МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»

Институт прикладной математики и механики

Высшая школа прикладной математики и вычислительной физики

Секция «Телематика»

Отчет о прохождении производственной практики

Разработка специализированного процессора для сортировки массива алгоритмом сортировки вставками

Студент:

Верещагина С. Е.
группа: 3630201/90002

Проверил:

старший преподаватель,
Чуватов М. В.

Санкт-Петербург – 2021

Содержание

Введение

В данном отчете представлено описание этапов проектирования специализированного процессора для решения задачи сортировки массивов, состоящих из целочисленных положительных элементов.

Отчет состоит из трех основных частей, в которых рассматриваются алгоритм сортировки, система команд, архитектура процессора, способного реализовать алгоритм сортировки вставкой.

Архитектура процессора и набор команд разработаны для выполнения узкого набора операций.

Работа была выполнена в графическом инструменте для разработки и моделирования логических схем Logisim.

1 Постановка задачи

Необходимо реализовать специализированный процессор для сортировки массива алгоритмом сортировки вставками.

Для проектирования процессора требуется:

- Определиться с архитектурой проектируемого процессора
- Разработать точный набор инструкций

Необходимо продумать:

- Какие команды реализовать?
- Как сделать адресацию массива?
- Каким образом хранить данные, в каком регистре?
- Как реализовать запись в массив?
- Как сделать чтение из массива?
- Как сделать перестановку данных?

Ограничения:

1. Исходные данные (размещаются в ОЗУ):
   - одномерный массив произвольных целых чисел;
   - диапазон принимаемых элементами значений от -32768 до 32767;
   - количество элементов от 2 до 64.
2. Ограничения среды моделирования:
   - разрядности параллельных интерфейсов (шин) – не более 16 бит;
   - разрядность командного регистра – не более 16 бит;
   - разрядности регистров общего назначения, сумматора, логических элементов – не более 8 бит;
   - принстонская архитектура (общее ОЗУ для команд и данных, общая шина для команд, адресов, данных);
   - по окончании работы программы на индикаторе должны выводиться элементы отсортированного массива, узлы дерева в порядке обхода выбранным методом, начальный и конечный индексы всех найденных совпадений подстроки с исходной строкой.

2 Решение поставленной задачи

2.1 Структура данных

Данные, хранящиеся в памяти можно подразделить на 3 типа.

1. Команда, в которой старший байт занимает код команды, а в младшем байте лежит операнд, если такой имеется.
2. Переменная – занимает младший байт ячейки памяти.
3. Элемент массива – 2 байта, занимает всю ячейку памяти, сначала идет младший байт, затем старший.

Для реализации алгоритма сортировки вставкой необходимо следующее:

- Сортируемый массив положительных чисел. Располагается в ОЗУ, начиная с адреса 0xa0.
- Количество элементов массива, лежит по адресу 0xff.
- Удвоенное количество элементов массива. 8-битное целое число число, располагается в оперативной памяти по адресу 0xfe. Необходимо, так как в данной реализации элемент массива занимает две ячейки памяти.
- Счетчик итераций. С помощью него мы либо переходим влево, отнимая зачение 2, либо вправо, прибавляя значение 2.

2.2 Архитектура процессора

В разработанном процессоре инструкции и данные хранятся в одной оперативной памяти – ОЗУ. Это является признаком Принстонской архитектуры, а так же архитектуры Фон Неймана. Последняя является очень гибкой и просто релизуемой архитектурой, однако подверженной ошибкам, вроде переполнения буфера.[1]

Сущность фон-неймановской концепции вычислительной машины можно свести к четырем принципам:

- двоичного кодирования;
- программного управления;
- однородности памяти;
- адресности. [1]

2.3 Модель памяти

На разработанной схеме присутствуют два блока памяти: ОЗУ и ПЗУ. В ОЗУ хранятся закодированная последовательность команд, сортируемый массив и все необходимые переменные для реализации алгоритма сортировки. Декодер представляет собой ПЗУ.

2.3.1 ОЗУ

В спроектированном процессоре команды и операнды хранятся в одной области памяти. Оперативная память имеет разрядность 8 бит. Разрядность адреса также 8 бит. Для чтения команды из оперативной памяти в адресный регистр записывается значение счетчика команд. Затем по записанному адресу считывается команда и записывается в регистр.

При обращении на чтение или запись – обмен данными происходит через шину. На рис. 1 представлена ОЗУ, где лежит программа сортировки вставкой.

```
00 24 02 34 fa  14 fa 15 fe  80 55 10 fa  21 02 40 21
10 78 94 21 01  50 21 78 92  21 01 50 21  78 95 21 01
20 50 21 78 93  70 2e 10 fa  21 02 50 30  fa 60 04 10
30 fa 21 02 40  21 78 a5 21  01 50 21 78  a3 21 01 50
40 21 78 a4 21  01 50 21 78  a2 21 02 10  fa 44 34 fa
50 60 04 14 fa  60 04 24 00  34 fa 60 63  10 fa 21 02
60 54 34 fa 15  fe 80 74 10  fa 21 78 94  21 01 50 21
70 78 92 b0 60  5c ff 00 00  02 00 08 00  00 00 00 00
80 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
90 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
a0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
b0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
c0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
d0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
e0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
f0 00 00 00 00  00 00 00 00  00 00 00 00  00 00 04 02
```

Рис. 1: ОЗУ с данными и закодированной последовательностью команд.

2.3.2 ПЗУ

В данной памяти хранятся команды из системы команд. Память имеет разрядность данных 18 бит и работает только на чтение. Данные после тения сразу поступают в декодер. Адрес поступает из командного регистра. На рис. 2 представлена ПЗУ в которой хранятся команды из системы команд.

```
00 00000 00000 00000 00000  08800 06400 00000 00000  08800 04500 00880 00500  08800 06400 00000 00000
10 08800 04500 00000 00000  08800 06400 00000 00000  08800 04402 00810 00280  08800 06400 00000 00000
20 10100 00000 00000 00000  08800 06400 00000 00000  00140 00000 00000 00000  08800 06400 00000 00000
30 08800 00402 00811 02400  08800 06400 00000 00000  08802 04420 00819 04000  08800 06400 00000 00000
40 08802 04420 00815 04000  08800 06400 00000 00000  00840 00500 00810 00000  08800 06400 00000 00000
50 00840 00280 00810 00000  08800 06400 00000 00000  20000 00000 00000 00000  08800 06400 00000 00000
60 00000 00000 00000 00000  08800 06400 00000 00000  00000 00000 00000 00000  00000 00000 00000 00000
70 00000 00000 00000 00000  00000 00000 00000 00000  00000 00000 00000 00000  00000 00000 00000 00000
```

Рис. 2: ПЗУ

2.3.3 Регистры общего назначения

Спроектированный процессор имеет 5 регистров общего назначения.

Код регистра используется для мультиплексирования. Он указывается в младшем бите команды, если аргументом является регистр.

Регистр R0 содержит либо счетчик итераций, либо с его помощью можно обратиться к одной из частей элемента массива. Элемент массива хранится в двух ячейках и прибавляя единицу к тому, что лежит в регистре R0 мы переходим старшим или к младшим битам.

Регистр R1 содержит значение, которое либо прибавляется к счетчику, либо отнимается. Также в данный регистр заносится адрес нулевого элемента, чтобы была возможность обратиться к любому элементу массива.

Регистры R0 и R1 складываются. Выход сумматора заносится в адресный регистр и по этому значению в адресном регистре считывается элемент и записывается в регистр, который нам нужен.

Регистр R2 хранит старшие биты первого элемента.

Регистр R3 хранит старшие биты второго элемента.

Регистр R4 хранит младшие биты первого элемента.

Регистр R5 хранит младшие биты второго элемента.

2.3.4 Регистры условного и безусловного перехода

С помощью регистров **JR и RTSK** реализуется логика безусловных и условных переходов.

В регистр JR записывается адрес, по которому нужно перейти в случае выполнения условия. Это регистр условного перехода.

RTSK - регистр текущего счетчика команд. В регистр RTSK сохраняется значение текущего счетчика команд. При выполнении условия перехода, текущее значение счетчика команд сохраняется в данный регистр.

При обращении к регистрам на чтение и запись данные поступают с шины.

2.3.5 Арифметико-логическое устройство

Арифметико-логическое устройство (далее АЛУ) состоит из сумматора, декодера и самостоятельно реализованного компаратора. С помощью АЛУ возможно выполнять операции сложения и вычитания. Также используются декодер и элемент И для того чтобы включить регистр, номер которого мы указали в командном регистре. Исполнительные устройства имеют разрядность 8 бит.

Операнды в АЛУ поступают с шины. Результат также поступает на шину.

2.3.6 Система команд

1. NOP – загружает команду по адресу из счетчика команд в регистр команд.
2. LDR Rn [mem] – сохраняет в регистр Rn значение ячейки памяти, адрес которой хранится в операнде.
3. LDI Rn val – сохраняет в регистр Rn значение переменной.
4. STR Rn [mem] – сохраняет значение регистра Rn по адресу в памяти.
5. SUB Rn – сохраняет в регистр Rn разность значений, лежащих в регистрах R0 и R1.
6. ADD Rn – сохраняет в регистр Rn сумму значений, лежащих в регистрах R0 и R1.
7. UCJ Rn [mem] – команда безусловного перехода. Реализует безусловный переход к команде по адресу в памяти.
8. CJ1 [mem] – условный переход к команде по адресу в памяти, если выполнено условие, что первый элемент больше следующего за ним элемента
9. CJ2 [mem] – условный переход к команде по адресу в памяти, если выполнено условие, что элемент 1 больше элемента 2, либо равен ему. Данная команда реализована для проверки выполнения цикла while.
10. VAL Rn – сохраняет в регистр Rn значение по адресу. Адрес представляет собой сумму операндов, лежащих в регистрах R0 и R1
11. VALS Rn – сохраняет в памяти значение регистра Rn по адресу, который является суммой значений регистров R0 и R1.
12. OUT – выводит на экран значение регистра вывода.
13. STOP – при выполнении данной команды выключается счетчик команд, в командный регистр перестают записываться новые значения.

3 Реализация

3.1 Сортировка вставкой

Сортировка вставками (англ. Insertion sort) — алгоритм сортировки, в котором элементы входной последовательности просматриваются по одному, и каждый новый поступивший элемент размещается в подходящее место среди ранее упорядоченных элементов. Вычислительная сложность — $O(n^2)$

```
for i = 2 to n do
x = A[i]
j = i
while (int j > 1 and A[j-1] > x) do
A[j] = A[j-1]
j = j - 1
end while
A[j] = x
end for
```

3.2 Общая схема

Была реализована логическая схема процессора в программе Logisim. Процессор является машиной фон Неймана, поэтому для данных, адресов и команд используется общая 8-битная шина. При работе компьютера команды извлекаются по адресу, хранящемуся в счетчике команд. Сохранение значений в регистры общего назначения, адресный регистр, счетчик команд и командный регистр реализуется с помощью управляющих сигналов, которые осуществляют вывод данных на шину и запись в необходимый регистр. Также из-за ограничений на разрядность регистров при сравнении двух 16-битных значений первое число загружается в регистры R2 и R4, в регистр R2 - старший байт, в регистр R4 - младший байт, а второе в регистры R3 и R5 - в регистр R3 - старший байт, в регистр R5 - младший байт.

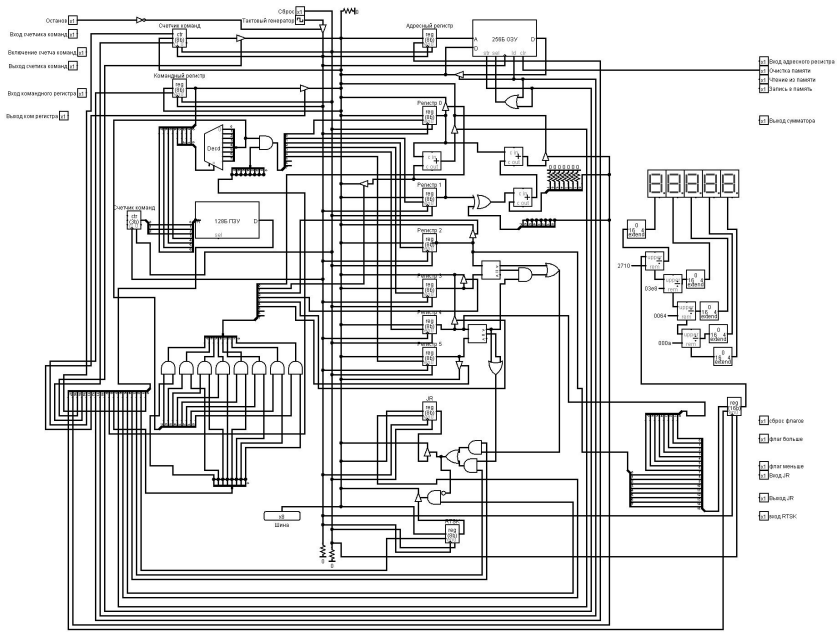

Рис. 3: Общая схема

3.3 Регистры общего назначения

На разработанной схеме присутствуют пять регистров общего назначения: R0, R1, R2, R3, R4, R5. Разрядность регистров 8 бит. Входы и выходы регистров общего назначения управляются декодером, значения передаются по общей шине.

Входы регистров JR и RTSK управляются с помощью управляющих сигналов. Выходы этих регистров управляются подсистемой условных переходов.

3.4 Подсистема условных переходов

Подсистема условных переходов состоит из двух частей:

1) первая часть реализует операцию сравнения операдов, лежащих в регистрах R2-R5

2) вторая часть загружает необходимый адрес команды в счетчик команд.

Данная система поддерживает два вида условных переходов и один безусловный.

При реализации подсистемы условных переходов адрес команды, к которой необходимо перейти, сохраняется в регистр JR. Управляющий сигнал передает значение регистра JR на шину, затем адрес сохраняется в счетчик команд.

Есть два условия: условие «больше» и условие «больше или равно».

Нас интересуют условие цикла «если i больше или равно n» и условие «строго больше», относящееся к элементам массива.

Если значение (i-1)-го элемента массива больше значения i-го элемента, то срабатывает флаг, и на шину с помощью управляемого буфера поступает адрес, по которому нужно перейти, и записывается в счетчик команд.

В зависимости от срабатывания флага на шину выпускается либо значение, либо адрес по которому нужно перейти в случае не выполнения условия, либо адрес по которому нужно перейти, если условие выполняется. При не выполнении условия, срабатывает инвертор и на шину выпускается текущее значение счетчика команд. Инвертированное значение и управляющий сигнал дают в логическом И 1 и включают управляемый буфер для регистра, который хранит текущее значение счётчика. Если не реализовывать данную логику, то может потеряться значение счетчика команд.

Условие «больше или равно» - второй вариант условного перехода. Проверка на прекращение цикла. Если использовать для сравнения элементов условие «больше или равно», то цикл может превратиться в бесконечный в случае, если в элемента равны и постоянно меняются местами.

3.5 Декодер

Декодер реализован с помощью командного регистра, постоянного запоминающего устройства, счетчика шагов, управляющих сигналов и системы обработки номера регистра.

Разрядность адреса: 7 бит. Старшие 4 бита представляют собой код команды и подаются на вход декодера из командного регистра. Младшие три бита формируются из значения в счетчике шагов. Разрядность данных декодера: 18 бит. С помощью декодера осуществляется подача 18 управляющих сигналов: вывод результата, вычитание, выход счетчика команд, включение счетчика команд, вход командного регистра, выход командного регистра, вход адресного регистра, чтение из памяти, запись в память, вход регистра Rn, выход регистра Rn, выход сумматора, вход регистра JR, выход регистра JR, флаг1, флаг2, вход регистра RTSK, включение счетчика команд.

Работа с регистром, указанным в коде команды, осуществляется с помощью декодера. Младшие четыре бита команды - значение, на которое с помощью декодера сдвигается единица влево. Выход декодера - управляющий сигнал на вход и выход значений регистра.

3.6 Организация ОЗУ

Для работы с памятью используются управляющие сигналы: вход адресного регистра, чтение из памяти, запись в память и очистка памяти. Подача данных с выхода ОЗУ на шину регулируется управляемым буфером.

- Программа сортировки хранится в ячейках памяти 0x00-0x75.
- Значение счетчика хранится по адресу 0xfa.
- Элементы массива хранятся в ячейках, начиная с 0x78 и заканчивая 0xf9, в зависимости от количества элементов в массиве.
- Количество элементов в сортируемом массиве лежит по адресу 0xff.

Для корректной работы программы пользователю необходимо вписать количество элементов массива в ячейку 0xff и удвоенное количство элементов массива в ячейку 0xfe. Также пользователю нужно вписать элементы массива в ячейки памяти, начиная с 0x78 и заканчивая 0xf9, в зависимости от количества элементов в массиве. Доступ к ячейкам памяти, в которых хранятся команды, осуществляется с помощью счетчика команд, значение которого записывается в адресный регистр.

4 Результат работы программы

Тестирование сортировки производилось на двух массивах разной размерности. Оба массива были отсортированы успешно.

Заключение

В результате данной научно-исследовательской работы был разработан специализированный процессор для сортировки массивов от 2 до 64 целочисленных 16-битных элементов алгоритмом сортировки вставкой. Были реализованы система команд и специализированное вычислительное устройство для разработанной системы команд. Также для реализованного процессора была написана программа, выполняющая сортировку массива.

Преимуществом данного процессора является разная длина кодов команд,что обеспечивает экономию оперативной памяти. Команды и данные хранятся в единой последовательно адресуемой памяти, что соответствует архитектуре фон Неймана, используется одна шина, на которую передаются и данные, и команды.

Программа, реализующая сортировку вставками, была протестирована на массивах разной длины. В результате тестирования система выполнила сортировку успешно во всех случаях.

Список литературы

[1] Цилькер, Б.Я.; Орлов, С.А. «Организация ЭВМ и систем».2004 г. – 654 стр.

[2] Logisim Документация
http://www.cburch.com/logisim/ru/docs.html