ГУАП

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ПРЕПОДАВАТЕЛЬ

| Ассистент |  |  |  | Д.О.Шевяков |
| --- | --- | --- | --- | --- |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

| | ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ | | --- | | **УСЛОВНЫЕ И БЕЗУСЛОВНЫЕ ПЕРЕХОДЫ**  Вариант 5 | |  | | по курсу: АРХИТЕКТУРА ЭВМ | |
| --- | --- | --- | --- | --- |

РАБОТУ ВЫПОЛНИЛ

| СТУДЕНТ ГР. № | 4128 |  |  |  | В. А. Воробьев |
| --- | --- | --- | --- | --- | --- |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

**СОДЕРЖАНИЕ**

[**1 Цель и постановка задачи** 3](#_3znysh7)

[**1.1 Цель работы** 3](#_2et92p0)

[**1.2 Задания** 3](#_tyjcwt)

[**2 Ход и результаты выполнения работы** 4](#_3dy6vkm)

[**2.1 Исходные данные** 4](#_1t3h5sf)

[**2.2 Листинг программы** 5](#_4d34og8)

[**3 Вывод** 12](#_2s8eyo1)

**1 Цель и постановка задачи**

**1.1 Цель работы**

Изучение архитектуры МП Intel 8086, изучение структуры простейшей ассемблерной программы, ознакомление с системой арифметико-логических команд процессора, организация вычисления на языке ассемблера.

**1.2 Задания**

1. Для всех заданий исходное число (числа) хранится в двухбайтовой ячейке (ячейках) сегмента данных, результат необходимо сохранить в однобайтовую ячейку сегмента данных. Под словосочетанием «сохранить результат» понимается запись результата в однобайтовую ячейку в сегменте данных. Во всех заданиях следует использовать только итерационные циклы и условные операторы.

2. Подсчитать количество десятичных цифр в числе

**2 Ход и результаты выполнения работы**

**2.1 Исходные данные**

Проверим работу программы на 3 различных числах:

**

Рисунок 1 – Алгоритм программы в соответствии с вариантом

**2.2 Листинг программы**

SStack segment 'stack'

DB 256 DUP (?)

SStack ends

SData segment 'data'

InputNum DW 0

DigitCount DD ?

DigitCountText DW "Digit Count = $"

SData ends

SCode segment 'code'

ASSUME CS:SCode, DS:SData, SS:SStack

PrintNum PROC NEAR

; print ASCII num

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV BX, 10

XOR CX, CX

OR AX, AX

JNS @@DIV:

NEG AX

PUSH AX

MOV AH, 02H

MOV DL, '-'

INT 21H

POP AX

@@DIV:

XOR DX, DX

DIV BX

PUSH DX

INC CX

OR AX,AX

JNZ @@DIV

MOV AH, 02H

@@STORE:

POP DX

ADD DL, '0'

INT 21H

LOOP @@STORE

POP DX

POP CX

POP BX

POP AX

; Print new line

PUSH AX

MOV AH, 0EH

MOV AL, 0AH

INT 10H

MOV AH, 0EH

MOV AL, 0DH

INT 10H

POP AX

RET

PrintNum ENDP

Main PROC FAR

MOV AX, SData

MOV DS, AX

MOV AX, InputNum

MOV CX, 0

CMP AX, 0

JGE GetDigits

NEG AX

GetDigits:

CountDigitsLoop:

MOV BX, 10

XOR DX, DX

DIV BX

INC CX

TEST AX, AX

JNZ CountDigitsLoop

MOV DigitCount, CX

MOV AX, DigitCount

LEA DX, DigitCountText

CALL PrintNum

MOV AH, 0

INT 21H

RET

Main ENDP

SCode ends

END Main

**2.3 Таблица трассировки программы**

Таблица 1 – Таблица трассировки программы

| **Исходные данные** | |
| --- | --- |
| X1 | 182 |
| X2 | 0 |
| X3 | -12 |
| **Операция** | **Результат** |
| X1 | 3 |
| X2 | 1 |
| X3 | 2 |

**2.4 Результат работы программы**

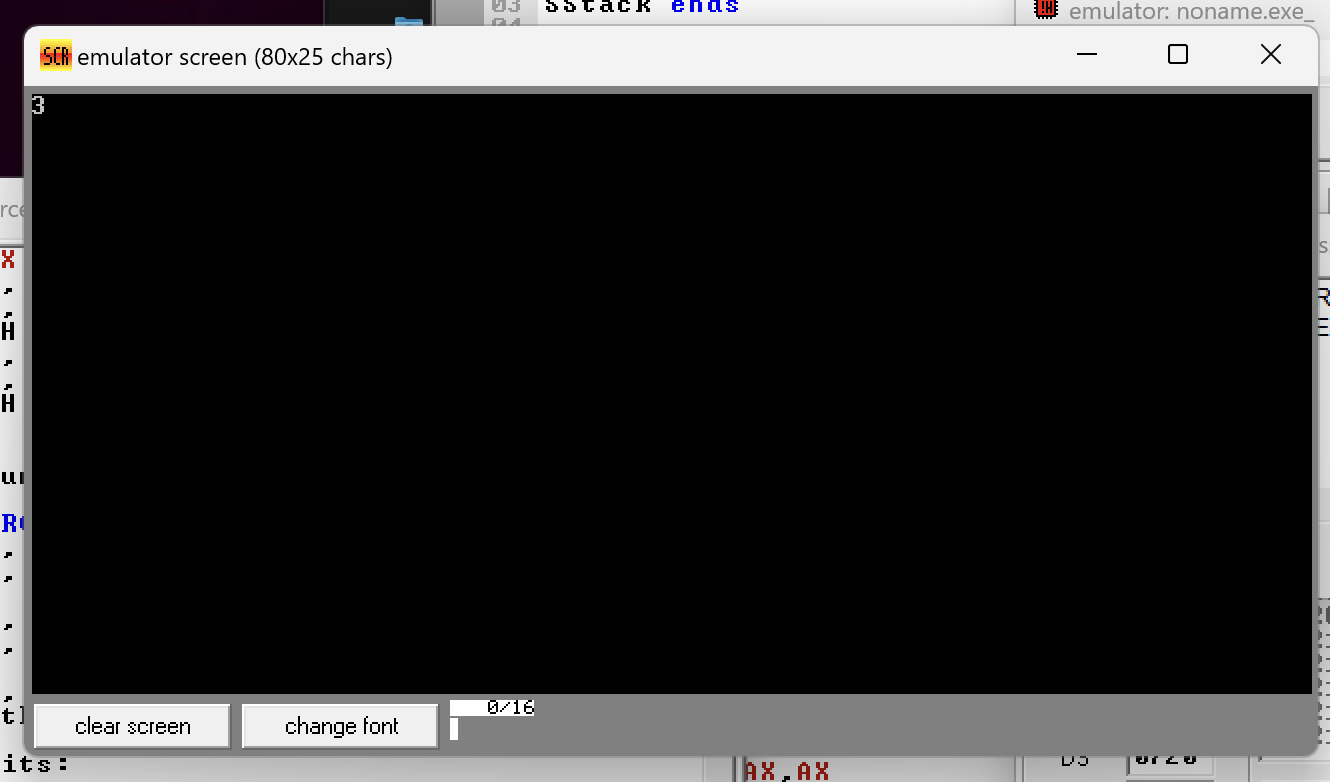
**

Рисунок 2 – Результат работы программы 1

**3 Вывод**

В ходе выполнения лабораторной работы были изучены: архитектура МП Intel 8086, структура простейшей ассемблерной программы, система арифметико-логических команда процессора и организация вычислений на языке ассемблера.

В результате был написан код на языке ассемблера Intel 8086 с помощью программы emu8086, выполняющий необходимые операции. Кроме того, дополнительно был реализован вывод результатов операций в консоль.