

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Пермский национальный исследовательский политехнический
университет»

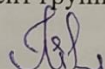
Кафедра ИТАС

КУРСОВАЯ РАБОТА

по дисциплине «ОРГАНИЗАЦИЯ ЭВМ И СИСТЕМ»

Тема: «Структурно-алгоритмическое проектирование ЭВМ»

Выполнил студент группы РИС-19-16

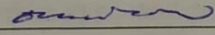
 Миннахметов Э.Ю.

Руководитель

 Погудин А.Л.

Дата сдачи 18.12.2021

Дата защиты 22.12

Оценка 

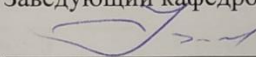
Пермь 2021

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Пермский национальный исследовательский политехнический университет»

Кафедра ИТАС

«УТВЕРЖДАЮ»

Заведующий кафедрой ИТАС

 Р.А. Файзрахманов

« 13 » сентября 2021 год

ЗАДАНИЕ
на выполнение курсовой работы

Фамилия И.О. Миннахметов Эльдар Юлдашевич

Факультет Электротехнический Группа РИС-19-16

Начало выполнения работы: 13 сентября 2021 года

Контрольные сроки просмотра работы: 22.09.2021, 20.10.2021, 10.11.2021

Защита работы: 22.12.21

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

2. Исходные данные к работе (проекта):

Объект исследования – Контроллер ассоциативной памяти

Предмет исследования – Алгоритм работы и структура контроллера ассоциативной памяти

Цель работы (проекта) – Разработать контроллер ассоциативной памяти, хранящей 64*8-разрядных двоичных кодов с выходом по «равно признаку» и «не равно признаку». Признаком поиска может быть 1,2,4,8 двоичных разрядов. Результатом поиска, выводимым на ШД, является весь байт.

3. Содержание:

3.1 Исследование предметной области курсовой работы

3.2 Анализ исходных данных задания на курсовую работу

3.3 Спецификация устройства на уровне «черного ящика»

3.4 Представление устройства в виде операционной и управляющей частей

3.5 Разработка структуры устройства

3.6 Составление алгоритма работы устройства.

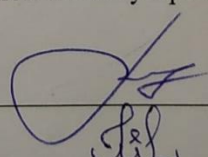
3.7 Разработка микропрограммы работы устройства

3.8 Составление полной спецификации устройства

3.9 Составление фрагмента функциональной схемы устройства

3.10 Контрольный пример

Руководитель курсовой работы _____



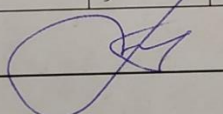
А.Л. Погудин

Задание получил _____

Э.Ю. Миннахметов

КАЛЕНДАРНЫЙ ГРАФИК ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

№ пп	Этапы работы	Объём этапа, %	Сроки выполнения		Приме- чание
			Начало	Конец	
1.	Исследование предметной области курсо- вой работы	9	13.09.2021	17.09.2021	<i>Введен</i>
2.	Анализ исходных данных задания на кур- совую работу	9	20.09.2021	01.10.2021	<i>Введен</i>
3.	Спецификация устройства на уровне «чер- ного ящика»	9	04.10.2021	08.10.2021	<i>Введен</i>
4.	Представление черного ящика в виде опе- рационной и управляющей частей	9	11.10.2021	15.10.2021	<i>Введен</i>
5.	Разработка структуры устройства	9	18.10.2021	22.10.2021	<i>Введен</i>
6.	Составление алгоритма работы устрой- ства	9	25.10.2021	29.10.2021	<i>Введен</i>
7.	Составление полной спецификации устройства	9	1.11.2021	12.11.2021	<i>Введен</i>
8.	Составление фрагмента функциональной схемы устройства	9	15.11.2021	19.11.2021	<i>Введен</i>
9.	Контрольный пример	9	22.11.2021	03.12.2021	<i>Введен</i>
10.	Оформление курсовой работы	10	06.12.2021	17.12.2021	<i>Введен</i>
11.	Защита курсовой работы	9	20.12.2021	22.12.2021	<i>Введен</i>

Руководитель курсовой работы  А.Л. Погудин

« 15 » сентября 2021 года

РЕФЕРАТ

Отчет 27 с., 21 рис., 1 табл., 3 источника.

АРИФМЕТИКО-ЛОГИЧЕСКОЕ УСТРОЙСТВО, УПРАВЛЯЮЩЕЕ УСТРОЙСТВО, ПОИСК, ЗАНУЛЕНИЕ, СОХРАНЕНИЕ ЗНАЧЕНИЯ.

Цель работы – разработка алгоритма работы и структуры работы устройства для выполнения трех команд.

При разработке устройства использовались концепции «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.

СОДЕРЖАНИЕ

Перечень используемых условных обозначений, сокращений и терминов.....	6
ВВЕДЕНИЕ.....	7
1 Исследование предметной области	8
1.1 Устройство управления	8
1.2 Ассоциативная память	9
1.3 Поиск	10
1.4 Зануление	10
1.5 Сохранение значения	10
2 Разработка устройства	11
2.1 Анализ исходных данных задания на курсовую работу	11
2.2 Спецификация устройства на уровне «черного ящика»	11
2.3 Представление черного ящика в виде операционной и управляющей частей	12
2.4 Разработка структуры операционной части устройства	12
2.5 Составление схемы алгоритма работы устройства и его микропрограммы	13
2.6 Разработка схемы алгоритма работы	13
2.7 Составление полной спецификации устройства	16
2.8 Разработка функциональной схемы	17
2.9 Контрольный пример	18
2.10 Временная диаграмма работы УУ	21
ЗАКЛЮЧЕНИЕ	22
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ	23
ПРИЛОЖЕНИЕ А	24

Перечень используемых условных обозначений, сокращений и терминов

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

ВВЕДЕНИЕ

Цель работы (проекта) – Разработать контроллер ассоциативной памяти, хранящей 64*8-разрядных двоичных кодов с выходом по «равно признаку» и «не равно признаку». Признаком поиска может быть 1,2,4,8 двоичных разрядов. Результатом поиска, выводимым на ШД, является весь байт. 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 С и результаты расчета контрольного примера.

1 Исследование предметной области курсовой работы

1.1 Устройство управления

Управляющее устройство (УУ) - устройство управления, часть вычислительной машины (ВМ), координирующая работу всех её устройств, предписывая им те или иные действия в соответствии с заданной программой. Управляющее устройство вырабатывает управляющие сигналы, обеспечивающие требуемую последовательность выполнения операций, контролирует работу машины в различных режимах, обеспечивает взаимодействие человека-оператора с ВМ.

Для выполнения своих функций УУ должно иметь входы, позволяющие определить состояние управляемой системы, и выходы, через которые реализуется управление поведением системы.

Входная информация:

Тактовые импульсы – с каждым тактовым импульсом УУ инициирует выполнение одной или нескольких микроопераций.

Код операции – код операции текущей команды поступает из регистра команды и используется, чтобы определить, какие микрооперации должны выполняться в течение машинного цикла.

Сигналы из системной шины – часть сигналов с системной шины, обеспечивающая передачу в управляющее устройства запросов прерывания, подтверждений и т.д.

В свою очередь УУ, а точнее микропрограммный автомат, формирует следующую выходную информацию:

Внутренние сигналы управления – эти сигналы воздействуют на внутренние схемы центрального процессора и относятся к одному из двух типов: тем, которые вызывают перемещение данных из регистра в регистр, и тем, что инициируют определенные функции операционного устройства ВМ.

Сигналы в системную шину – также относятся к одному из двух типов: управляющие сигналы в память и управляющие сигналы в модули ввода/вывода.

После извлечения команды из памяти, она загружается в регистр команд. Дешифратор команд, входящий в устройство управления, преобразует код команды в управляющие сигналы:

- внутренние, необходимые для считывания/записи данных в регистры и управления АЛУ;
- внешние, подаваемые на шину управления.

Арифметическо - логическое устройство (АЛУ) — блок процессора, который служит для выполнения арифметических и логических преобразований над словами, называемыми в этом случае операндами.

АЛУ в зависимости от выполнения функций можно разделить на две части:

2) операционное устройство (АЛУ), в котором реализуется заданная последовательность микрокоманд (команд).

В АЛУ выполняются требуемые операции:

- поиск
- зануление
- сохранение значения

1.2 Ассоциативная память

На рисунке П1 представлена схема ассоциативной памяти. Она состоит из управляющей части и операционной части. На рисунке П2 как будут храниться данные в ассоциативной памяти, поиск в которой будет реализовываться бинарной последовательностью, из которой будет получаться индекс в массиве.

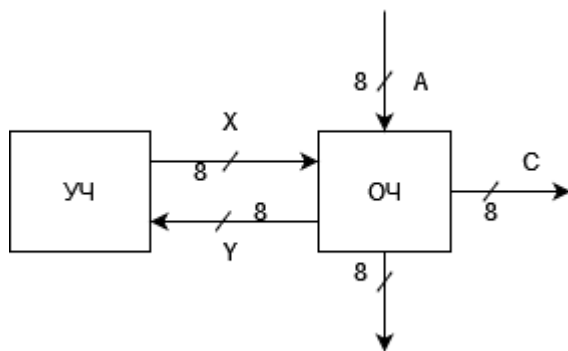


Рисунок П1. – Устройство ассоциативной памяти

Ассоциативная память		
Индекс	Адрес	Значение
0	00000000	0
1	00000001	0
...
62	00111110	0
63	00111111	0

Рисунок П2. – Представление ассоциативной памяти

1.3 Поиск

На вход поступает бинарная последовательность, которая будет преобразовываться в индекс, а на выходе байт, записанный в массиве по полученному индексу.

1.4 Зануление

На вход поступает бинарная последовательность, которая будет преобразовываться в индекс, по которому будет зануляться значение в массиве, а на выходе ответ о выполнении операции.

1.5 Сохранение значения

На вход поступает бинарная последовательность, которая будет преобразовываться в индекс, по которому будет записываться значение в массив, а на выходе ответ о выполнении операции.

2 Разработка устройства

2.1 Анализ исходных данных задания на курсовую работу

Согласно заданию, устройство должно быть предназначено для выполнения следующих операций:

- поиск;
- зануление;
- сохранение значения.

Разрядность операндов и результата должна быть – 8 бит.

Исходя из этого видно, что входы(А,В) и выход (результат операции С) должны иметь 8 разрядов.

2.2 Спецификация устройства на уровне «черного ящика»

Упрощенно разрабатываемое устройство можно представить как 3 команды, изображенные на рис.3-5.

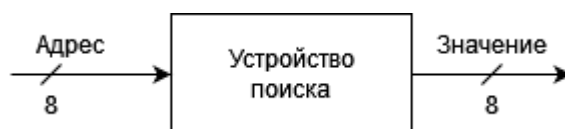


Рисунок П3. – Система выводов устройства поиска

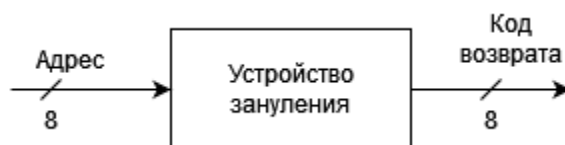


Рисунок П4. – Система выводов устройства зануления

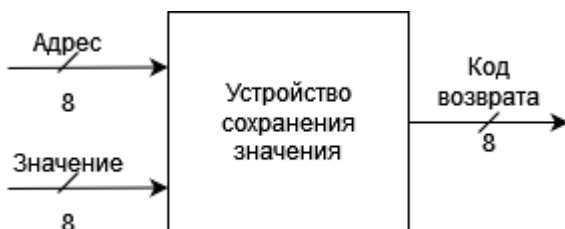


Рисунок П5. – Система выводов устройства сохранения значения

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

Пусть операнды размещаются в регистрах А и В, как показано на рисунке 6. Выделим три блока для каждой команды. После каждого выполнения команды значение операнда А изменяется, а значение операнда В остается неизменным. Результат выполнения трех команд (изменения операнда А) записывается в регистр С.

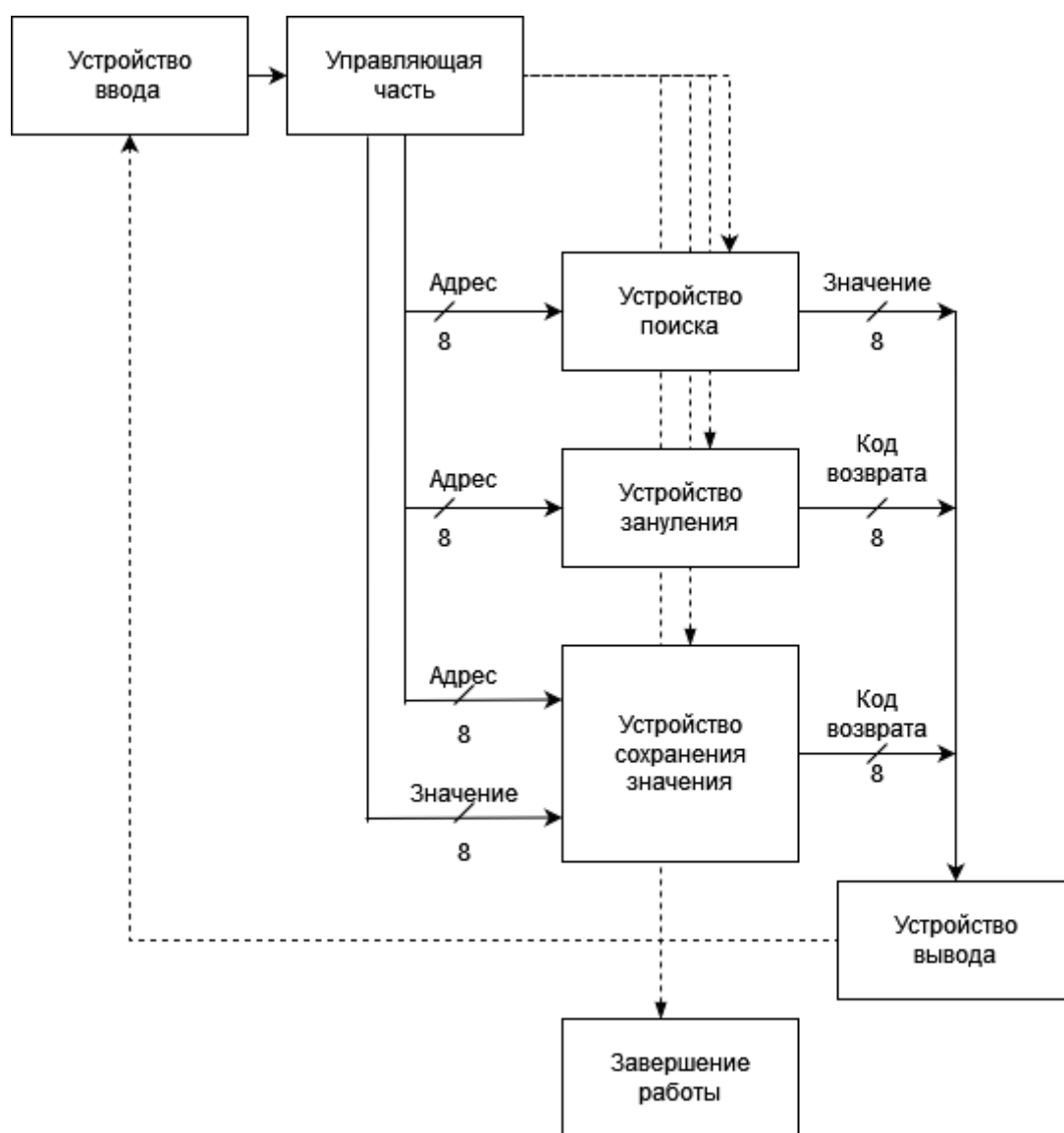


Рисунок П6. — Разбиение схемы устройств на ОЧ и УЧ.

2.4 Разработка структуры операционной части устройства

Пусть операнды размещаются в регистре А (уменьшаемое) и в регистре В (вычитаемое). Алгоритм приведен на рисунке 7.

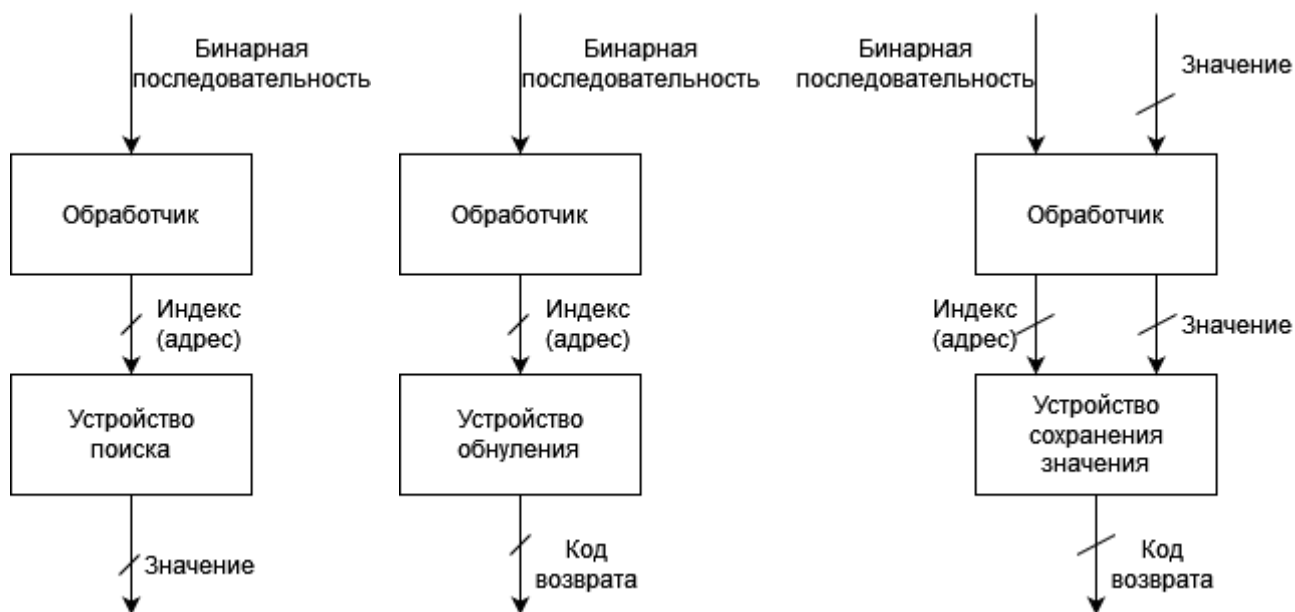


Рисунок П7. — Упрощенная структура ОЧ четырёх команд.

2.5 Составление схемы алгоритма работы устройства и его микропрограммы

Схема алгоритма выполнения трех команд (поиска, зануления и сохранения значения) приведена на рисунке П7.

Отметим операторные блоки символами Y_i , а логические блоки символами X_i .

Операторных блоков получилось 20, логических – 5.

Таким образом, для операционной части (ОЧ) устройства потребуется 20 сигналов управления из управляющей части устройства, а для управляющей части (УЧ) устройства – 5 осведомительных сигналов из операционной части.

2.6 Разработка схемы алгоритма работы

Схема алгоритма на уровне микроопераций изображена на рисунках 8-11.

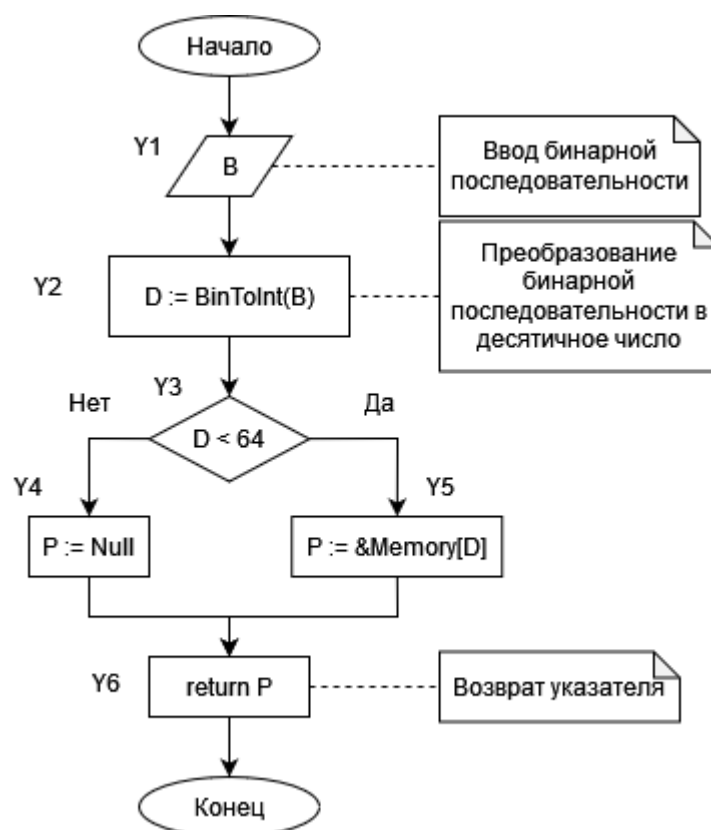


Рисунок П8. – Схема алгоритма возврата указателя ячейки, метод getPtr()

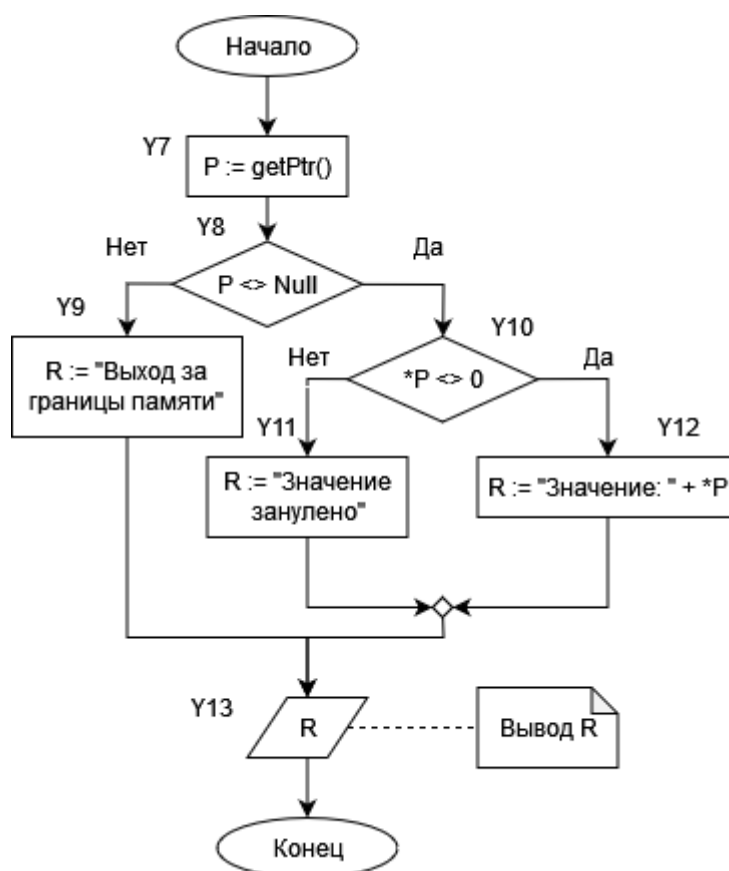


Рисунок П9. – Схема алгоритма чтения значения по указателя

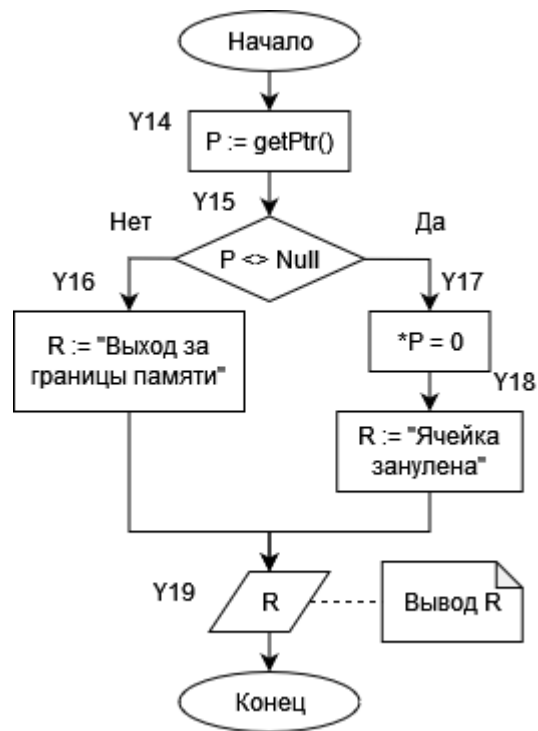


Рисунок П10. – Схема алгоритма чтения значения по указателя

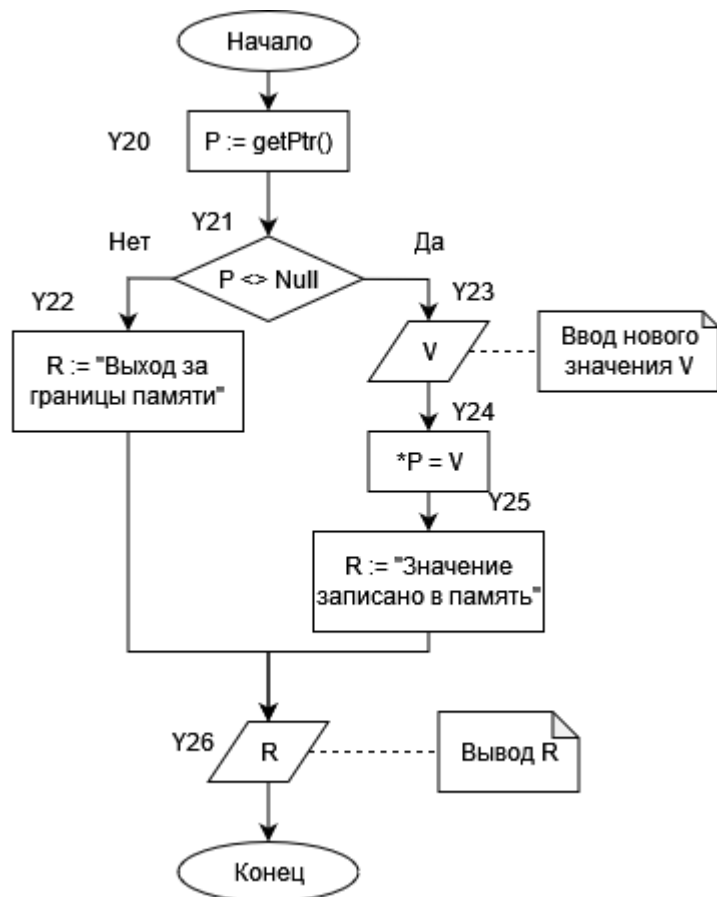


Рисунок П11. – Схема алгоритма чтения значения по указателю

2.7 Составление полной спецификации устройства

Опишем в таблице все линии и сигналы, полученные в процессе разработки ассоциативной памяти.

Таблица 1. Сигналы операционной части

Имя сигнала/шины и разрядность	Тип (In/Out)	Назначение сигнала
Y1	I для ОЧ	Ввод бинарной последовательности
Y2	I для ОЧ	Преобразование бинарной последовательности в десятичное число
Y3	I для ОЧ	Условие вхождения числа в диапазон адресов ассоциативной памяти
Y4	I для ОЧ	Присвоение указателю Null
Y5	I для ОЧ	Присвоение указателю адреса на ячейку памяти
Y6	O для ОЧ	Возврат указателя
Y7	I для ОЧ	Получение указателя
Y8	I для ОЧ	Проверка указателя на неравенство Null
Y9	I для ОЧ	Сохранение сообщения «Выход за границы памяти»
Y10	I для ОЧ	Проверка значения ячейки на неравенство нулю
Y11	I для ОЧ	Сохранение сообщения «Значение занулено»
Y12	I для ОЧ	Сохранение сообщения «Значение: » + Значение ячейки
Y13	O для ОЧ	Вывод сообщения
Y14	I для ОЧ	Получение указателя
Y15	I для ОЧ	Проверка указателя на неравенство Null
Y16	I для ОЧ	Сохранение сообщения «Выход за границы памяти»
Y17	I для ОЧ	Зануление ячейки памяти

Продолжение таблицы 1. Сигналы операционной части

Имя сигнала/шины и разрядность	Тип (In/Out)	Назначение сигнала
Y18	О для ОЧ	Вывод сообщения
Y19	И для ОЧ	Получение указателя
Y20	И для ОЧ	Проверка указателя на неравенство Null
Y21	И для ОЧ	Сохранение сообщения «Выход за границы памяти»
Y22	И для ОЧ	Ввод нового значения
Y23	И для ОЧ	Присвоение ячейке памяти нового значения
Y24	И для ОЧ	Сохранение сообщения «Значение записано в память»
Y25	О для ОЧ	Вывод сообщения

2.8 Разработка фрагмента функциональной схемы ассоциативной памяти

Фрагмент схемы УЧ дан на рисунке П10. Схема составлена с блока Y8 по блок Y12 (см. рисунок П5).

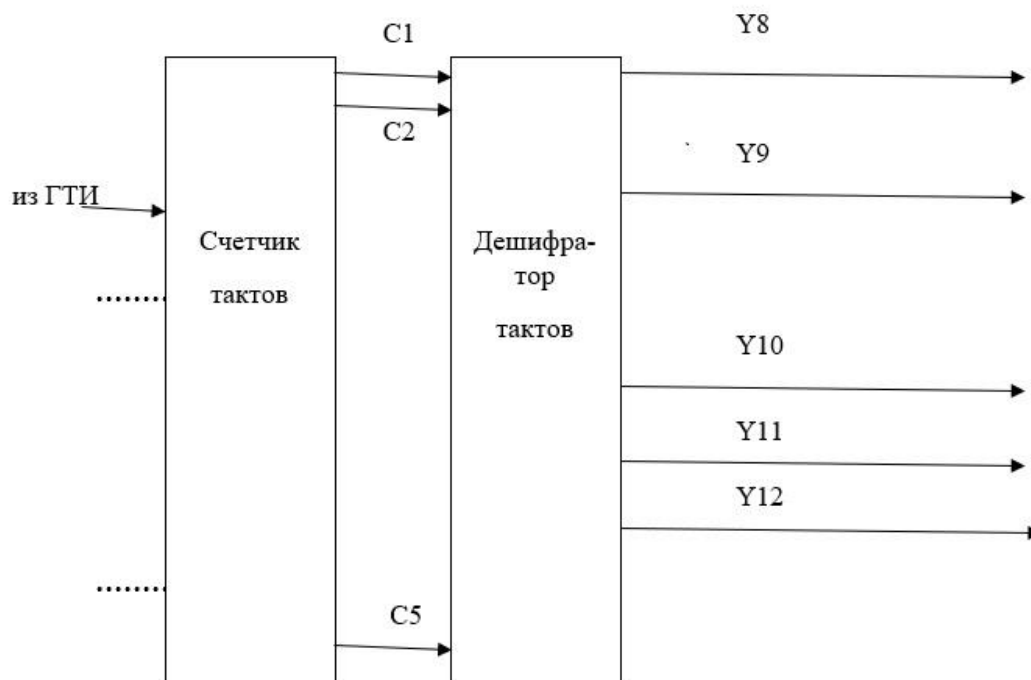


Рисунок П12. - Фрагмент схемы УЧ

2.9 Контрольный пример

Выполнение программы для исходных данных показано на рисунке П11.

Листинг программы представлен в приложении А.

```
Меню:
  1. Создать устройство ассоциативной памяти.
  2. Вывести все записи.
  3. Прочитать из памяти.
  4. Записать в память.
  5. Удалить из памяти.
  6. Удалить устройство ассоциативной памяти.
  (*). Выход.
Введите номер действия: 1
Устройство ассоциативной памяти создано.
```

Рисунок П13. – Создание устройства ассоциативной памяти

```
Меню:
  1. Создать устройство ассоциативной памяти.
  2. Вывести все записи.
  3. Прочитать из памяти.
  4. Записать в память.
  5. Удалить из памяти.
  6. Удалить устройство ассоциативной памяти.
  (*). Выход.
Введите номер действия: 2
Содержимое памяти:
  <пусто>
Всего записей: 0
```

Рисунок П14. – Создание устройства ассоциативной памяти

```
Меню:
  1. Создать устройство ассоциативной памяти.
  2. Вывести все записи.
  3. Прочитать из памяти.
  4. Записать в память.
  5. Удалить из памяти.
  6. Удалить устройство ассоциативной памяти.
  (любое другое число). Выход.
Введите номер действия: 3
Введите индекс: 11
Значение по адресу 00000011 - занулено.
```

Рисунок П15. – Создание устройства ассоциативной памяти

```
Меню:
  1. Создать устройство ассоциативной памяти.
  2. Вывести все записи.
  3. Прочитать из памяти.
  4. Записать в память.
  5. Удалить из памяти.
  6. Удалить устройство ассоциативной памяти.
  (*). Выход.
Введите номер действия: 4
Введите индекс: 11
Введите значение: e
00000011 -> e
```

Рисунок П16. – Создание устройства ассоциативной памяти

```
Меню:
  1. Создать устройство ассоциативной памяти.
  2. Вывести все записи.
  3. Прочитать из памяти.
  4. Записать в память.
  5. Удалить из памяти.
  6. Удалить устройство ассоциативной памяти.
  (*). Выход.
Введите номер действия: 2
Содержимое памяти:
  00000011 -> e
  00101010 -> г
Всего записей: 2
```

Рисунок П17. – Создание устройства ассоциативной памяти

```
Меню:
  1. Создать устройство ассоциативной памяти.
  2. Вывести все записи.
  3. Прочитать из памяти.
  4. Записать в память.
  5. Удалить из памяти.
  6. Удалить устройство ассоциативной памяти.
  (*). Выход.
Введите номер действия: 5
Введите индекс: 11
Байт по индексу 00000011 обнулён.
```

Рисунок П18. – Создание устройства ассоциативной памяти

```
Меню:
1. Создать устройство ассоциативной памяти.
2. Вывести все записи.
3. Прочитать из памяти.
4. Записать в память.
5. Удалить из памяти.
6. Удалить устройство ассоциативной памяти.
(*). Выход.
Введите номер действия: 6
Устройство ассоциативной памяти удалено.
```

Рисунок П19. – Создание устройства ассоциативной памяти

```
Меню:
1. Создать устройство ассоциативной памяти.
2. Вывести все записи.
3. Прочитать из памяти.
4. Записать в память.
5. Удалить из памяти.
6. Удалить устройство ассоциативной памяти.
(любое другое число). Выход.
Введите номер действия: 7
Спасибо за работу! До свидания!

Process finished with exit code 0
```

Рисунок П20. – Создание устройства ассоциативной памяти

2.10 Временная диаграмма работы УУ

На рисунке П21 приведена временная диаграмма работы УУ.

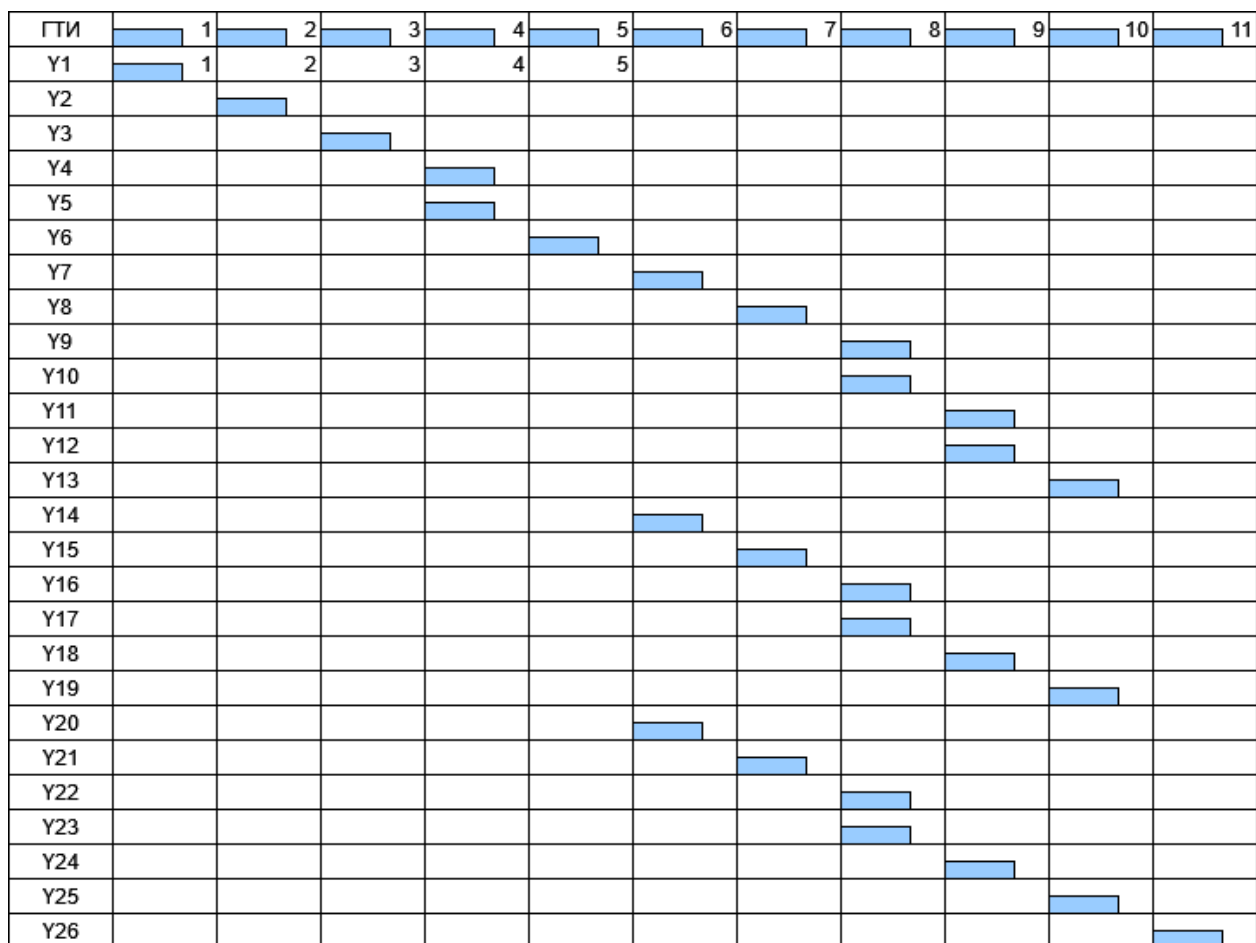


Рисунок П21. – Фрагмент временной диаграммы работы операционной части

Закрашенные интервалы времени соответствуют логическим 1, горизонтальные штриховые линии указывают интервалы времени, в которых значения X не имеют смысла, т.к. в эти интервалы сигналы X не проверяются в УЧ. Вертикальные штриховые линии разделяют временные такты.

ЗАКЛЮЧЕНИЕ

Задача курсовой работы – разработка алгоритма работы и структуры работы устройства для выполнения четырех команд

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 и временная диаграмма работы устройства.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Павловская Т.А. С/С++. Программирование на языке высокого уровня: Учеб. пособие. – СПб.:Питер, 2007. – 461 с.
2. Павловская Т.А., Щупак Ю.А. С/С++. Программирование на языке высокого уровня. Структурное программирование: Практикум. – СПб.:Питер, 2003. – 240 с.
3. Жмакин А. П. Архитектура ЭВМ: 2-е изд., перераб. и доп.: учеб. пособие. — СПб.: БХВ-Петербург, 2010. — 352 с.

ПРИЛОЖЕНИЕ А

Листинг файла main.c

```
#include "assoc.h"
#include <locale.h>

int main() {
    setlocale(LC_ALL, "Russian");
    run();
    return 0;
}
```

Листинг файла assoc.h

```
#pragma once

void run();

void create(void **mem);
void out(void **mem);
void read(void **mem);
void write(void **mem);
void rem(void **mem);
void drop(void **mem);
```

Листинг файла assoc.c

```
#include "assoc.h"

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef unsigned char byte;
typedef void (*task)(void**);

struct bin {
    char s[9];
};

const byte BYTE_COUNT = 64;

const char *DIALOG =
    "Меню:\n"
    "\t1. Создать устройство ассоциативной памяти.\n"
    "\t2. Вывести все записи.\n"
    "\t3. Прочитать из памяти.\n"
    "\t4. Записать в память.\n"
    "\t5. Удалить из памяти.\n"
    "\t6. Удалить устройство ассоциативной памяти.\n"
    "\t(любое другое число). Выход.\n"
    "Введите номер действия: ";

const char *INCORRECT = "Некорректный индекс!";

const task TASKS[] = {
    &create,
    &out,
```

```

        &read,
        &write,
        &rem,
        &drop
};

int read_index(byte *result);
byte read_value();
struct bin byte_to_bin(byte b);
byte bin_to_byte(struct bin b);

void run() {
    int number;
    void *mem = NULL;
    while(1) {
        printf("%s", DIALOG);
        scanf("%d", &number);
        if(1 <= number && number <= 6) {
            if(number > 1 && mem == NULL) {
                printf("Устройство ассоциативной памяти не было создано!\n\n");
                continue;
            } else {
                TASKS[number - 1](&mem);
                printf("\n");
            }
        } else {
            printf("Спасибо за работу! До свидания!\n");
            return;
        }
    }
}

void create(void **mem) {
    if(*mem) {
        free(*mem);
    }
    *mem = malloc(BYTE_COUNT);
    for(byte i = 0; i < BYTE_COUNT; ++i) {
        ((byte*)*mem)[i] = 0;
    }
    printf("Устройство ассоциативной памяти создано.\n");
}

void out(void **mem) {
    byte count = 0;
    printf("Содержимое памяти:\n");
    for(byte i = 0; i < BYTE_COUNT; ++i) {
        byte value = ((byte*)*mem)[i];
        if(value) {
            printf("\t%s -> %c\n", byte_to_bin(i).s, value);
            ++count;
        }
    }
    if(count == 0) {
        printf("\t<пусто>\n");
    }
    printf("Всего записей: %d\n", count);
}

void read(void **mem) {

```

```

byte index;
if (read_index(&index)) {
    byte value = ((byte *) *mem)[index];
    if (value) {
        printf("%s -> %c\n", byte_to_bin(index).s, value);
    } else {
        printf("Значение по адресу %s - занулено.\n", byte_to_bin(index).s);
    }
} else {
    printf("%s\n", INCORRECT);
}
}

void write(void **mem) {
    byte index;
    if (read_index(&index)) {
        byte value = read_value();
        ((byte *) *mem)[index] = value;
        printf("%s -> %c\n", byte_to_bin(index).s, value);
    } else {
        printf("%s\n", INCORRECT);
    }
}

void rem(void **mem) {
    byte index;
    if (read_index(&index)) {
        if (((byte *) *mem)[index]) {
            ((byte *) *mem)[index] = 0;
            printf("Байт по индексу %s обнулён.\n", byte_to_bin(index).s);
        } else {
            printf("Байт по индексу %s уже был обнулён.\n", byte_to_bin(index).s);
        }
    } else {
        printf("%s\n", INCORRECT);
    }
}

void drop(void **mem) {
    free(*mem);
    *mem = NULL;
    printf("Устройство ассоциативной памяти удалено.\n");
}

int read_index(byte *result) {
    struct bin b;
    printf("Введите индекс: ");
    scanf("%s", b.s);
    *result = bin_to_byte(b);
    getchar();
    return *result < 64;
}

byte read_value() {
    byte value;
    printf("Введите значение: ");
    scanf("\n%c", &value);
    return value;
}

```

```

struct bin byte_to_bin(byte b) {
    struct bin r;
    r.s[8] = '\0';
    for(int i = 0; i < 8; ++i) {
        r.s[i] = (b & 128) ? '1' : '0';
        b <<= 1;
    }
    return r;
}

byte bin_to_byte(struct bin b) {
    byte r = 0;
    for(byte i = 0, n = strlen(b.s); i < n; ++i) {
        r <<= 1;
        r |= (b.s[i] != '0' ? 1 : 0);
    }
    return r;
}

```