**Национальный исследовательский ядерный университет «МИФИ»**

|  |  |
| --- | --- |
|  | **Факультет Кибернетики и информационной безопасности**  **Кафедра «Компьютерные системы и технологии»** |

**ОТЧЕТ**

**по выполнению лабораторного практикума**

**по куру «Микропроцессорные устройства и системы»**

Студент гр. Б18-503 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Григорьев М.П. /

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Ёхин М.Н. /

**Лабораторная работа №3**

Вариант 122

Условие:

1) Сумма количества “1” в разрядах 0, 1, 5 байта “A”, количества “0” в разрядах 2, 4, 7 байта “B” и модуля разности адреса байтов “A” и “B”

2) Сдвиг влево на N разрядов, где N=(addr A) \* ( |addr B – 2|) с вводом “0”

Формат управляющего слова:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Адрес операнда А | | Адрес операнда B | | Код операции |
| P1.3 | P1.2 | P1.2 | P1.0 | P2.0 |

**Исходный код программы:**

ORG 0H ; Область программы

LJMP PROGRAM

ORG 03H ; Область памяти прерываний

LJMP INTER

PROGRAM:

P4 EQU 0C0h

MOV IE, #10000001b

SETB IT0 ; Устанавливаем флаг, что улавливаем все прерывания

OUTPUT:

MOV P4, A ; Результат в P4

SJMP OUTPUT

INTER:

MOV DPTR, #7FFAh

MOVX A, @DPTR

MOV 20H, A; Сохраняем ввод

MOV DPTR, #8000h

MOV R5, #0 ; Address A

JNB ACC.3, A1

INC DPTR

INC R5

A1:

JNB ACC.4, B0

INC DPTR

INC DPTR

INC R5

INC R5

B0:

MOVX A, @DPTR

MOV R0, A; R0 - A

MOV A, 20H

MOV DPTR, #8004h

MOV R6, #0 ; Address B

JNB ACC.1, B1

INC DPTR

INC R6

B1:

JNB ACC.2, MOV\_B

INC DPTR

INC DPTR

INC R6

INC R6

MOV\_B:

MOVX A, @DPTR

MOV R1, A; R1 - B

MOV A, 20H

JB ACC.0, LOG1; Analyze operation A0 == Logic

JNB ACC.0, ARIFM

LOG1:

LJMP LOGIC

ARIFM:

MOV A, #00h

MOV R2, A; 0 TO COUNT 1 IN A (R0)

MOV R3, A; 0 TO COUNT 0 IN B (R1)

MOV A, R0; COUNT 1 IN A

JNB ACC.0, COUNT\_1\_1

INC R2 ; COUNT A0

COUNT\_1\_1:

JNB ACC.1, COUNT\_1\_5

INC R2 ; COUNT A1

COUNT\_1\_5:

JNB ACC.5, COUNT\_0

INC R2 ; COUNT A5

COUNT\_0:

MOV A, R1; COUNT ZEROS IN B

JB ACC.2, COUNT\_0\_2

INC R3

COUNT\_0\_2:

JB ACC.4, COUNT\_0\_7

INC R3

COUNT\_0\_7:

JB ACC.7, ADDRESSES

INC R3

ADDRESSES:

CLR C

MOV A, R6 ; B Adress

SUBB A, R5; Sub A Adress | B-A

JNC ARIF\_RES

CLR C

MOV A, R5 ; A Adress

SUBB A, R6; Sub B Adress | A-B

ARIF\_RES:

ADD A, R2

ADD A, R3

LJMP OP\_END

LOGIC:

CLR C

MOV A, R6 ; addr B -> reg A

SUBB A, #2h ; Sub 2 -> addr B - 2

JNC LOGIC\_CYCLE

CLR C

MOV A, #2h ; #2 -> reg A

SUBB A, R6 ; sub B -> 2 - addr B

LOGIC\_CYCLE:

MOV B, R5 ; A -> reg B

MUL AB ; addr A \* (addr B-2)

MOV R5, A

MOV A, R0 ; A -> Acc

CYCLE:

CJNE R5, #00h, LEFT\_SHIFT

AJMP OP\_END

LEFT\_SHIFT:

CLR C

RLC A

DEC R5

AJMP CYCLE

OP\_END:

RETI

END

**Пример заполнения памяти:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7FFAh | … | 8000 | 8001 | 8002 | 8003 | 8004 | 8005 | 8006 | 8007 |
| 00 |  | 23 | 23 | 00 | FF | 23 | 23 | FF | 23 |

**Тестовые случаи:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Адрес A | Адрес B | A | B | Операция | Результат |
| 10 | 10 | 00000000 | 11111111 | Арифметическая | 00000000 |
| 10 | 01 | 00000000 | 00100011 | Арифметическая | 00000100 |
| 11 | 11 | 11111111 | 00100011 | Арифметическая | 00000110 |
| 00 | 11 | 00100011 | 00100011 | Арифметическая | 00001001 |
| 11 | 00 | 11111111 | 00100011 | Арифметическая | 00001001 |
| 11 | 10 | 11111111 | - | Логическая | 11111111 |
| 11 | 00 | 11111111 | - | Логическая | 11000000 |
| 11 | 11 | 11111111 | - | Логическая | 11111000 |
| 11 | 01 | 11111111 | - | Логическая | 11111000 |

**Результаты тестирования программы:**

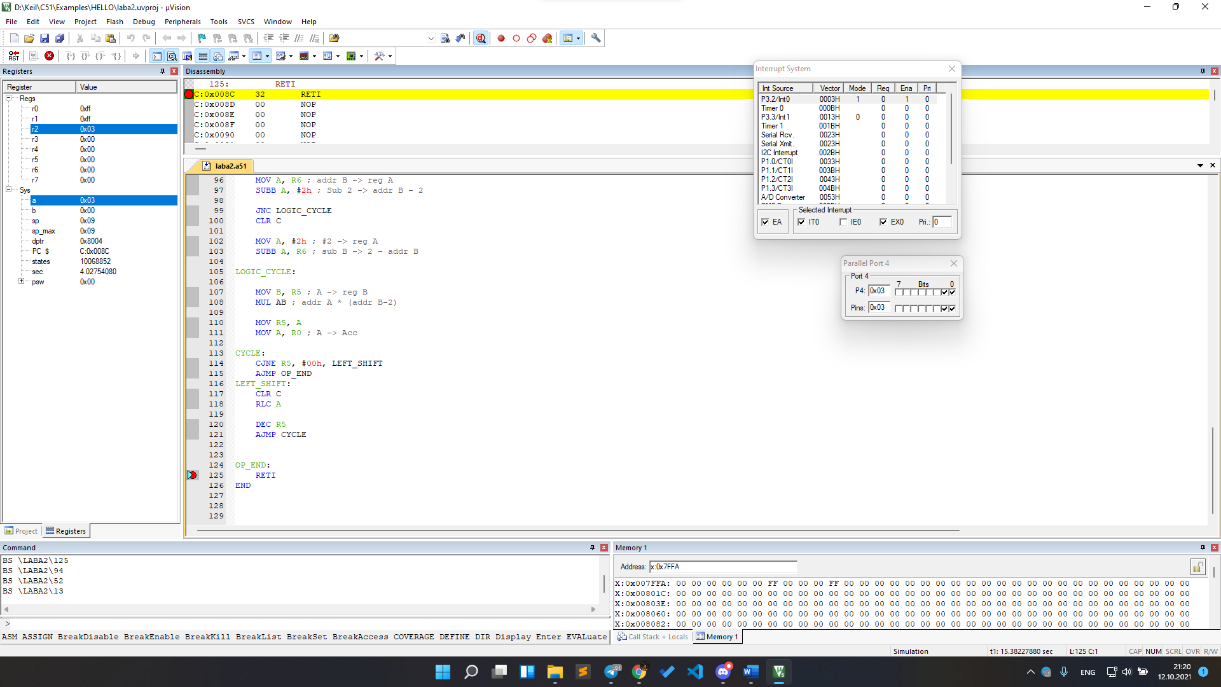


Рисунок 1 – Проверка арифметической операции 1

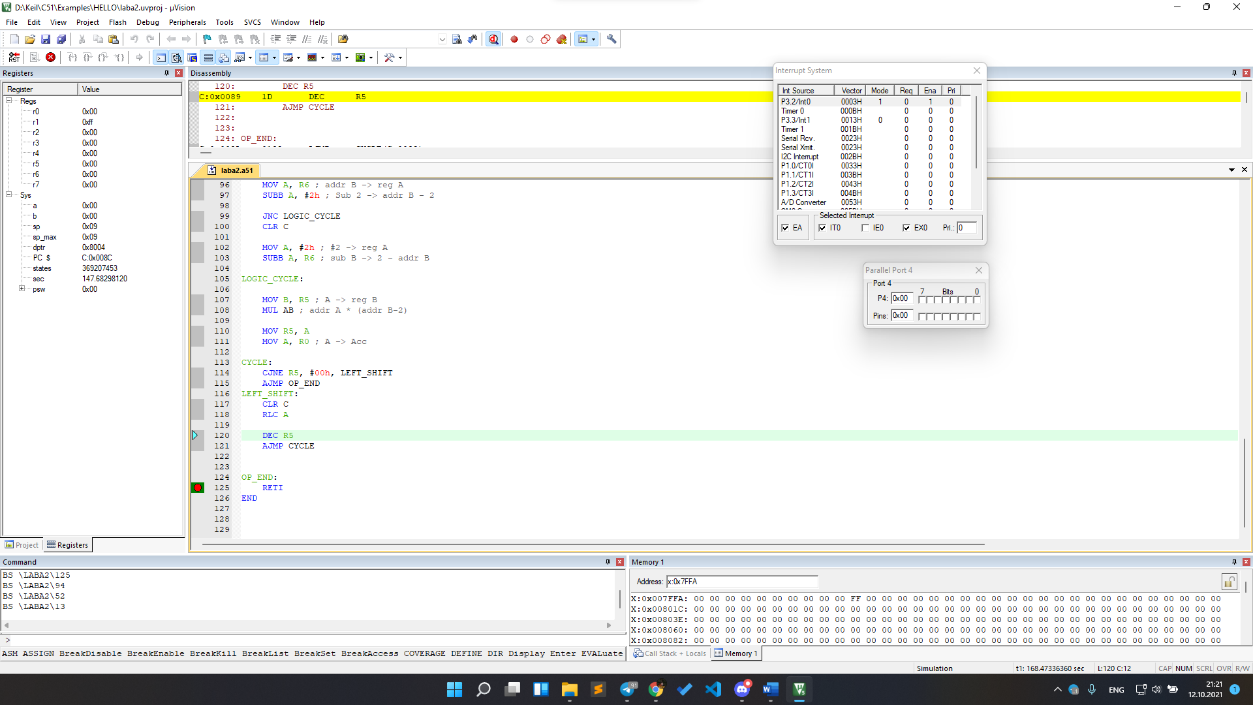


Рисунок 2 – Проверка арифметической операции 2

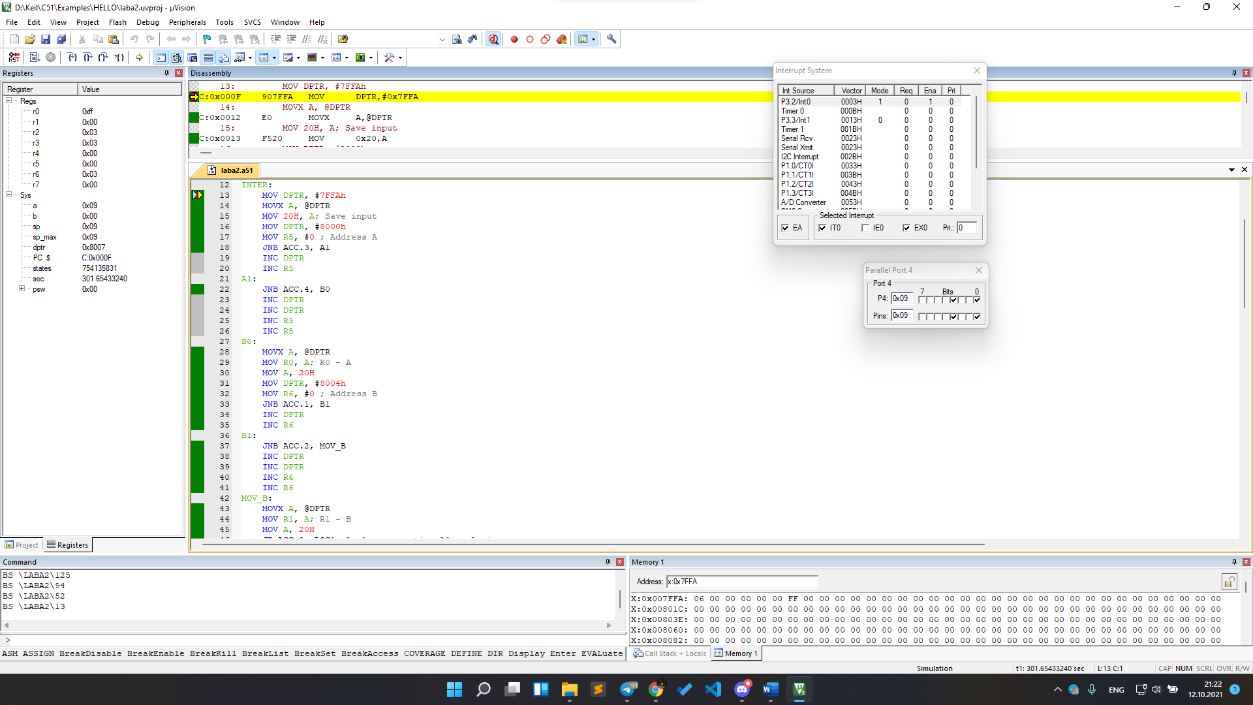


Рисунок 3 – Проверка арифметической операции 3

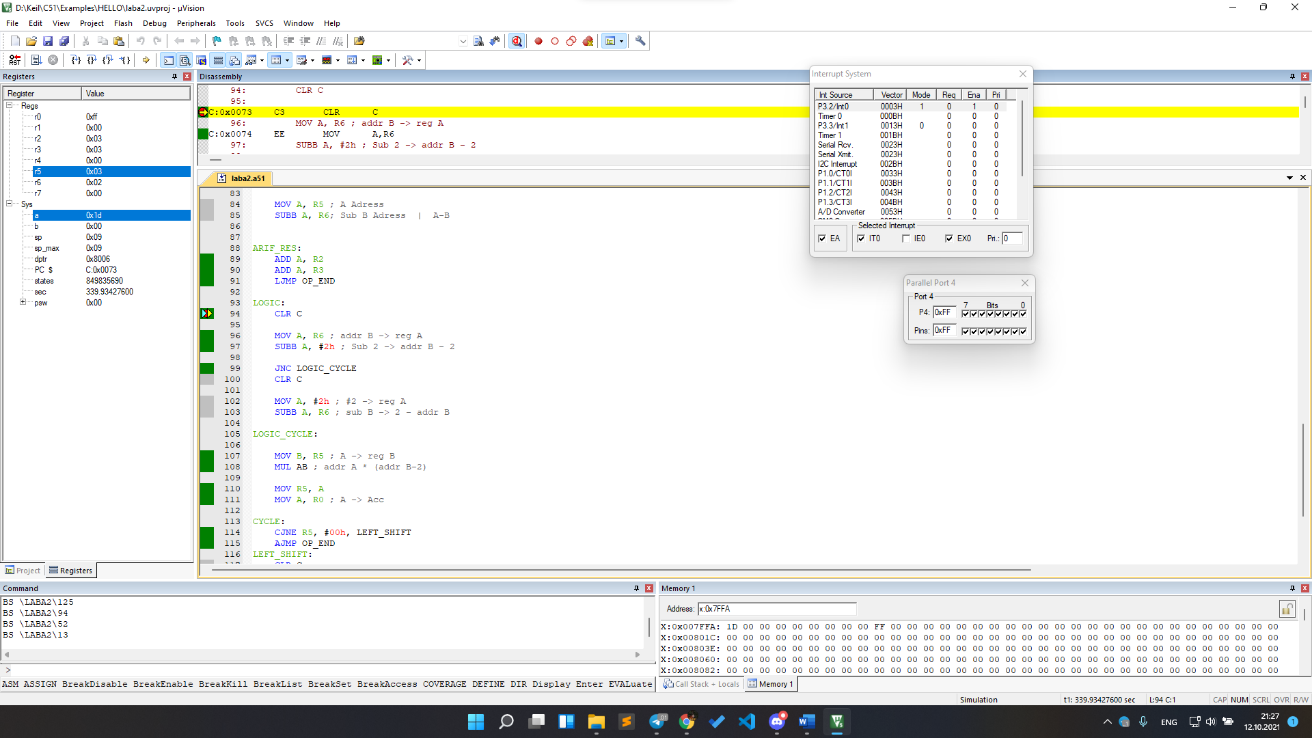


Рисунок 3 – Проверка логической операции 1

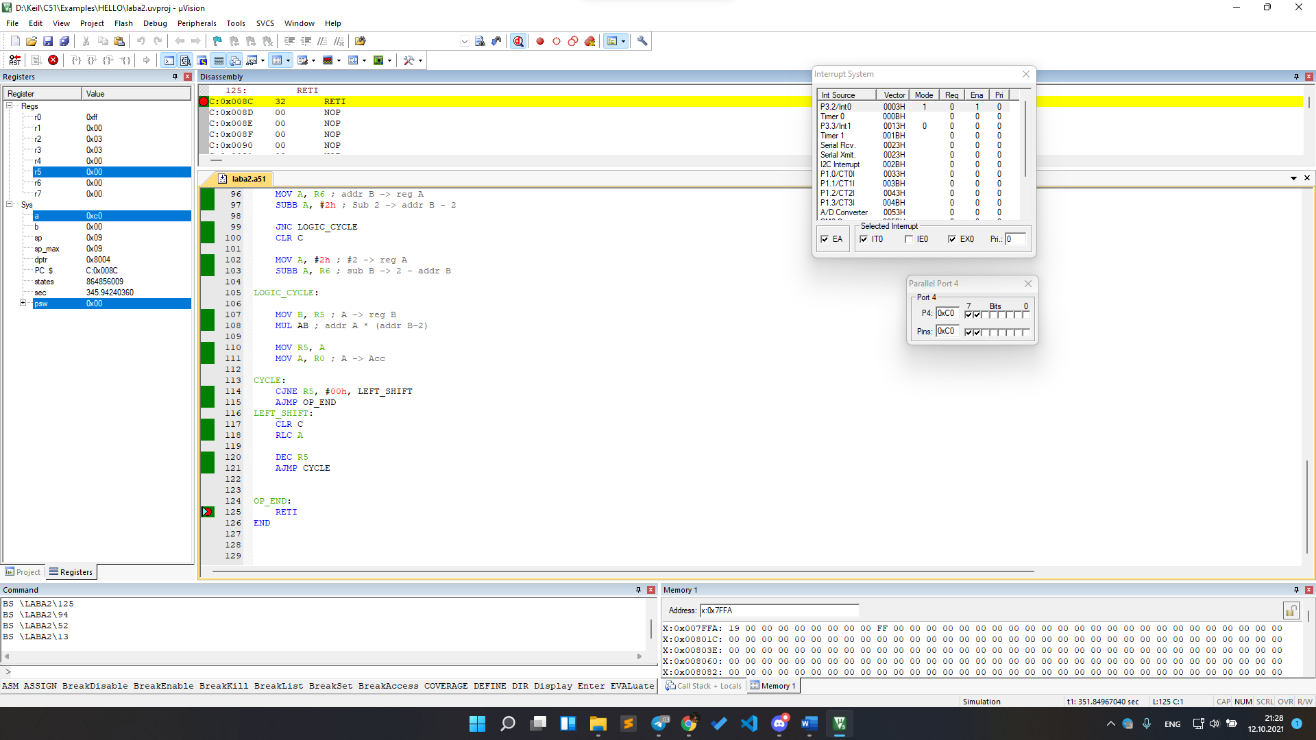


Рисунок 4 – Проверка логической операции 2

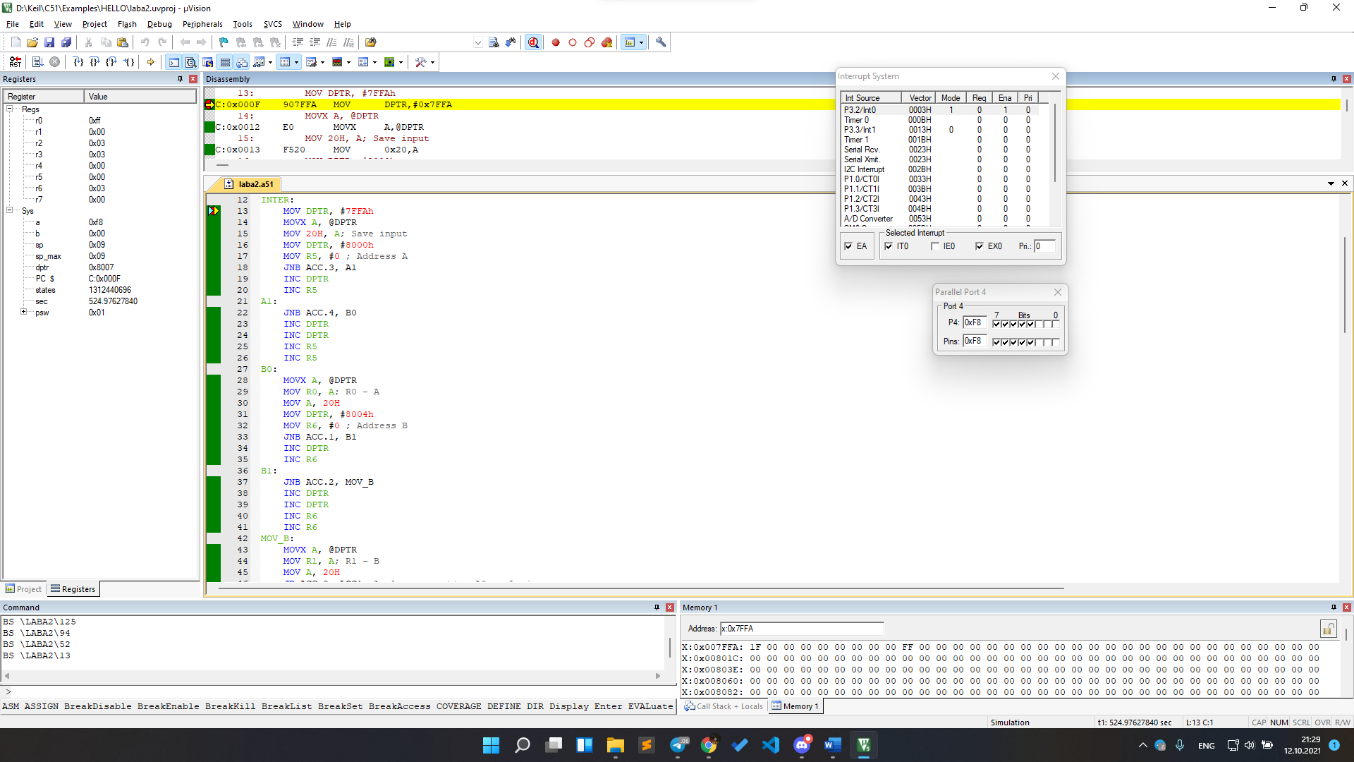


Рисунок 4 – Проверка логической операции 3