

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
Пермский национальный исследовательский политехнический университет
(ПНИПУ)

Факультет: Электротехнический (ЭТФ)

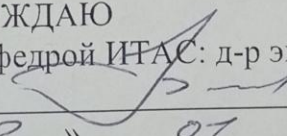
Направление: 09.03.04 – Программная инженерия (ПИ)

Профиль: Разработка программно-информационных систем (РИС)

Кафедра информационных технологий и автоматизированных систем (ИТАС)

УТВЕРЖДАЮ

Зав. кафедрой ИТАС: д-р экон. наук, проф.

 Р.А. Файзрахманов
« 13 » 01 2023 г.

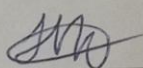
КУРСОВАЯ РАБОТА

по дисциплине

«Организация ЭВМ и систем»

на тему

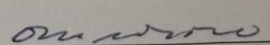
«Структурно-алгоритмическое проектирование ЭВМ»

Студент:  13.01.23 Нечаев Игорь Александрович
(подпись, дата)

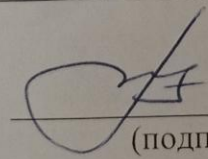
Группа: РИС-21-1бзу

Дата сдачи 13.01.23

Дата защиты 13.01.23

Оценка 

Руководитель КР:

 13.01.23
(подпись, дата)

к.т.н., доц. каф. ИТАС Погудин А.Л.

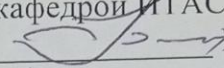
Пермь 2023

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
Пермский национальный исследовательский политехнический университет
(ПНИПУ)

Факультет: Электротехнический (ЭТФ)
Направление: 09.03.04 – Программная инженерия (ПИ)
Профиль: Разработка программно-информационных систем (РИС)
Кафедра информационных технологий и автоматизированных систем (ИТАС)

УТВЕРЖДАЮ

Зав. кафедрой ИТАС: д-р экон. наук, проф.

 Р.А. Файзрахманов
« 16 » 05 2023 г.

ЗАДАНИЕ
на выполнение курсовой работы

Фамилия, имя, отчество: Нечаев Игорь Александрович
Факультет Электротехнический Группа РИС-21-1бзу
Начало выполнения работы: 16.05.2022
Контрольные сроки просмотра работы: 27.05, 04.06, 09.01, 11.01
Защита работы: 13.01.2023

1. Наименование темы: «Структурно-алгоритмическое проектирование ЭВМ».

2. Исходные данные к работе (проекта):

Объект исследования – Арифметико-логическое устройство (АЛУ).

Предмет исследования – Алгоритм работы и структура АЛУ.

Цель работы (проекта) – разработать контролер КЭШ-памяти с прямым способом отображения основной памяти (ОЗУ) на КЭШ и со сквозной записью.

3. Содержание:

3.1 Исследование предметной области курсовой работы

3.2 Кэш память

3.3 Контроллер кэш памяти

3.4 Стратегия записи в КЭШ

3.5 Стратегия размещения блоков в ОЗУ КЭШ

3.6 Разработка устройства

3.7 Анализ исходных данных задания на курсовую работу

3.8 Спецификация устройства на уровне «черного ящика»

3.9 Представление «черного ящика» в виде операционной и управляющей частей

3.10 Разработка структуры операционной части устройства

3.11 Разработка схемы алгоритма работы

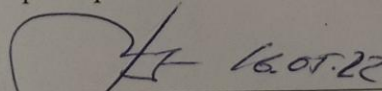
3.12 Составление полной спецификации устройства

3.13 Разработка схемы алгоритма работы

3.14 Разработка временной диаграммы работы устройства

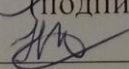
3.15 Контрольный пример

Руководитель КР:


(подпись, дата)

к.т.н., доц. каф. ИТАС Погудин А.Л.

Задание получил:

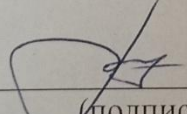

(подпись, дата)

И.А. Нечаев

КАЛЕНДАРНЫЙ ГРАФИК ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

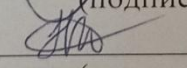
№ пп	Этапы работы	Объём этапа, %	Сроки выполнения	
			Начало	Конец
1.	Исследование предметной области	10	16.05.22	20.05.22
2.	Устройство управления	5	23.05.22	27.05.22
3.	Сложение	5	30.05.22	03.06.22
4.	Вычитание	5	06.06.22	10.06.22
5.	Сравнение	5	13.06.22	17.06.22
6.	Логическое «И»	5	20.06.22	08.07.22
7.	Адресация	5	11.07.22	15.07.22
8.	Разработка устройства.	5	18.07.22	22.07.22
9.	Анализ исходных данных задания на курсовую работу	5	25.07.22	05.08.22
10.	Спецификация устройства на уровне «черного ящика»	5	08.08.22	12.08.22
11.	Представление черного ящика в виде операционной и управляющей частей	5	15.08.22	19.08.22
12.	Разработка структуры операционной части устройства	5	22.08.22	09.09.22
13.	Составление схемы алгоритма работы устройства и его микропрограммы	5	12.09.22	16.09.22
14.	Разработка схемы алгоритма работы	5	19.09.22	23.09.22
15.	Составление полной спецификации устройства	5	26.09.22	14.10.22
16.	Разработка фрагмента функциональной схемы	5	17.10.22	11.11.22
17.	Контрольный пример	5	14.11.22	09.12.22
18.	Временная диаграмма работы УУ	5	12.12.22	16.12.22
19.	Оформление курсовой работы	5	23.12.22	12.01.23
20.	Защита курсовой работы		13.01.23	

Руководитель КР:


(подпись, дата)

к.т.н., доц. каф. ИТАС Погудин А.Л..

Задание получил:


(подпись, дата)

Нечаев Игорь Александрович

РЕФЕРАТ

Отчет 22 с., 8 рис., 1 табл., 4 источника, 1 приложение.

Кэш-память; контроллер КЭШ-памяти; ОЗУ; КЭШ-промах; КЭШ-попадание; прямое отображение КЭШ на ОЗУ.

Цель работы – разработать контролер КЭШ-памяти с прямым способом отображения основной памяти (ОЗУ) на КЭШ и со сквозной записью.

При разработке использовалась концепция «черного ящика», т.е. первоначальное определение общих функций устройства и системы входных и выходных сигналов. В основе дальнейшей работы с «черным ящиком» использовался принцип декомпозиции, т.е. последовательное разложение функций на подфункции до получения описания функций на элементарном уровне.

В результате работы была составлен алгоритм работы и структура устройства.

Приведен контрольный пример в числовой форме.

**ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ УСЛОВНЫХ ОБОЗНАЧЕНИЙ,
СОКРАЩЕНИЙ И ТЕРМИНОВ**

ЧЯ	Черный ящик
УЧ	Управляющая часть устройства
ОЧ	Операционная часть устройства
АЛУ	Арифметико-логическое устройство
МО	Микрооперация
МПР	Микропрограмма
Рг	Регистр

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СОКРАЩЕНИЙ И ТЕРМИНОВ	5
ВВЕДЕНИЕ	7
1 Исследование предметной области курсовой работы.....	8
1.1 Кэш память.....	8
1.2 Контроллер кэш памяти.....	9
1.3 Стратегия записи в КЭШ.....	10
1.4 Стратегия размещения блоков в ОЗУ КЭШ.....	11
2 Разработка устройства	13
2.1 Анализ исходных данных задания на курсовую работу	13
2.2 Спецификация устройства на уровне «черного ящика»	13
2.3 Представление «черного ящика» в виде операционной и управляющей частей.....	14
2.4 Разработка структуры операционной части устройства	14
2.5 Разработка схемы алгоритма работы	15
2.6 Составление полной спецификации устройства.....	16
2.7 Разработка временной диаграммы работы устройства.....	16
2.8 Контрольный пример.....	18
ЗАКЛЮЧЕНИЕ	19
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	20
ПРИЛОЖЕНИЕ	21

ВВЕДЕНИЕ

Цель работы (проекта) – разработать контролер КЭШ-памяти с прямым способом отображения основной памяти (ОЗУ) на КЭШ и со сквозной записью.

Объектом исследования является устройство управления. Предметом исследования – алгоритм работы и структура устройства.

В работе представлена спецификация устройства на уровне «черного ящика», разработана схема алгоритма работы устройства и его микропрограммы, составлена полная спецификация устройства. Построена временная диаграмма работы устройства управления. Приведен листинг разработанной программы на языке программирования C# и результаты расчета контрольного примера.

1 Исследование предметной области курсовой работы

1.1 Кэш память

Система кэш-памяти процессора состоит из двух блоков - контроллера кэш-памяти и собственно самой кэш-памяти.

Кэш-память процессора изготавливают в виде микросхем статической памяти (Static Random Access Memory, сокращенно - SRAM). По сравнению с другими типами памяти, статическая память обладает очень высокой скоростью работы. Однако, эта скорость зависит также от объема конкретной микросхемы. Чем значительней объем микросхемы, тем сложнее обеспечить высокую скорость ее работы. Если такая особенность учтена производителем, то кэш-память процессора содержит несколько блоков, называемых уровнями. В большинстве процессоров используется трехуровневая система кэша:

- Кэш-память первого уровня – очень маленькая, но самая быстрая микросхема памяти. Ее объем не превышает нескольких десятков килобайт. Работает она без каких-либо задержек. В ней содержатся данные, которые чаще всего используются процессором. Количество микросхем памяти, как правило, равно количеству его ядер. Каждое ядро имеет доступ только к своей микросхеме;

- Кэш-память второго уровня немного медленнее кэш-памяти первого, но и объем ее более существенный (около несколько сотен килобайт). Служит она для временного хранения важной информации, вероятность запроса которой ниже, чем у информации, находящейся в первом уровне;

- Кэш-память третьего уровня – еще более объемная, но и более медленная схема памяти. Тем не менее, она быстрее оперативной памяти. Ее размер может достигать нескольких десятков мегабайт. В отличие от 1 и 2 уровней, она является общей для всех ядер процессора. Служит для временного хранения важных данных с относительно низкой вероятностью запроса, а также для обеспечения взаимодействия ядер процессора между собой.

1.2 Контроллер кэш памяти

Это устройство, управляющее содержанием кэша, получением необходимой информации из оперативной памяти, передачей ее процессору, а также возвращением в оперативную память результатов вычислений. Когда ядро процессора обращается к контроллеру за какими-то данными, тот проверяет, есть ли эти данные в кэш-памяти. Если это так, ядру моментально отдается информация из кэша. В противном случае ядру приходится ожидать поступления данных из медленной оперативной памяти. Ситуация, когда в кэше не оказывается нужных данных, называется кэш-промахом. Задача контроллера – сделать так, чтобы кэш-промахи происходили как можно реже, а в идеале – чтобы их не было вообще.

Размер кэша процессора по сравнению с размером оперативной памяти несоизмеримо мал. В нем может находиться лишь копия крошечной части данных, хранимых в оперативной памяти. Но, несмотря на это, контроллер допускает кэш-промахи не часто. Эффективность его работы определяется несколькими факторами:

- Размером и структурой кэш-памяти. Чем больше ресурсов имеет в своем распоряжении контроллер, тем ниже вероятность кэш-промаха;
- Эффективностью алгоритмов, по которым контроллер определяет, какая именно информация понадобится процессору в следующий момент времени;
- Сложностью и количеством задач, одновременно решаемых процессором. Чем сложнее задачи и чем их больше, тем чаще "ошибается" контроллер.

Процессор редко использует весь объем ОЗУ практически одновременно. Зачастую все обращения процессора к памяти сосредоточены в небольшой области (как показывает статистика - 5-10% от общего объема). Если данные из этой области как-либо аппаратно скопировать в КЭШ, а затем постоянно сверять КЭШ и ОЗУ на предмет целостности данных, то можно обеспечить режим работы, при котором процессор будет обращаться

только к КЭШ-памяти, тратя на это значительно меньше ресурсов и времени, чем обычно.

Когда процессор обращается к определенной ячейке памяти, сегмент памяти определенного объема (этот объем называется объемом страницы КЭШ) копируется в КЭШ полностью. Если процессор дальше не совершит глобальный скачек на другой, далекий от текущего, адрес памяти, то дальнейшая работа процессора будет происходить напрямую с КЭШ, минуя ОЗУ, а контроллер Кэш-памяти в промежутках, когда процессор занят вычислениями (либо параллельно с работой процессора), будет восстанавливать верные данные в ОЗУ, либо в КЭШ (в случае наличия устройств, напрямую работающих с памятью). Естественно, чем больше будет страниц и чем больше будет их объем - тем выше будет скорость работы процессора.

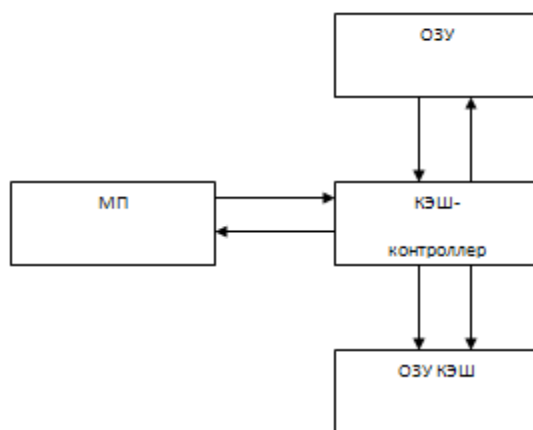


Рисунок 1 – Схема связи КЭШ-контроллера

1.3 Стратегия записи в КЭШ

Когда выполняется запись в Кэш-память, имеются две базовые возможности:

- сквозная запись (write through, store through) - информация записывается в два места: в блок КЭШ-памяти и в блок, и в ОЗУ;
- запись с обратным копированием (write back, copy back, store in) - информация записывается только в блок КЭШ-памяти.

Оба подхода к организации записи имеют свои преимущества и недостатки. При записи с обратным копированием операции записи выполняются со скоростью КЭШ-памяти, и несколько записей в один и тот же блок требуют только одной записи в память более низкого уровня. Поскольку в этом случае обращения к основной памяти происходят реже, требуется меньшая полоса пропускания памяти, что очень привлекательно для мультипроцессорных систем. Сквозная запись имеет также преимущество в том, что основная память имеет наиболее свежую копию данных. Это важно в мультипроцессорных системах, а также для организации ввода/вывода.

1.4 Стратегия размещения блоков в ОЗУ КЭШ

Принципы размещения блоков в КЭШ-памяти определяют три основных типа их организации:

- Если каждый блок основной памяти имеет только одно фиксированное место, на котором он может появиться в КЭШ-памяти, то такая КЭШ-память называется КЭШ с прямым отображением (direct mapped). Это наиболее простая организация КЭШ-памяти, при которой для отображения адресов блоков основной памяти на адреса КЭШ-памяти просто используются младшие разряды адреса блока. Таким образом, все блоки основной памяти, имеющие одинаковые младшие разряды в своем адресе, попадают в один блок КЭШ-памяти.

- Если некоторый блок основной памяти может располагаться на любом месте КЭШ-памяти, то КЭШ называется полностью ассоциативным (fully associative).

- Если некоторый блок основной памяти может располагаться на ограниченном множестве мест в КЭШ-памяти, то КЭШ называется множественно-ассоциативным (set associative). Обычно множество представляет собой группу из двух или большего числа блоков в КЭШ. Если

множество состоит из n блоков, то такое размещение называется множественно-ассоциативным с n каналами (n -way set associative).

2 Разработка устройства

2.1 Анализ исходных данных задания на курсовую работу

В современных ЭВМ используются несколько уровней КЭШ. Начиная с процессора i486, КЭШ первого уровня находится непосредственно в корпусе процессора, КЭШ второго уровня установлен виде элемента на материнской плате, работающего быстрее, чем ОЗУ, но медленнее, чем КЭШ первого уровня. Так как в задании нет указания на это, КЭШ не будет подразделен на уровни и будет рассмотрен как обобщенное устройство.

Так как в задании не указана разрядность шины адреса, она будет принята равной 32 бита. Запоминающий массив представляет собой 8 (N) строки.

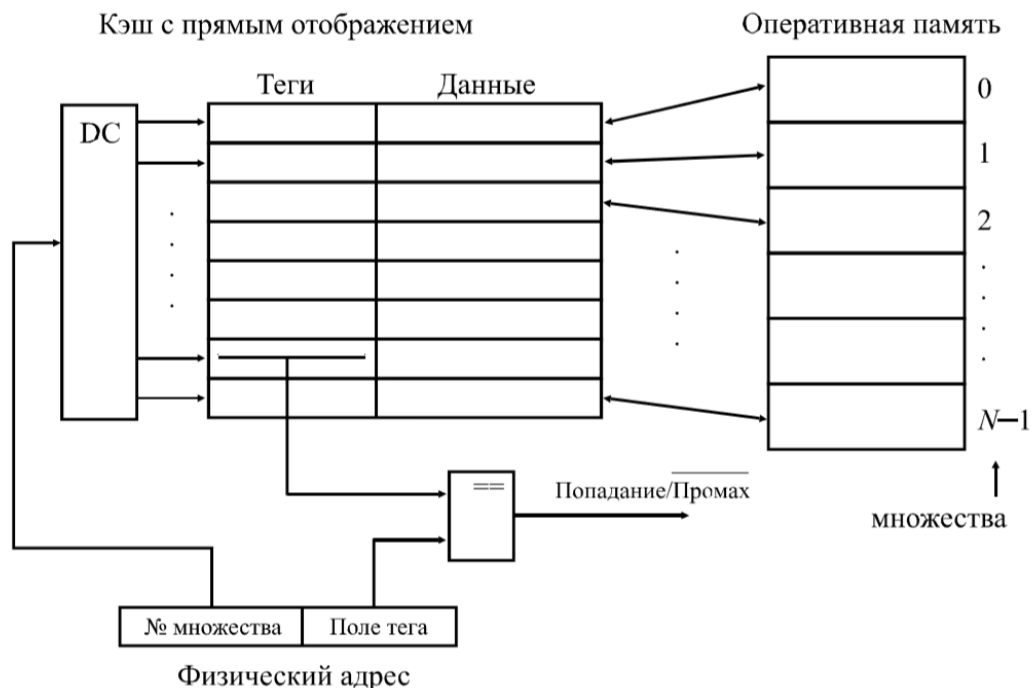


Рисунок 2 – Схема таблицы данных кэша с прямым отображением

2.2 Спецификация устройства на уровне «черного ящика»

Контроллер кэш-памяти — это устройство, расположенное между CPU и оперативной памятью и предназначенное для минимизации количества обращений к оперативной памяти.

В связи с этим к контроллеру со стороны ЦП должны подходить шины адреса, данных и управления. В то же время для обработки кэш-промахов и

загрузки новых данных контроллер должен иметь доступ к оперативной памяти ЭВМ и поэтому от контроллера кэш-памяти к ОЗУ тоже должны проходить три шины - адреса, данных и управления (рис. 2).

Разрядности шин адреса должны быть одинаковыми, и были приняты равными 32 бита. Разрядности шин данных - 8 бит. В связи с тем, что для памяти нужно только два управляющих сигнала - читать (код запроса - 1) и писать (код запроса - 0), то для их кодирования достаточно кода управления разрядностью 1 бит.



Рисунок 3 – Подключение кэш-контроллер к шинам

2.3 Представление «черного ящика» в виде операционной и управляющей частей

Упрощенно устройство разрабатываемого умножителя можно представить схемой из рисунка 2.

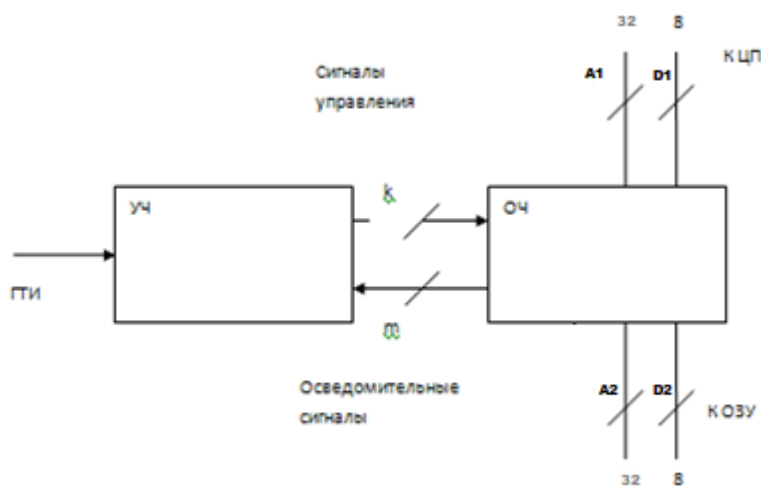


Рисунок 4 – Структурная схема УЧ и ОЧ

2.4 Разработка структуры операционной части устройства

Основным элементом ОЧ является ОЗУ КЭШ, предназначенное для хранения данных, адресов и частоты обращения. Логика поиска строки КЭШ

по адресу и поиск строки – кандидата на перезапись реализована с помощью схемы сравнения, АЛУ и дешифратора индекса (индекс -> адрес блока в ОЗУ КЭШ).

При разработке алгоритма работы контроллера кэш-памяти в качестве алгоритма поиска был использован алгоритм линейного поиска, который за один проход находит и строку с заданным адресом, и наименее часто используемую строку.

Для хранения номеров строк КЭШ, индекса, адресов потребуется несколько регистров, а именно: регистр адреса, регистр данных, регистр бита чтения.

Запрос к контроллеру, состоящий из кода запроса, адреса и данных, передается во входные регистры.

2.5 Разработка схемы алгоритма работы

Схема алгоритма работы КЭШ-контроллера с прямым способом отображения основной памяти (ОЗУ) на КЭШ и со сквозной записью изображена на рисунке 5.

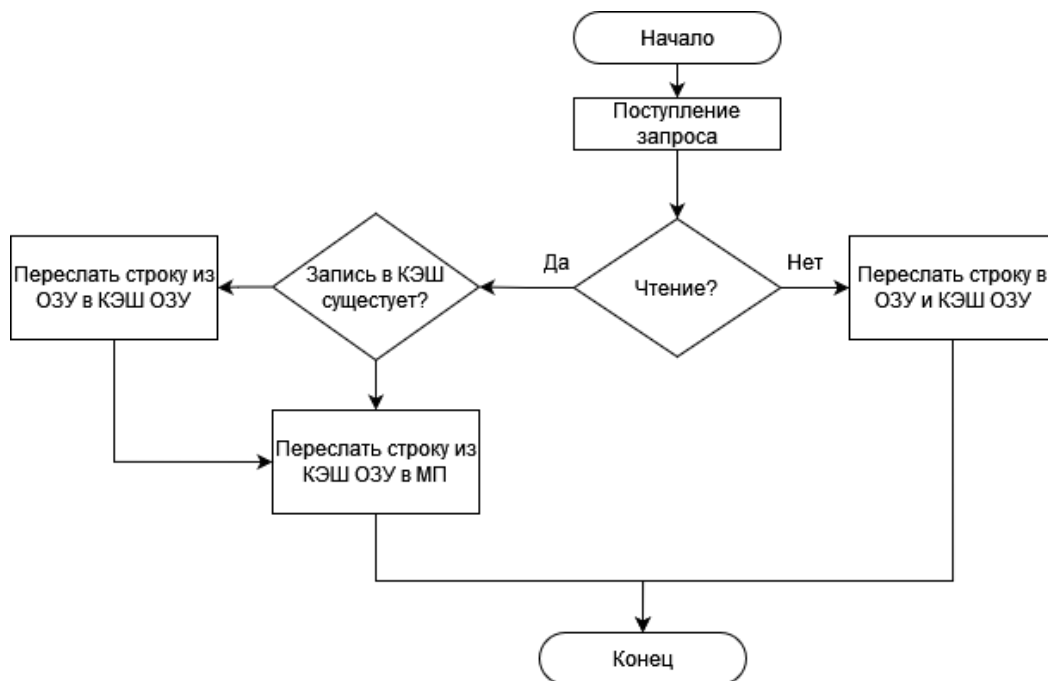


Рисунок 5 – Блок-схема алгоритма работы КЭШ-контроллера

2.6 Составление полной спецификации устройства

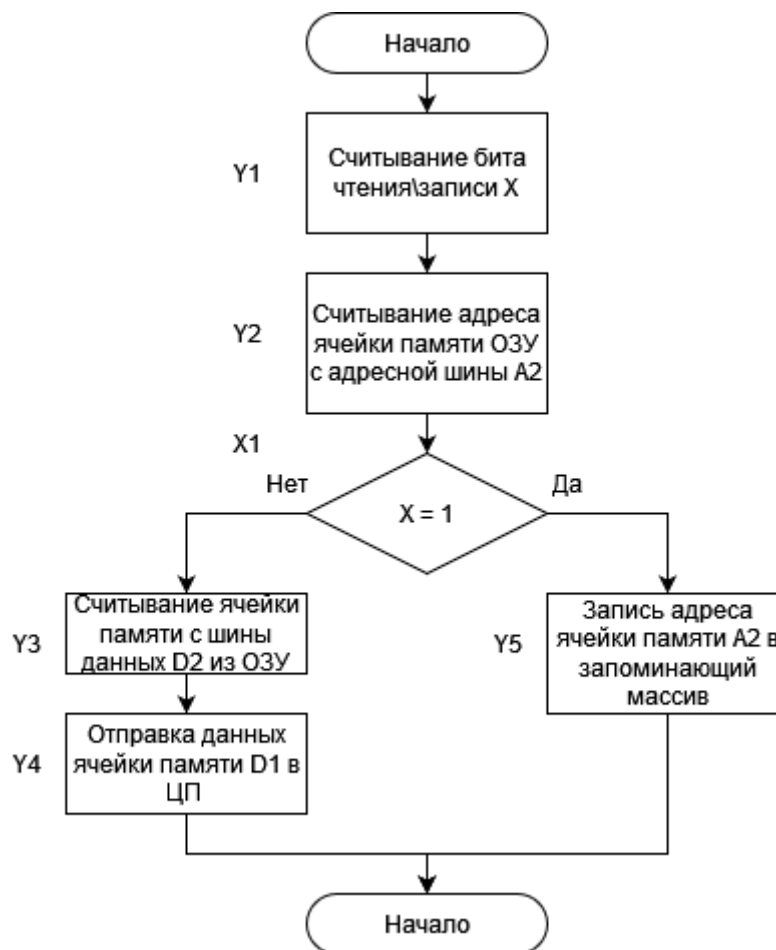


Рисунок 6 – Блок-схема на уровне микроопераций устройства

В таблице 1 приведено описание всех линий и сигналов, полученных в процессе работы устройства.

Таблица 1 – Расшифровка микроопераций и сигналов

Имя сигнала/шины и разрядность	Тип (In/Out)	Назначение сигнала
Y1	I для ОЧ	Сигнал о чтении бита с шины X
Y2	I для ОЧ	Сигнал о чтении ячейки памяти ОЗУ в шину A2
Y3	I для ОЧ	Сигнал о чтении ячейки памяти с шины D2
Y4	I для ОЧ	Сигнал о записи данных на шину D1 в ЦП
Y5	I для ОЧ	Сигнал о записи адреса ячейки памяти в шину A2
X1	I для ОЧ	Сигнал об установке бита чтения/записи (0\1)

2.7 Разработка временной диаграммы работы устройства

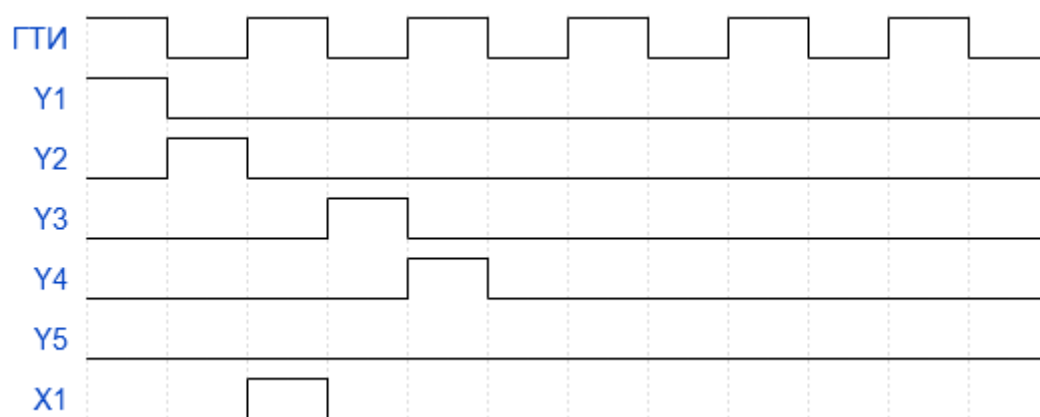


Рисунок 6 – Временная диаграмма при чтении (бит чтения = 1)

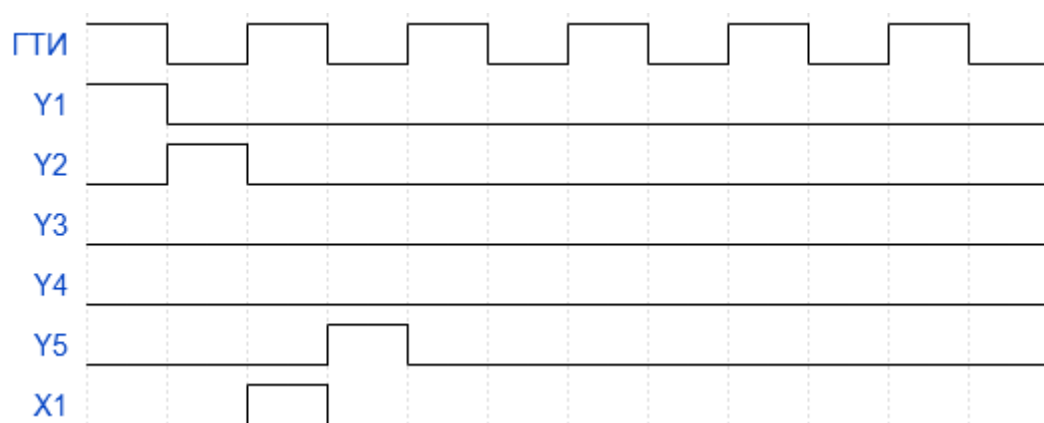


Рисунок 7 – Временная диаграмма при записи (бит чтения = 0)

По горизонтали идут интервалы времени, которые в возбужденном состоянии соответствуют логическим 1. Вертикальные штриховые линии разделяют временные такты. По вертикали расположены импульсы начиная от ГТИ – генератора тактовых импульсов и далее сигналов устройства.

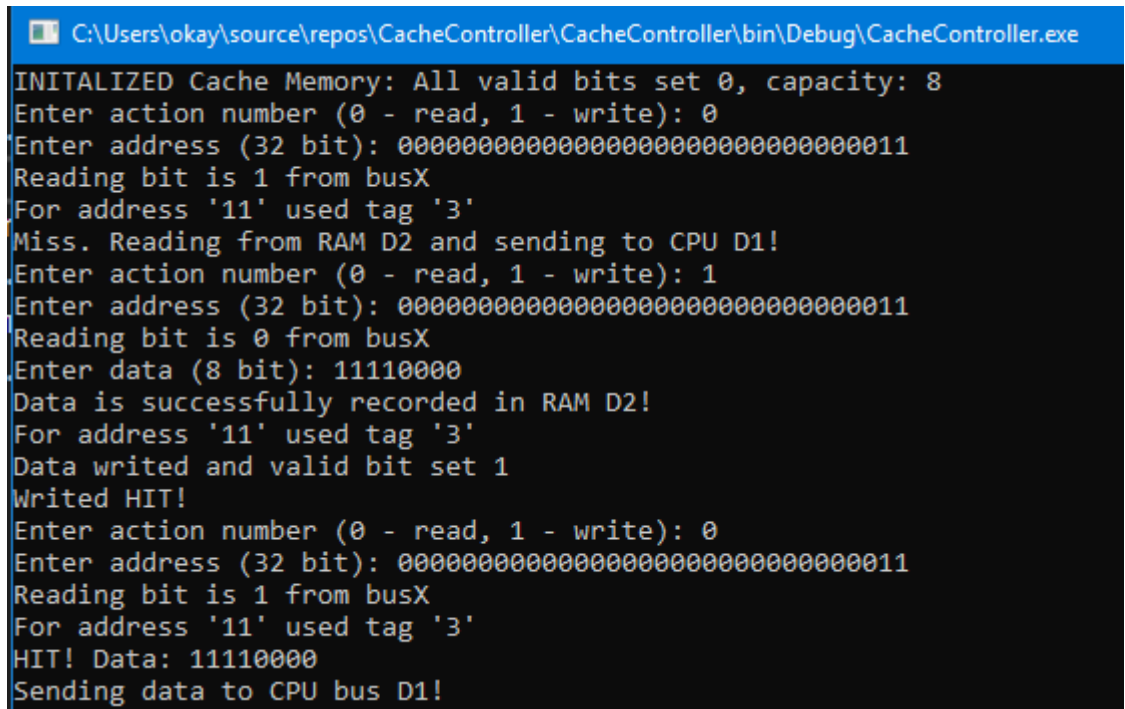
2.8 Контрольный пример

Исходные данные:

- Адрес ячейки памяти ОЗУ:
00000000000000000000000000000011

- Значение ячейки памяти: 11110000

Результат работы программы:



```
C:\Users\okay\source\repos\CacheController\CacheController\bin\Debug\CacheController.exe
INITIALIZED Cache Memory: All valid bits set 0, capacity: 8
Enter action number (0 - read, 1 - write): 0
Enter address (32 bit): 00000000000000000000000000000011
Reading bit is 1 from busX
For address '11' used tag '3'
Miss. Reading from RAM D2 and sending to CPU D1!
Enter action number (0 - read, 1 - write): 1
Enter address (32 bit): 00000000000000000000000000000011
Reading bit is 0 from busX
Enter data (8 bit): 11110000
Data is successfully recorded in RAM D2!
For address '11' used tag '3'
Data written and valid bit set 1
Writed HIT!
Enter action number (0 - read, 1 - write): 0
Enter address (32 bit): 00000000000000000000000000000011
Reading bit is 1 from busX
For address '11' used tag '3'
HIT! Data: 11110000
Sending data to CPU bus D1!
```

Рисунок 8 – Результат работы программы

ЗАКЛЮЧЕНИЕ

Задача курсовой работы – разработать контролер КЭШ-памяти с прямым способом отображения основной памяти (ОЗУ) на КЭШ и со сквозной записью.

Поставленная задача выполнена. В ходе курсовой работы была изучена специальная литература, разработана структура ОЧ, алгоритм их работы, спецификация сигналов, фрагмент функциональной схемы УЧ, контрольный числовой пример.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Жмакин, А. П. Архитектура ЭВМ: 2-е изд., перераб. и доп.: учеб. пособие. — СПб.: БХВ-Петербург, 2010. — 352 с.
2. Потапов, В.И., Шафеева, О.П., Червенчук, И.В. Основы компьютерной арифметики и логики: Учеб. пособие. — Омск: Изд-во ОмГТУ, 2004. — 172 с.
3. Потапов, И. В. Элементы прикладной теории цифровых автоматов: учеб. пособие / И. В. Потапов. — Омск: Изд-во ОмГТУ, 2011. — 156 с.
4. Тюкачев, Н. А. С#. Основы программирования : учебное пособие для вузов / Н. А. Тюкачев, В. Г. Хлебостроев. — 4-е изд., стер. — Санкт-Петербург : Лань, 2021. — 272 с. — ISBN 978-5-8114-7266-6.

ПРИЛОЖЕНИЕ

Код программы на языке C#

```
using System;
using System.Collections.Generic;

namespace CacheController
{
    class CacheMemory
    {
        public List<bool> valids;
        public List<Int32> addresses;
        public List<byte> data;
        public int capacity;

        public CacheMemory(int capacity)
        {
            Console.WriteLine("INITIALIZED Cache Memory: All valid bits
set 0, capacity: {0}", capacity);
            this.capacity = capacity;
            valids = new List<bool>(new bool[capacity]);
            addresses = new List<Int32>(new Int32[capacity]);
            data = new List<byte>(new byte[capacity]);
        }

        public void Write(Int32 address, byte data)
        {
            byte tag = (byte)(address % capacity);
            Console.WriteLine("For address '{0}' used tag '{1}'",
Convert.ToString(address, 2), tag);
            this.addresses[tag] = address;
            this.data[tag] = data;
            this.valids[tag] = true;
            Console.WriteLine("Data writed and valid bit set 1");
        }

        public byte? Read(Int32 address)
        {
            byte tag = (byte)(address % capacity);
            Console.WriteLine("For address '{0}' used tag '{1}'",
Convert.ToString(address, 2), tag);
            if (!valids[tag])
                return null;
            return this.data[tag];
        }
    }
}
```

```

class Program
{
    static void Main(string[] args)
    {
        var RAM = new Dictionary<Int32, byte>();
        bool isRead;
        Int32 addressBus;
        byte dataBus;

        var cacheMemory = new CacheMemory(8);
        while (true)
        {
            Console.Write("Enter action number (0 - read, 1 -
write): ");
            isRead = Console.ReadLine() == "0";
            Console.Write("Enter address (32 bit): ");
            addressBus = Convert.ToInt32(Console.ReadLine(), 2);
            Console.WriteLine("Reading bit is {0} from busX",
isRead ? 1 : 0);
            if (isRead)
            {
                var data = cacheMemory.Read(addressBus);
                if (data != null)
                {
                    Console.WriteLine("HIT! Data: " +
Convert.ToString((byte)data, 2));
                    Console.WriteLine("Sending data to CPU bus
D1!");
                } else {
                    Console.WriteLine("Miss. Reading from RAM D2
and sending to CPU D1!");
                }
            } else {
                Console.Write("Enter data (8 bit): ");
                dataBus = Convert.ToByte(Console.ReadLine(), 2);
                RAM[addressBus] = dataBus;
                Console.WriteLine("Data is successfully recorded
in RAM D2!");

                cacheMemory.Write(addressBus, dataBus);
                Console.WriteLine("Writed HIT!");
            }
        }
    }
}

```