# ЛАБОРАТОРНАЯ РАБОТА №2

# ИССЛЕДОВАНИЕ АРХИТЕКТУРЫ И СИСТЕМЫ КОМАНД ВОСЬМИРАЗРЯДНОГО МИКРОПРОЦЕССОРА

Цель работы

Исследовать архитектуру и основные блоки 8-разрядного процессора. Исследовать взаимодействие основных блоков процессора при выполнении команд разных типов. Приобрести навыки написания и отладки ассемблерных программ в эмуляторе KP580 Emulator.

Задачи

1. Изучить архитектуру и основные команды микропроцессора КР580ВМ80;
2. Изучить возможности эмулятора и экранного отладчика kp580\_new. Исследовать изменение в основных блоках процессора в ходе выполнения команд различных типов;
3. Составить блок-схему алгоритма функционирования программы в соответствии с заданным вариантом;
4. Реализовать ассемблерную программу:
   1. Заполнить вручную N=10 ячеек памяти произвольными однобайтными натуральными числами;
   2. Осуществить сортировку по возрастанию созданного массива натуральных чисел, используя алгоритм сортировки подсчётом;
   3. Вычислить сумму элементов массива, расположенных на позициях, кратных трём;
   4. Найти минимальный и максимальный элементы массива;
5. Рассчитать длительность выполнения полученных программ в зависимости от используемых команд;
6. Сделать выводы по результатам проведенных исследований и расчётов;

Ход работы

Был открыт эмулятор микропроцессора KP580 Emulator, где были рассмотрены основные блоки процессора: АЛУ, устройство управление, регистры флагов и общего назначения, аккумулятор и др. (Рисунок 1).

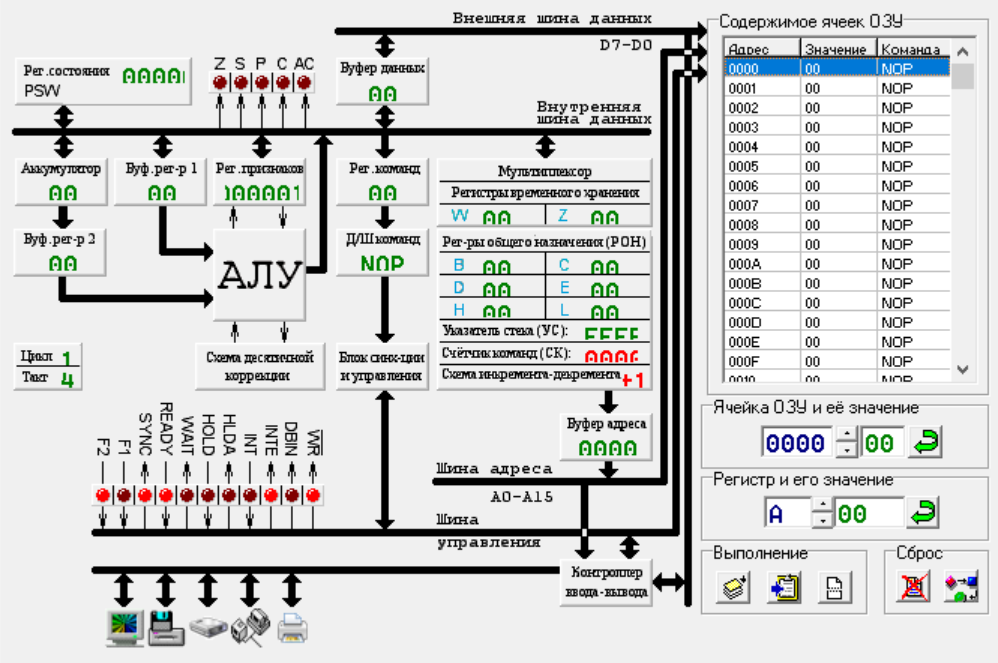


Рисунок 1 – Схема микропроцессора КР580ВМ80

Далее в эмуляторе для процессоров Intel 8080 был написан ассемблерный код, осуществляющий операции по варианту задания: запись массива в ячейки ОЗУ, сортировка массива методом подсчёта (дополнительные данные хранит через стек), подсчёт элементов с индексами, кратными трём, и поиск максимального и минимального элементов массива.

Был начат процесс ассемблирования программы. Сначала было произведено заполнение массива указанными в коде значениями (Рисунок 2).

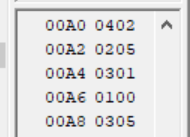


Рисунок 2 – Массив в ячейках ОЗУ

Далее была запущена сортировка массива. В результате значения в ячейках ОЗУ были перезаписаны в отсортированном порядке (Рисунок 3).

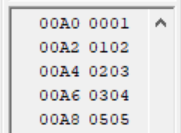


Рисунок 3 – Результат сортировки массива

Затем были запущены функции, отвечающие за подсчёт элементов с индексами, кратными 3, и поиск минимального и максимального элементов массива. В результате работы сумма была сохранена в регистре B, минимальный элемент в регистре E, максимальный – в D (Рисунок 4).

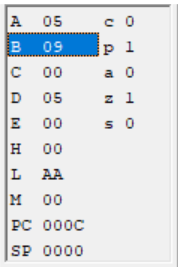


Рисунок 4 – Содержимое регистров в результате выполнения функций

Код программы

call fill\_array ; заполнение массива

call counting\_sort ; сортировка массива методом подсчёта

call sum\_of\_third ; подсчёт элементов, индексы которых кратны трём

call find\_min\_max ; поиск минимального и максимального элементов

hlt ; конец программы

fill\_array: ; функция заполнения массива

lxi h, 00A0h ; адрес начала массива в ОЗУ (в регистрах HL)

mvi m, 4 ; элемент 10

inx h

mvi m, 2 ; элемент 9

inx h

mvi m, 2 ; элемент 8

inx h

mvi m, 5 ; элемент 7

inx h

mvi m, 3 ; элемент 6

inx h

mvi m, 1 ; элемент 5

inx h

mvi m, 1 ; элемент 4

inx h

mvi m, 0 ; элемент 3

inx h

mvi m, 3 ; элемент 2

inx h

mvi m, 5 ; элемент 1

inx h

ret

counting\_sort: ; функция сортировки массива методом подсчёта

mvi b, 5 ; подсчитываемый элемент массива (в регистре B)

iteration:

lxi h, 00A0h ; адрес начала массива в ОЗУ (в рестрах HL)

mvi c, 10 ; счётчик количества элементов массива (в регистре C)

mvi d, 0 ; счётчик подсчитанных элементов B (в регистре D)

loop:

mov a, m ; загрузка в акк-р A значения элемента массива

cmp b ; сравнение элемента с подсчитываемым значением B

jnz not\_equal ; проверка регистра Z

inr d ; если регистр Z=1 увеличиваем счётчик D

not\_equal: ; иначе его не трогаем

inx h ; переход к следующей ячейке массива

dcr c ; уменьшаем счётчик числа элементов массива C

jnz loop ; если C!=0 обрабатываем следующую ячейку

push d ; загрузка в стек количества подсчитанных элементов

dcr b ; установка нового подсчитываемого элемента (B-1)

jm stop ; если B<0 переходим к этапу заполнения массива

jmp iteration ; если B>=0 начинаем подсчёт нового элемента B

stop:

lxi h, 00A0h ; адрес начала массива в ОЗУ (в рестрах HL)

mvi b, 0 ; записываемое в массив значение (регистр B)

mvi c, 6 ; счётчик числа различных значений массива, k (регистр C)

fill\_sorted: ; заполнение массива элементами в отсортированном порядке

pop d ; загрузка из стека количества элементов B в массиве

enter\_el: ; запись в массив значения B (D-раз)

mov m, b ; запись нового значения в отсортированный массив

inx h ; переход к следующей ячейке массива

dcr d ; узнать сколько элементов B осталось записать

jnz enter\_el ; если осталось для записи >0 элементов продолжить запись

inr b ; новое записываемое значение (B+1)

dcr c ; уменьшаем число значений, которые осталось записать

jnz fill\_sorted ; если записаны не все значения, продолжить запись

ret ; конец сортировки

sum\_of\_third: ; подсчёт суммы элементов, индексы которых кратны трём)

lxi h, 009Fh ; одна ячейка до начала массива в ОЗУ (в регистрах HL)

mvi c, 0 ; счётчик количества элементов массива (в регистре C)

mvi b, 0 ; сумма элементов кратных трём (в регистре B)

sum\_loop:

inx h ; перейти на три ячейки вперёд

inx h

inx h

mov a, c ; увеличить счётчик C на 3 (через аккумулятор)

adi 3

mov c, a

cpi 10 ; сравнить значение аккумулятора с 10 (проверка конца массива)

jc less\_or\_eq ; проверка A<10

jz less\_or\_eq ; проверка A=10

ret ; если счётчик элементов массива A>10 - конец подсчёта (ответ в B)

less\_or\_eq:

mov a, b ; прибавить значение ячейки к регистру для суммы B (через акк.)

add m

mov b, a

jmp sum\_loop ; перейти ещё на три элемента вперёд

find\_min\_max:

lxi h, 00A0h ; одна ячейка до начала массива в ОЗУ (в регистрах HL)

mvi c, 10 ; счётчик количества элементов массива (в регистре C)

mvi d, 0 ; регистр для максимального элемента (D)

mov e, m ; регистр для минимального элемента (E)

find\_max: ; сравнение с максимальным найденным элементом

mov a, m ; загрузка в акк-р значения ячейки массива

cmp d ; сравнение значения акк-ра со значением регистра D

jc find\_min ; если A<=D ничего не менять

mov d, a ; иначе перезаписать D значением акк-ра

find\_min: ; сравнение с минимальным найденным элементом

cmp e ; сравнение значения акк-ра со значением регистра E

jnc next\_cell ; если A>D ничего не менять

mov e, a ; иначе перезаписать E значением акк-ра

next\_cell:

inx h ; переход к следующей ячейке массива в ОЗУ

dcr c ; уменьшение счётчика элементов массива

jnz find\_max ; если счётчик не на нуле продолжить поиски

ret

Вывод

В ходе работы были изучены основы архитектуры ПК, архитектура и основные блоки 8-разрядных процессоров. Было исследовано взаимодействие блоков процессора при выполнении им команд разных типов, а также его связь с другими компонентами компьютера.

Также было проведено ознакомление с основами программирования микроэлектроники на ассемблере. Полученные знания были продемонстрированы при написании ассемблерной программы на эмуляторе процессора КР580ВМ80 (аналог Intel 8080).