чип), К53-72(чип), К53-74(чип), К53-77(чип), К53-78(чип), К53-82





суперконденсаторы (ионисторы)

К58-26, К58-27, К58-28, К58-29, К58-30, К58-31



накопители электрической энергии на основе модульной сборки суперконденсаторов

НЭЭ, МИК, МИЧ, ИТИ



Система менеджмента качества сертифицирована на соответствие требованиям ISO 9001

Россия, 427968, Удмуртская Республика,г. Сарапул, ул. Калинина, 3
Тел.: (34147) 2-99-53, 2-99-89, 2-99-77, факс: (34147) 4-32-48, 4-27-53
e-mail: elecond-market@elcudm.ru, http://www.elecond.ru