УДК 004.31

СТРУКТУРНАЯ ОРГАНИЗАЦИЯ БЛОКА АССОЦИАТИВНОГО СОПРОЦЕССОРА НА БАЗЕ ПРОГРАММИРУЕМОЙ ЛОГИЧЕСКОЙ ИНТЕГРАЛЬНОЙ СХЕМЫ ДЛЯ СПЕЦИАЛИЗИРОВАННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ[1]

А. И. Мартышкин

THE STRUCTURAL ORGANIZATION OF THE MODULE OF THE ASSOCIATIVE COPROCESSOR BASED ON PROGRAMMABLE LOGIC INTEGRATED CIRCUIT FOR SPECIALIZED COMPUTING SYSTEMS

A. I. Martyshkin

Аннотация. *Актуальность и цели.* Основная область применения электронных вычислительных машин (ЭВМ) и создаваемых на их основе вычислительных систем (ВС) и комплексов сегодня – работа с крупными объемами данных, при которой наиболее трудоемкими являются операции поиска и сортировки. Существующие ВС используют адресную архитектуру памяти, т.е. для того чтобы осуществить поиск данных в памяти, необходимо произвести чтение по конкретному адресу модуля памяти и сравнить его с аргументом поиска. В конечном итоге на поиск нужной информации в памяти уходят сотни, а то и тысячи машинных тактов. Это обстоятельство крайне негативно сказывается на быстродействии ВС в целом. Намного быстрее получать доступ к данным по ассоциации. В работе показан способ реализации блока ассоциативного сопроцессора для специализированных ВС. Целью данной работы является разработка и исследование структуры блока ассоциативного сопроцессора на базе программируемой логической интегральной схемы (ПЛИС) для специализированных вычислительных систем, выполняющего ассоциативные функции, а также функции хранения данных. Объектом разработки и исследования является ассоциативный сопроцессор на базе ПЛИС. *Материалы и методы.* Для достижения поставленных в работе целей применена САПР Web pack ISE фирмы Xilinx с возможностью создания и моделирования работы блока, что облегчает синтез проектов, ориентированных на использование современной элементной базы – ПЛИС. *Результаты.* Синтезированы и проверены VHDL-коды блока ассоциативного сопроцессора и его составных частей. Работоспособность блока сопроцессора проверена тестированием в используемой САПР Web pack ISE фирмы Xilinx, получены временные диаграммы, подтверждающие корректность представляемых результатов. *Выводы.* Результатами исследования являются полученные кроссплатформенные (для реализации на различных типах ПЛИС) VHDL-коды блока сопроцессора, из которых синтезирована битовая последовательность для конфигурирования ПЛИС.

Ключевые слова: ассоциативная память, блок аппаратного сопроцессора, вычислительная система, ячейка памяти, память фиксации реакций, адресация памяти, шинный интерфейс, цикл записи, цикл чтения.

Abstract. *Background.* The main applications of computers and created on the basis of their computational systems (CS) and complexes is the work with large amounts of data, which are the most time-consuming operations of searching and sorting. Existing CS used memory address architecture, ie, to realize the retrieval of data in the memory necessary to

[1] Работа выполнена при финансовой поддержке РФФИ (грант № 16-07-00012).

make reading at a particular address the memory module and compare it with the search argument. Ultimately, finding the desired information in memory takes hundreds, or even thousands of machine clock cycles. This situation is extremely negative impact on the aircraft speed in general. It is much faster to access data on the association. The paper shows a method for implementing associative coprocessor block for specialized computer systems. The purpose of this work is the development and research of associative coprocessor block structure on programmable logic integrated circuit (FPGA) for specialized computer systems, performing associative function and data storage functions. The object of research and development of the article is an associative coprocessor based on FPGA. *Materials and methods.* For achievement the objectives applied CAD Web pack ISE from Xilinx with the ability to create and simulate the operation of the unit, which facilitates the synthesis of projects focused on the use of modern electronic components – FPGAs. *Results.* Synthesized and tested VHDL-code of association coprocessor block and its components. The efficiency of coprocessor unit checked by testing in the CAD Web pack ISE the Xilinx Company, obtained the timing charts, confirming the correctness of the reported results. *Conclusions.* The results of the research are received crossplatform (for implementation in various types of FPGA) VHDL-code coprocessor unit, which is synthesized from the bit sequence to configure the FPGA.

Key words: associative memory, coprocessor module, computer system, memory cell, memory responses fixation, memory addressing, bus interface, write cycle, read cycle.

Введение

Сегодня основная область применения ЭВМ и создаваемых на их базе ВС и комплексов – работа с колоссальными объемами данных, при которой самыми трудоемкими являются операции поиска и сортировки данных. Существующие ВС используют адресную архитектуру памяти. В итоге на поиск нужной информации в памяти уходят сотни, а то и тысячи машинных тактов. Это обстоятельство крайне негативно сказывается на быстродействии ВС в целом. Намного быстрее можно получать доступ к данным по содержанию, сущность принципа которого заключается в следующем (рис. 1). Имеется массив данных емкостью N слов, где нужно найти все слова, начинающиеся с символа «А» и заканчивающиеся символом «Н». Аргументом поиска является слово А***Н, где знаком * помечены разряды, не влияющие на результат поиска. Запоминающий массив на аппаратном уровне строится так, что на выходе ячеек памяти (ЯП), содержимое которых совпадает со значением поступившего аргумента поиска, появляется сигнал – указатель совпадения. Далее по выработанным сигналам совпадения выполняется выборка ЯП.

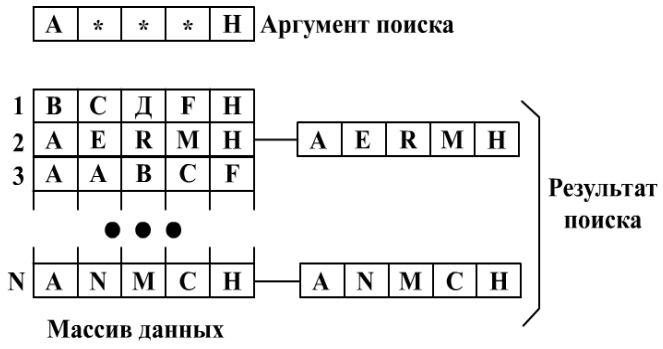


Рис. 1. Сущность принципа адресации по содержанию

1. Постановка задачи

Данная статья носит исследовательский характер. В ходе изучения рассматриваемой предметной области с целью поиска слабо проработанных, либо вовсе не затронутых вопросов были проанализированы публикации отечественных и зарубежных ученых [1–7]. Ряд проблемных вопросов, связанных с построением быстродействующего блока ассоциативного сопроцессора и его аппаратной реализацией, не нашел должного отражения в изданиях по данной тематике. Частично они рассмотрены в публикациях [8–16]. Целью статьи является разработка и исследование структуры блока ассоциативного сопроцессора на базе ПЛИС для специализированных ВС. Сформулированная тематика работы является актуальной ввиду глобальной информатизации. Для достижения обозначенной цели здесь решаются задачи по определению структурной организации устройства, алгоритмов его работы и возможности реализации блока аппаратного ассоциативного сопроцессора, который имеет возможность адресного и ассоциативного доступа к данным, хранящимся в памяти.

Устройство в совокупности состоит из двух частей: основной, реализующей функции сопроцессора, и части сопряжения с ВС [13].

Сегодня ассоциативный доступ к данным реализуется двумя способами: программным и аппаратным [1, 4]. Он может быть реализован в виде параллельного ассоциативного запоминающего устройства (АЗУ), где аргумент поиска поступает во все ЯП параллельно. В результате выполняется массовая операция сравнения и поиск осуществляется за один такт. Другим вариантом реализации аппаратного способа ассоциативного доступа к данным является последовательно-поразрядное АЗУ с поразрядным поиском. Здесь время поиска зависит от количества разрядов шины данных.

Процессы, аналогичные биологическим механизмам запоминания и обработки информации, можно представить моделями ассоциативной памяти (АП), позволяющими отобразить ассоциации между информационными объектами. Все эти ассоциации могут реализоваться в виде простых конструкций – троек компонентов: упорядоченной пары информационных объектов O и V, и типа отношения A: $O \xleftrightarrow{A} V$ [4]. Одна из простейших моделей АП для отображения таких отношений показана на рис. 2,*а*. Модель состоит из ассоциативной запоминающей среды, связанной с двумя каналами ввода и одним каналом вывода информации. На этапе записи с первого канала ввода на вход K подается входная информация, а по второму каналу – признаковая информация C, представляющая контекст, где входная информация записывается в память.

На этапе ассоциативной выборки при появлении ключа K на выходе памяти формируется ответная реакция R, связанная с K. Таким образом, записанная в память информация может выбираться с применением любых ее фрагментов, используемых в качестве поисковых. Задавая различный контекст C, можно конкретизировать информацию, подлежащую выборке. В контексте введенного определения АП стоит вопрос об организации накопления и поиска структурированных данных таким образом, чтобы доступ к ним был возможен на основе ассоциативной выборки. На рис. 2,*б* представлена модель АП, показывающая, как можно организовать запись и выборку

элементов информации, а также осуществить цикличность процесса поиска, когда выбранный элемент информации становится ключом для поиска новой информации [4].

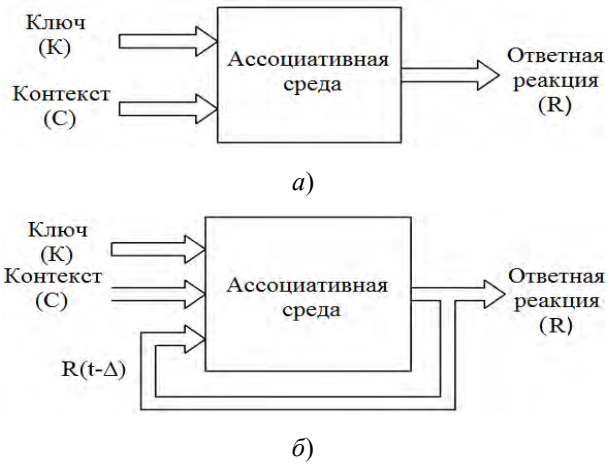


Рис. 2. Модель ассоциативной памяти: без обратной связи (*а*), с обратной связью (*б*)

По трем входным каналам одновременно могут вводиться наборы значений. Выходной канал служит для выборки информации. По первому каналу в момент времени t подается адресная информация $K(t)$, по второму – признак $C(t)$. Отклик $R(t)$ по каналу обратной связи подается также на вход ассоциативной среды. При функционировании такой АП ключи $K(t)$ и признаки $C(t)$ подаются через интервалы времени, соответствующие задержке канала обратной связи.

Процесс работы памяти будем рассматривать в предположении, что тройка $[K(t), C(t), R(t - \Delta)]$ представляет собой единый статический образ, заданный в момент времени t, причем возможна его одновременная запись в память за одну операцию. Допустим также, что на этапе записи $R(t)$ и $K(t)$ одинаковы.

На этапе записи на входы АП поступают $K(t)$ и $C(t)$, при этом на выходе формируется $R(t)$, идентичный $K(t)$. Затем с задержкой Δ на входе формируется $R(t - \Delta)$. Каждая новая тройка, появляющаяся на входах, записывается в память. На этапе выборки из АП на вход подается ключ K, связанный с контекстом C, после чего K можно снять с входа. В результате на выходе в качестве отклика появляется копия K. Когда на входе памяти появится задержанный сигнал $R(t - \Delta)$, новым ключом становится пара (C, R), приводящая к ассоциативной выборке следующего образа $R(t)$ и т.д. Таким образом выбирается вся записанная последовательность образов вместе с контекстом. Рассмотренная модель реализует память, пригодную для записи и выборки структурированной информации.

Подключение описываемого блока к ВС возможно рядом способов [17]: к шине центрального процессора, к интерфейсу USB, к шине PCI. При подключении к шине ЦП устройство придется включать в состав системной платы, что приведет к увеличению стоимости сопроцессора. Быстродействие

такой ВС окажется высоким. При подключении с помощью интерфейса USB сопроцессор будет иметь вид отдельного внешнего блока, при этом работа с ним будет осуществляться последовательно, что приведет к снижению быстродействия ВС. Однако такой блок сравнимо дешевле предыдущего. Подключение сопроцессора к шине PCI позволит реализовать его в виде сравнительно дешевой платы расширения в отличие от упомянутых аналогов. Работа с блоком будет осуществляться по параллельному интерфейсу, что позволит достичь максимальной производительности по сравнению с аналогами.

Проанализировав достоинства и недостатки перечисленных способов организации основной части устройства и части сопряжения с ВС, решили основную часть выполнить в виде параллельного АЗУ, так как данный способ обладает максимальной производительностью. Подключение к специализированной ВС реализуется по шине PCI, так как она обладает достаточно высокой пропускной способностью. Более того, блок ассоциативного сопроцессора будет размещаться в адресном пространстве ввода/вывода ВС.

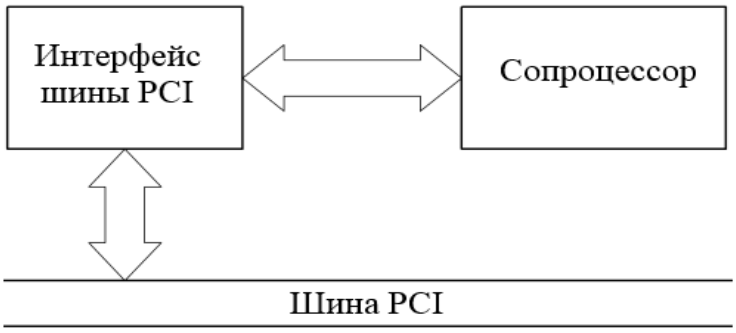
2. Определение структуры блока ассоциативного сопроцессора

Упрощенную структурную схему сопроцессора можно представить в виде двух частей (рис. 3,*а*): сопроцессор (основная часть) и интерфейс шины PCI (часть сопряжения). Проанализировав выбранный способ реализации основной части в составе сопроцессора, решили, что главным блоком является модуль памяти АЗУ, представляющий собой массив ЯП и выполняющий функции хранения данных и поиска ассоциаций с аргументом. ЯП состоит из запоминающего элемента, выполняющего функции хранения данных, и схемы сравнения, осуществляющей поиск, т.е. вырабатывающей сигналы, указывающие на равенство или неравенство содержимого ЯП аргументу (рис. 3,*б*).

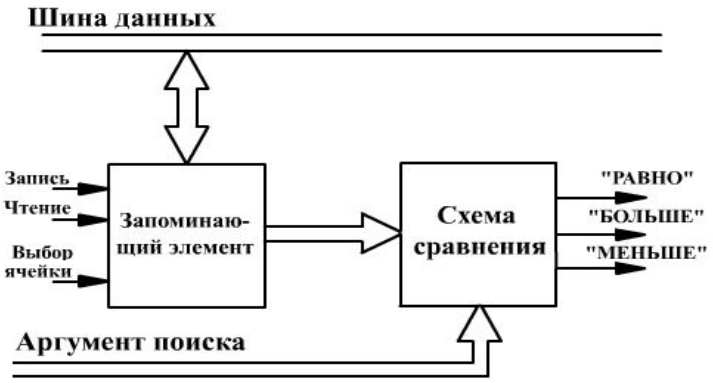
Для реализации функций записи и чтения данных из ЯП в состав сопроцессора нужно добавить блок мультиплексора и дешифратора адреса для выработки сигнала «Выбор ячейки» конкретной ЯП. Также необходимо включить в состав блока сопроцессора модуль памяти фиксации реакций (ПФР) для хранения значений откликнувшихся ЯП. Для подсчета количества откликнувшихся ЯП в состав сопроцессора введен блок подсчета совпадений. Блок анализатора многократных совпадений (АМС) необходим для приоритетного выбора, по результатам работы которого вырабатываются сигналы для блока шифратора, который производит преобразование двоичной последовательности в адрес, при ассоциативном чтении или записи поступающий на входы селектора адреса. Для хранения аргумента поиска в блоке имеется регистр аргумента. Управляет работой сопроцессора блок дешифрации команд, вырабатывающий управляющие сигналы (рис. 3,*в*).

В работе [13] рассмотрена функциональная организация блока сопроцессора. Здесь подробнее опишем алгоритмы, согласно которым работают некоторые составные части блока ассоциативного сопроцессора и устройство в целом.

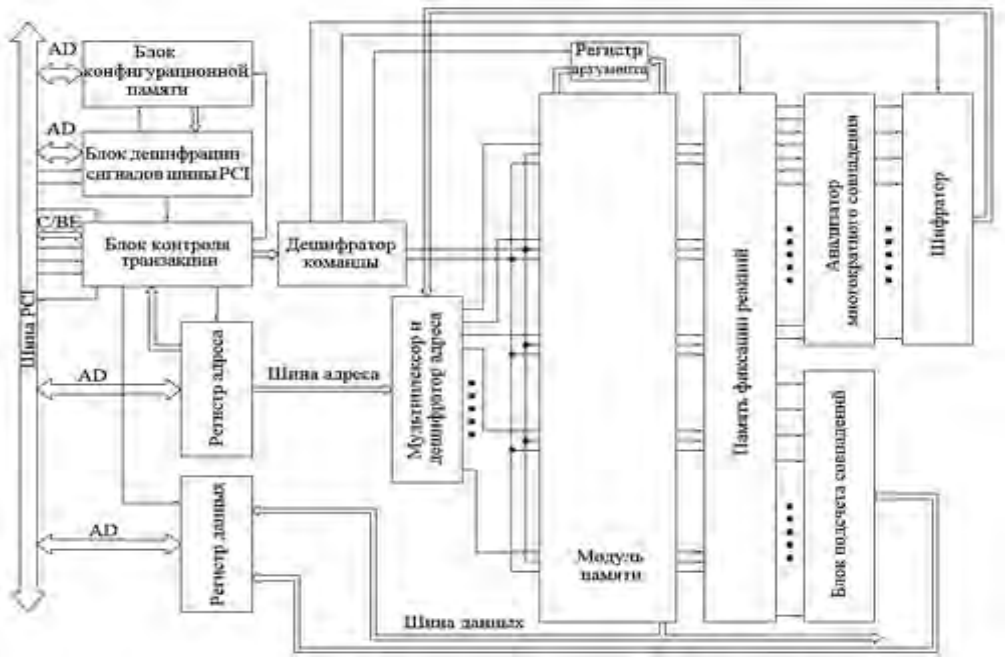
В сопроцессор команды поступают по отдельной внутренней шине – шине команд, которые дешифрируются и подаются на составные блоки устройства. Команды сопроцессора приведены в табл. 1.



а)



б)



в)

Рис. 3. Структурная схема: упрощенная блока ассоциативного сопроцессора (*а*); ячейки памяти (*б*); подробная блока ассоциативного сопроцессора (*в*)

Таблица 1

Команды ассоциативного сопроцессора

Команда	Описание команды
0000	readRAM – адресное чтение
0001	WriteRAM – адресная запись
0010	WrArg – запись аргумента поиска в регистр аргумента
0011	FixHit – команда фиксации откликнувшихся ячеек
0100	rdAEqual – выдача адреса ячейки, содержимое которой равно аргументу
0101	rdAMore – выдача адреса ячейки, содержимое которой больше аргумента
0110	rdALess – выдача адреса ячейки, содержимое которой меньше аргумента
с 0111 по 1111	зарезервировано

Дешифратор команды реализован на базе дешифратора, к четырехразрядному входу которого подключена шина команд. По команде readRAM выполняется функция адресного чтения, т.е. декодируется адрес, поступивший на шину адреса, на вход CS соответствующей ЯП подается логическая единица и на выходную шину данных поступает содержимое этой ЯП. По команде WriteRAM выполняется функция адресной записи. Декодируется адрес, на вход CS и WRITE соответствующей ЯП подаются логические единицы. Сигналы с входной шины данных поступают на вход регистра ЯП, т.е. осуществляется запоминание. По команде WrArg сигналы с входной шины данных поступают на вход регистра аргумента. По команде FixHit происходит фиксация сигналов Equal, More и Less в ПФР, и на шине выдачи количества ЯП появляются сигналы, содержащие информацию о количестве ЯП, содержимое которых равно аргументу, больше аргумента и меньше аргумента. По команде rdAEqual анализируется содержимое ПФР, и на шину адреса откликнувшейся ячейки поступает адрес первой ЯП, содержимое которой равно аргументу. После чего происходит сброс ЯП в ПФР, соответствующей выданному адресу. По команде rdAMore анализируется содержимое ПФР, и на шину адреса откликнувшейся ячейки поступает адрес первой ЯП, содержимое которой больше аргумента. После чего происходит сброс ЯП в ПФР, соответствующей выданному адресу. По команде rdALess анализируется содержимое ПФР, и на шину адреса откликнувшейся ячейки поступает адрес первой ЯП, содержимое которой меньше аргумента. После чего происходит сброс ЯП в ПФР, соответствующей выданному адресу.

Блок конфигурационной памяти выполнен на регистре, отвечающем за выбор пространства адресов устройства PCI. Распределение адресов пространства ввода/вывода PCI-интерфейса приведено в табл. 2.

АМС можно реализовать на базе регистра сдвига (рис. 4,*а*) и приоритетного анализатора (рис. 4,*б*). Основными элементами АМС на базе регистра сдвига являются закольцованный регистр сдвига и счетчик адреса. Результат поиска всех ЯП записывается в этот закольцованный регистр сдвига (ПФР). Затем на регистр и на счетчик подается последовательность тактовых импульсов. Содержимое регистра сдвигается в сторону верхних разрядов до тех

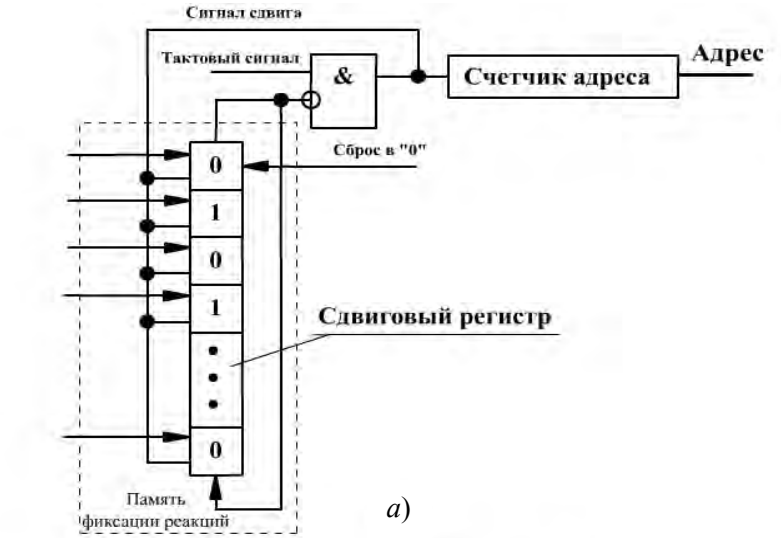
пор, пока в первом из них не окажется логическая «1». В этот момент тактовый сигнал автоматически блокируется. Если в исходном состоянии в счетчике были записаны одни нули, то по окончании счета его содержимое непосредственно указывает адрес первого совпавшего слова. Этот код заносится в адресный регистр, после чего осуществляется считывание слова. Далее единица в первом разряде регистра сбрасывается и автоматически возобновляется подача тактовых импульсов. Вновь содержимое регистра сдвигается вверх до появления в его первом разряде очередной единицы. После этого считывается следующее совпавшее слово и т.д. до тех пор, пока не будет обслужена вся очередь.

Таблица 2

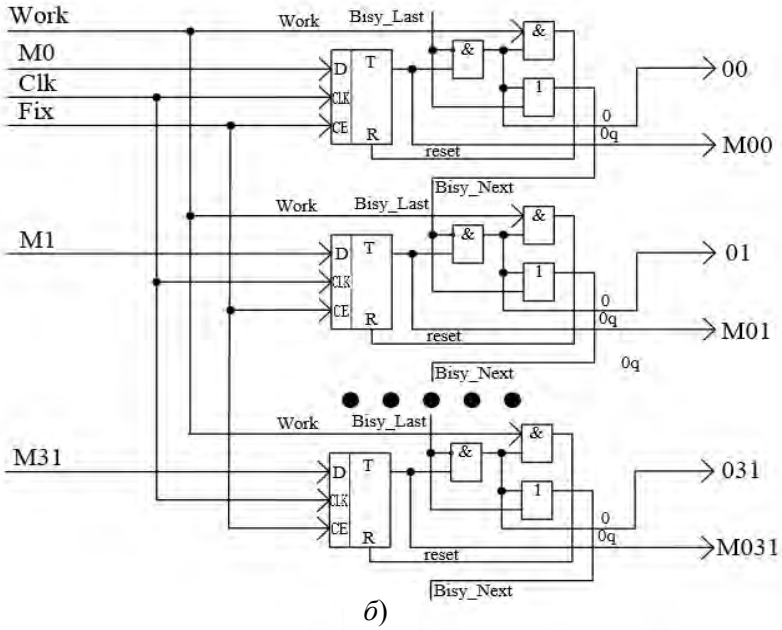
Распределение адресов ввода/вывода PCI-интерфейса

AD[31:10]	AD[9:7]	AD[4:0]	Описание	
			Чтение	Запись
BAR+	000	00000–11111	Чтение из ячеек памяти ассоциативного сопроцессора	Запись в ячейки памяти ассоциативного сопроцессора
	001	xxxxx	Чтение количества откликнувшихся ячеек	Запись аргумента поиска в регистр аргумента
	010	xxxxx	Чтение из ячейки памяти, содержимое которой равно аргументу	Запись в ячейку памяти, содержимое которой равно аргументу
	011	xxxxx	Чтение из ячейки памяти, содержимое которой больше аргумента	Запись в ячейку памяти, содержимое которой больше аргумента
	100	xxxxx	Чтение из ячейки памяти, содержимое которой меньше аргумента	Запись в ячейку памяти, содержимое которой меньше аргумента
	101	xxxxx	–	Фиксация откликнувшихся ячеек в памяти фиксации реакций

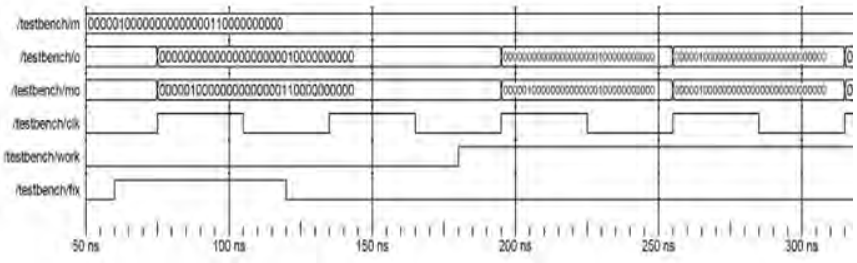
АМС на базе приоритетного анализатора состоит из D-триггеров, выполняющих функции ПФР и комбинационной логики. Эта схема работает по фронту сигнала CLK. Приоритетный анализатор представляет собой логическую цепь, позволяющую выделять среди своих входов, установленных в «1», линию с наименьшим номером. Он построен по принципу последовательного соединения разрядов. Каждый единичный вход этой цепи блокирует действие линий с большими номерами, в результате чего в единицу устанавливается только выход, соответствующий первой активной линии, сигнал с которого автоматически идет на выход, и функцией сигнала «Сброс» является сброс первого из «ответивших» триггеров.



а)



б)



в)

Рис. 4. Анализатор многократного совпадения на базе регистра сдвига (*а*) и на базе приоритетного анализатора (*б*); временные диаграммы работы памяти фиксации реакций и анализатора многократного совпадений (*в*)

ПФР и АМС реализуем на базе схемы приоритетного анализатора, т.к. эта схема обладает большим быстродействием, чем схема на базе регистра сдвига. Временные диаграммы работы схемы представлены на рис. 4,*в*.

Порядок работы с рассмотренным в работе блоком следующий:

1) необходимо заполнить ЯП АЗУ;
2) в регистр аргумента нужно записать аргумент поиска;
3) произвести фиксацию совпадений;
4) считать количество совпадений;
5) работа с откликнувшимися ячейками, т.е. чтение или запись в эти ячейки.

Рассматриваемый блок ассоциативного сопроцессора реализован на ПЛИС фирмы Xilinx. Разработан проект, состоящий из четырех основных модулей, реализующих главные части устройства: ассоциативный сопроцессор, ПФР и АМС, описание PCI интерфейса и сопряжение ассоциативного сопроцессора с ним.

Заключение

На основании описания ассоциативного сопроцессора на структурном уровне был разработан VHDL-код блока, после чего получен файл для конфигурирования ПЛИС. Работоспособность устройства и его отдельных блоков проверена тестированием и отладкой разработанных VHDL-кодов. Ассоциативный сопроцессор реализован аппаратно, что позволяет выполнять трудоемкие операции поиска и сравнения, тем самым разгружая центральный процессор и увеличивая производительность ВС в целом.

Список литературы

1. Бикташев, Р. А. Архитектура параллельных вычислительных систем / Р. А. Бикташев, В. С. Князьков. – Пенза : РИО Упринформпечати, 1993. – 166 с.
2. Кохонен, Т. Ассоциативные запоминающие устройства : пер. с англ. / Т. Кохонен. – М. : Мир, 1982. – 384 с.
3. Smith, J. E. The Miroarchitecture of Superscalar Processors / J. E. Smith, G. S. Sohi // Proc. IEEE. – 1995. – December. – Vol. 83, № 12. – P. 1609–1624.
4. Огнев, И. В. Ассоциативные среды / И. В. Огнев, В. В. Борисов. – М. : Радио и связь, 2000. – 312 с.
5. Bergamaschi, R. A. Designing Systems-on-Chip Using Cores / R. A. Bergamaschi, W. R. Lee // Proc. 37th Annual Design Automation Conference. – 2000. – P. 420–425.
6. Kozyrakis, C. E. Vector vs. Superscalar and VLIW Architectures for Embedded Multimedia Benchmarks / C. E. Kozyrakis, D. A. Patterson // Proc. 35th International Symposium on Microarchitecture. – Istanbul, Turkey, 2002. – November.
7. Kozyrakis, C. E. Scalable Vector Processors for Embedded Systems / C. E. Kozyrakis, D. A. Patterson // IEEE Micro. – 2003. – Vol. 23, № 6. – P. 36–45.
8. Constantinides, G. A. Synthesis and Optimization of DSP Algorithms / G. A. Constantinides, P. Y. K. Cheung, W. Luk. – Boston : Kluwer Academic Publishers, 2004. – 162 p.
9. Kuon, I. Measuring the gap between FPGAs and ASICs / I. Kuon, J. Rose // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 2007. – Vol. 62, № 2. – P. 203–215.

10. Mattson, T. G. Programming the Intel 80-core network-on-a-chip Terascale Processor / T. G. Mattson, R. van der Wijngaart, M. Frumkin // Proc. 2008 ACM/IEEE Conference on Supercomputing (SC'08). – 2008.
11. Зубковский, П. С. Сопроцессор комплексных вычислений / П. С. Зубковский, Е. В. Ивасюк, С. И. Аряшев // Проблемы разработки перспективных микро- и наноэлектронных систем – 2010 : сб. тр. / под общ. ред. академика РАН А. Л. Стемпковского. – М. : ИППМ РАН, 2010. – С. 356–359.
12. Пантелеев, А. Ю. Векторный сопроцессор для реализации DSP-алгоритмов в числах с плавающей точкой / А. Ю. Пантелеев, И. И. Шагурин // Научная сессия МИФИ 2012 : сб. науч. тр. – М. : МИФИ, 2012. – Т. 1. – С. 80–81.
13. Мартышкин, А. И. Функциональная организация модуля ассоциативного сопроцессора для специализированных вычислительных систем / А. И. Мартышкин // Модели, системы, сети в экономике, технике, природе и обществе. – 2016. – № 3 (19). – С. 157–166.
14. Мартышкин, А. И. Комплекс программ для определения характеристик диспетчеров задач многопроцессорных систем с использованием приоритетных стохастических сетей массового обслуживания / А. И. Мартышкин, Р. А. Бикташев, Н. Г. Востоков // Фундаментальные исследования. – 2013. – № 10-1. – С. 13–20.
15. Мартышкин, А. И. Численный метод для определения пропускной способности приоритетного потока заявок в многопроцессорной системе с общим диспетчером задач по каждому конкретному типу приоритета / А. И. Мартышкин, Р. А. Бикташев, А. А. Воронцов // XXI век: итоги прошлого и проблемы настоящего плюс. – 2014. – № 3 (19). – С. 137–145.
16. Мартышкин, А. И. Математическое моделирование диспетчеров задач с пространственным разделением с неоднородным потоком задач на обслуживание и ограниченной длиной очереди / А. И. Мартышкин, А. А. Воронцов, О. О. Валова // XXI век: итоги прошлого и проблемы настоящего плюс. – 2015. – № 3 (25). – С. 142–149.
17. Цилькер, Б. Я. Организация ЭВМ и систем / Б. Я. Цилькер, С. А. Орлов. – 2-е изд. – СПб. : Питер, 2011. – 688 с.

Мартышкин Алексей Иванович
кандидат технических наук, доцент,
кафедра вычислительных машин
и систем,
Пензенский государственный
технологический университет
E-mail: alexey314@yandex.ru

Martyshkin Alexey Ivanovich
candidate of technical sciences,
associate professor,
sub-department of computers and systems,
Penza State Technological University

УДК 004.31
Мартышкин, А. И.
Структурная организация блока ассоциативного сопроцессора на базе программируемой логической интегральной схемы для специализированных вычислительных систем / А. И. Мартышкин // Модели, системы, сети в экономике, технике, природе и обществе. – 2016. – № 4 (20). – С. 128–138.