МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ

БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет автоматики и вычислительной техники

Кафедра радиоэлектронных средств

Отчет по дисциплине

«Цифровые устройства и микропроцессоры»

Лабораторная работа №2

«ПРИНЦИПЫ ВЫПОЛНЕНИЯ КОМАНД ВЕТВЛЕНИЯ, ОРГАНИЗАЦИЯ ЦИКЛОВ И ПОДПРОГРАММ»

Вариант №11

|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. ИНБс-3301 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Д.Л. Железнов |
| Проверил: преподаватель кафедры ФПМТ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | М.А. Земцов |

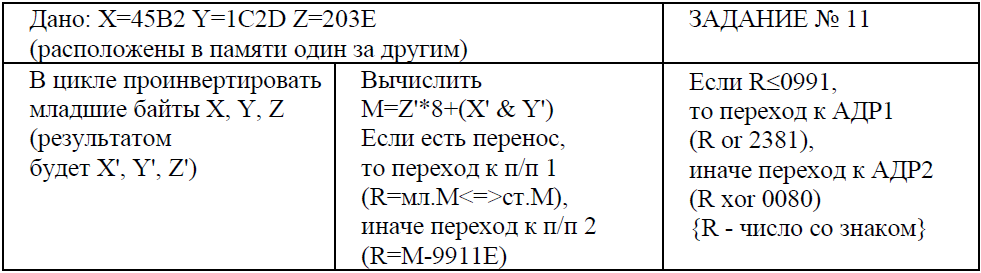
Киров 2022

**Цель работы:** изучение принципов выполнения команд ветвления, организации циклов и подпрограмм микропроцессоров с архитектурой x86.

**Ход работы:**

Исходные данные представлены в таблице 1.

Таблица 1 – Исходные данные



**Текст программы представлен ниже:**

.686

.model flat,stdcall

.stack 100h

.data

arr dw 045B2h,01C2Dh,0203Eh ; массив исходных данных

m dw 0000000011111111b ; обьявление маски для инверсии младших байтов у X Y Z

.code

ExitProcess PROTO STDCALL :DWORD

Start:

xor eax,eax ;очистка регистра EAX

xor ebx,ebx ;очистка регистра EBX

xor ecx,ecx ;очистка регистра ECX

xor edx,edx ;очистка регистра EDX

lea esi,[arr]

lea edi,[arr]

mov ecx,3 ; указываем в регистре ECX количество итераций

mov bx,m ; записал маску в bx в регистр EBX

@cycle: ; начало цикла

lodsw ; прочитал число

xor ax,bx ; XORом число на маску, которая находится в bx

stosw ; записал число

loop @cycle ; конец цикла

lea esi,[arr]

mov cx,ax ; значение в ax в cx

lodsw ; прочитал число

mov bx,ax ; значение в регистре ax в регистр bx

lodsw ; прочитал число

mov dx, ax

mov ax, bx

mov bx, dx

xor edx, edx

and ax, bx

mov dx, ax

mov ax, cx

mov cx, dx

mov ebx, 8h; если bx то с переносом

mul ebx

add eax,ecx

;stc установка флага CY - переноса

jc pp1; флаг переноса равен 1

jnc pp2; флаг переноса равен 0

pp1:

ror eax, 16

jmp ifc

pp2:

mov ebx, 9911Eh

sub eax, ebx

jmp ifc

ifc:

mov ebx, 0991h

cmp eax, ebx

jl adr1

jg adr2

je adr1

adr1:

or eax, 2381h

jmp exit

adr2:

xor eax, 0080h

exit:

Invoke ExitProcess,1

End Start

**Расчет вручную:**

Для инверсии младших байт умножим число на маску 0000 0000 1111 1111

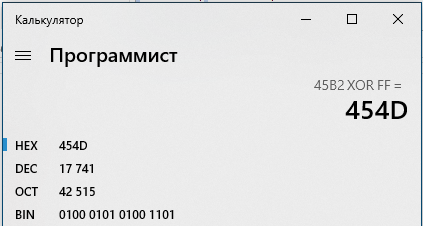


Рисунок 1 – Инверсия младшего байта у первого числа

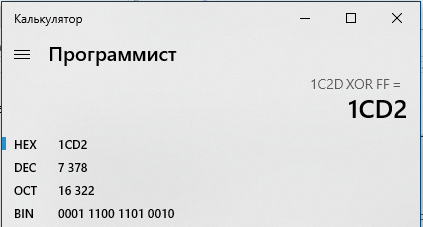


Рисунок 2 – Инверсия младшего байта у второго числа

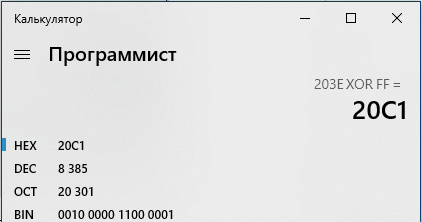


Рисунок 3 – Инверсия младшего байта у третьего числа

Z’\*8



X’&Y’



M=Z’\*8 + (X’&Y’)



Чтобы определить наличие переноса, необходимо проверить, есть ли единица у флага CF. Так как CF=0=> идем по метке п/п 2:

R=M-9911E

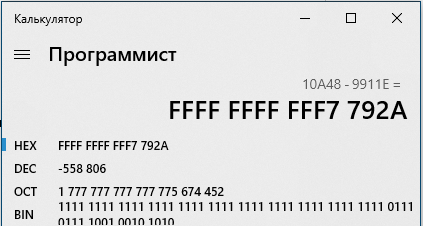


Рисунок 4 – Выполнение п/п 2

Сравниваем полученное число с числом 0991. Если оно меньше или равно, то переходим к АДР1 иначе к АДР2. Полученное число меньше 0991 и, следовательно, переходим к АДР1.

R or 2381

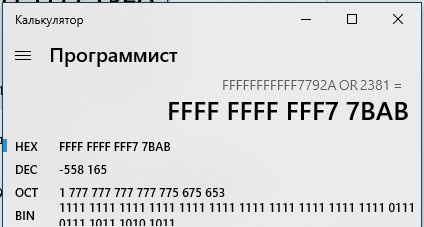


Рисунок 5 – Выполнение АДР1

**Расчет в программе:**

Результаты инверсии младших байт в программе:







Подсчет формулы:

Z’\*8



X’&Y’



M=Z’\*8 + (X’&Y’)



Двигаемся по ветке п/п 2:



Переходим к АДР1(так как число меньше)



**Верификация программы**

Для проверки п/п 1 установим флаг переноса командой stc (set carry flag) и проверим, как происходит обмен байтами, т.е. есть число 0001 0A48, то в результате должно быть 0A48 0001.



После этого программа должна попасть в ветку АДР2 (так как число больше) и найти значение R xor 0080:

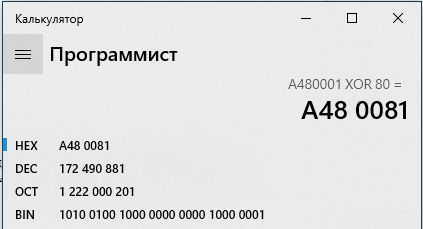
 

Рисунок 6 – Переход к АДР2 в калькуляторе и в программе

**Вывод:** в ходе данной лабораторной работы познакомились с циклами, ветвлениями, и их способом реализации на языке Assembler, а также с возможностью сравнивать числа со знаком и производить обмен байтами.