Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

*Кафедра вычислительных систем*

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ**

Вариант

Выполнил:

Студент второго курса

Группы ИП-412

Максимов А. С.

Проверил:

Работа защищена

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_2018 г.

С оценкой «\_\_\_\_\_»

Новосибирск 2018г

Оглавление

[**1.1.** **Описание работы** 4](#_Toc531814451)

[**1.2.** **Транслятор с языка Simple Assembler** 4](#_Toc531814452)

[**1.3.** **Транслятор с языка Simple Basic** 5](#_Toc531814453)

[**2.1. Программная реализация** 5](#_Toc531814454)

[**2.2. Блок – схема используемых алгоритмов** 7](#_Toc531814455)

[**2.2.1. Основной модуль** 7](#_Toc531814456)

[**2.2.2. Функция CU** 9](#_Toc531814457)

[**2.2.3. Функция ALU** 11](#_Toc531814458)

[**3.1. Результат** 12](#_Toc531814459)

[**Листинг** 14](#_Toc531814460)

[**Командная панель** 14](#_Toc531814461)

[**Транслятор с языка Simple Assembler** 14](#_Toc531814462)

[**Транслятор с языка Simple Basic на язык Simple Assembler** 14](#_Toc531814463)

[**Список литературы** 15](#_Toc531814464)

**Введение**

Устройство управления осуществляет координацию работы всех других устройств в составе ЭВМ. В качестве исходных данных выступают:

1) Код операции

2) Текущее состояние устройств в составе ЭВМ

В качестве результата выступают управляющие сигналы.

Управляющие сигналы в закодированном виде хранятся в памяти и, в зависимости от входных сигналов УУ, считываются из памяти, декодируются и подаются на другие устройства. После считывания очередной машинной команды поле кода операции сохраняется в регистре операций, откуда код подается на устройство формирования адреса. Также на это устройство подаются состояния всех устройств в составе ЭВМ. УФА на основе комбинации состояний и кода операции текущей машинной команды формирует адрес микрокоманды. По этому адресу из банка памяти микрокоманд извлекается микрокоманда, которая сохраняется в регистре микрокоманд. Каждая микрокоманда содержит два поля: код (закодированная информация о необходимых управляющих сигналах) и адрес следующей микрокоманды, т. к. во многих случаях машинная команда для своей реализации требует нескольких комбинаций управляющих сигналов – таким образом это адресное поле в совокупности с состоянием устройств в составе ЭВМ используется для формирования адреса следующей микрокоманды. Данный код подается на дешифратор. Дешифратор раскодирует информацию об управляющих сигналах и подает управляющие сигналы на все устройства в составе ЭВМ. Т. к. микрокоманды хранятся в памяти, изменением содержимого памяти можно менять логику работы УУ.

В современных ЭВМ общего назначения используется микропрограммное УУ. При этом в ряде специализированных ВМ используется УУ с жесткой логикой.

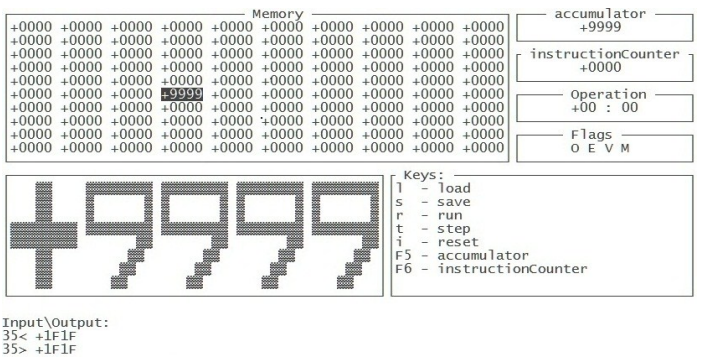
В рамках курсовой работы необходимо доработать модель Simple Computer так, чтобы она обрабатывала команды, записанные в оперативной памяти.

* 1. **Описание работы**

В рамках курсовой работы необходимо доработать модель Simple Computer так, чтобы она обрабатывала команды, записанные в оперативной памяти, а также реализовать команду, соответствующую варианту выполняемого задания:

|  |  |  |
| --- | --- | --- |
| JP | 58 | Переход к указанному адресу памяти, если результат предыдущей операции чётный |

Для управления моделью (определения начальных состояний узлов Simple Computer, запуска программ на выполнения, отражения хода выполнения программ) требуется создать консоль.



* 1. **Транслятор с языка Simple Assembler**

Разработка программ для Simple Computer может осуществляться с использованием низкоуровневого языка Simple Assembler. Для того чтобы программа могла быть обработана Simple Computer, необходимо реализовать транслятор, переводящий текст Simple Assembler в бинарный формат, которым может быть считан консолью управления.

Пример программы на Simple Assembler:

00 READ 09 ; (Ввод А)

01 READ 10 ; (Ввод В)

02 LOAD 09 ; (Загрузка А в аккумулятор)

03 SUB 10 ; (Отнять В)

04 JNEG 07 ; (Переход на 07, если отрицательное)

05 WRITE 09 ; (Вывод А)

06 HALT 00 ; (Останов)

07 WRITE 10 ; (Вывод В)

08 HALT 00 ; (Останов)

09 = +0000 ; (Переменная А)

10 = +9999 ; (Переменная В)

Программа транслируется по строкам, задающим значение одной ячейки памяти. Каждая строка состоит как минимум из трех полей: адрес ячейки памяти, команда (символьное обозначение), операнд. Четвертым полем может быть указан комментарий, который обязательно должен начинаться с символа точка с запятой. Дополнительно используется команда =, которая явно задает значение ячейки памяти в формате вывода его на экран консоли (+XXXX)

* 1. **Транслятор с языка Simple Basic**

Для упрощения программирования пользователю модели Simple Computer должен быть предоставлен транслятор с высокоуровневого языка Simple Basic. Файл, содержащий программу на Simple Basic, преобразуется в файл с кодом Simple Assembler. Затем Simple Assembler - файл транслируется в бинарный формат.

В языке Simple Basic используются следующие операторы: rem, input, output, goto, if, let, end.

Пример программы на Simple Basic:

10 REM Это комментарий

20 INPUT A

30 INPUT B

40 LET C = A – B

50 IF C < 0 GOTO 20

60 PRINT C

70 END

**2.1. Программная реализация**

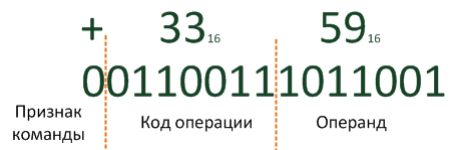
Память состоит из ячеек (массив), каждая из которых хранит 15 двоичных разрядов. Ячейка – минимальная единица, к которой можно обращаться при доступе к памяти. Все ячейки последовательно пронумерованы целыми числами. Номер ячейки является её адресом и задается 7-миразрядным числом. Предполагаем, что Simple Computer оборудован памятью из 100 ячеек (с адресами от 0 до 99).

Регистры являются внутренней памятью процессора. Центральный процессор Simple Computer имеет: аккумулятор, используемый для временного хранения данных и результатов операций, счетчик команд, указывающий на адрес ячейки памяти, в которой хранится текущая выполняемая команда и регистр флагов, сигнализирующий об определённых событиях. Аккумулятор имеет разрядность 15 бит, счетчика команд – 7 бит. Регистр флагов содержит 5 разрядов: переполнение при выполнении операции, ошибка деления на 0, ошибка выхода за границы памяти, игнорирование тактовых импульсов, указана неверная команда.

Арифметико-логическое устройство (англ. arithmetic and logic unit, ALU) — блок процессора, который служит для выполнения логических и арифметических преобразований над данными. В качестве данных могут использоваться значения, находящиеся в аккумуляторе, заданные в операнде команды или хранящиеся в оперативной памяти. Результат выполнения операции сохраняется в аккумуляторе или может помещаться в оперативную память. В ходе выполнения операций АЛУ устанавливает значения флагов «деление на 0» и «переполнение».

Управляющее устройство (англ. control unit, CU) координирует работу центрального процессора. По сути, именно это устройство отвечает за выполнение программы, записанной в оперативной памяти. В его функции входит: чтение текущей команды из памяти, еѐ декодирование, передача номера команды и операнда в АЛУ, определение следующей выполняемой команды и реализации взаимодействий с клавиатурой и монитором. Выбор очередной команды из оперативной памяти производится по сигналу от системного таймера. Если установлен флаг «игнорирование тактовых импульсов», то эти сигналы устройством управления игнорируются. В ходе выполнения операций устройство управления устанавливает значения флагов «указана неверная команда» и «игнорирование тактовых импульсов»

Получив текущую команду из оперативной памяти, устройство управления декодирует её с целью определить номер функции, которую надо выполнить и операнд. Формат команды следующий: старший разряд содержит признак команды (0 – команда), разряды с 8 по 14 определяют код операции, младшие 7 разрядов содержат операнд.



**2.2. Блок – схема используемых алгоритмов**

## **2.2.1. Основной модуль**

Нет

Да

Нет

Да

Клавиша R?

Клавиша стрелочка?

Изменение положения курсора

Считывание клавиши

Инициализация памяти, привязка сигналов к функции

Начало

Установка флага «Игнорирование тактовых импульсов».

Вызов alarm(CU)

Конец

Клавиша Ctrl - C?

Да

Нет

Да

Нет

Редактирование

Клавиши редактирования?

## **2.2.2. Функция CU**

Нет

Да

Команда равна X?

Начало

Декодирование команды

Команда арифметико – логическая?

Да

Нет

Вызов функции ALU, передаем в нее команду и операнд

Конец (вернуть значение ALU)

Выполнение команды возможно?

Да

Нет

Конец (вернуть значение instructionCounter + 1)

Выполнение команды

Остановка таймера, установить флаг «неизвестная команда / неверная команда»

Конец (вернуть значение instructionCounter)

## **2.2.3. Функция ALU**

Начало

Команда равна X?

Да

Конец (вернуть -1)

Конец (вернуть 0)

Нет

Да

Выполнение команды

Выполнение команды возможно?

# **3.1. Результат**

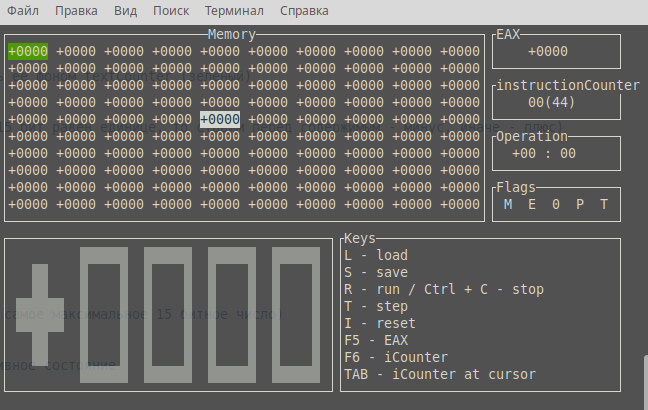
В ходе выполнения работы получилось три проекта: командная панель, транслятор с языка Simple Assembler, транслятор с высокоуровневого языка Simple Basic на Simple Assembler.

Командная панель представляет собой приложение, выполняющее операции, задаваемые в оперативную память ЭВМ.

Транслятор с языка Assembler позволяет перевести текст Simple Assembler в бинарный формат, который может быть считан командной панелью.

Транслятор с высокоуровневого языка Simple Basic преобразовывает файл, содержащий программу на Simple Basic в файл с кодом Simple Assembler

Реализованная командная панель предоставлена ниже:



**Вывод**

В ходе выполнения проекта мне удалось максимально близко воссоздать процесс выполнения команд в ЭВМ. Была реализован простейший алгоритм работы ЭВМ, считывание команды, декодирование и её дальнейшее выполнение. Также были реализованы трансляторы с языков Simple Assembler и Simple Basic. Последний потребовал глубокого понимания языка Simple Assembler.

# **Листинг**

## **Командная панель**

## **Транслятор с языка Simple Assembler**

## **Транслятор с языка Simple Basic на язык Simple Assembler**

# **Список литературы**

1. Организация ЭВМ и систем. Практикум // С.Н. Мамойленко, Новосибирск: ГОУ ВПО «Сиб- ГУТИ», 2005 г., URL:

2. Архитектура компьютера. 4-е изд. // Э. Танненбаум. – СПб.: Питер, 2003.

3. Организация ЭВМ. 5-е изд. / К. Хамахер, З. Вранешич, С. Заки. – СПб.: Питер; Киев: Изда- тельская группа BHV, 2003.

4. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем: учебник для ВУЗов. – СПб.: Питер, 2004.