МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики
Кафедра вычислительных технологий

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №4

по дисциплине

«Системы реального времени»

Работу выполнил студент группы 45/2 _____ Т. Э. Айрапетов

Отчет принял

доц. каф. ИТ_____ А. Н. Полетайкин

Краснодар
2024

Тема: Программирование ввода-вывода в режиме реального времени

Цель: изучение принципов организации ввода информации извне в УВМ и вывода информации из УВМ вовне, организации временной задержки при обработке данных на языке ассемблера, а также приобретение практических навыков программирования указанных операций

Задание.

Вариант 1

Номер варианта	Адреса портов		Параметры УСРР, байт				Параметры синхронизации		
			старший		младший				
	A_{IN}	A_{OUT}	U_0	ΔU	SR	RS	τ, мс	R_C	N
1	501h	500h	01h	10h	1, 2	6, 7	145	15	15

Рисунок 1 - Заданные параметры СРВ

Имеется СРВ, включающая в себя некоторую аппаратную часть периферийных устройств (ПУ) и ядро в виде УВМ (рис. 2), которая осуществляет обмен с периферией через один 16-разрядный порт ввода с заданным адресом A_{IN} и один 16-разрядный порт вывода с заданным адресом A_{OUT}. Входные 8-разрядные данные поступают на младший байт порта ввода.

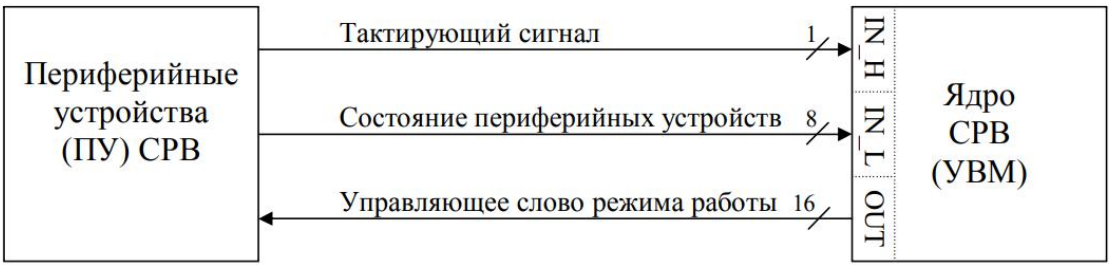


Рисунок 2 - Структурная схема СРВ

Для актуализации входных данных необходимо предварительно выводить через порт вывода заданное управляющее слово режима работы (УСРР). Отправление УСРР тактируется высоким уровнем сигнала через заданный разряд R_C порта ввода. Перед отправлением УСРР его необходимо модифицировать в 2 этапа:

1. Начальное значение старшего байта УСРР (УСРР[1]) – заданная комбинация U_0. При каждом следующем запросе данных значение УСРР[1] меняется на заданное приращение ΔU. Вследствие аппаратных временных затрат периферийной части актуализация данных происходит через заданный интервал времени τ

2. Младший байт УСРР (УСРР[0]) подвергается побитовой модификации следующим образом:

- на каждой нечётной итерации заданные биты SR устанавливаются, а биты RS сбрасываются;
- на каждой чётной итерации заданные биты SR сбрасываются, а биты RS устанавливаются.

Составить программу на языке ассемблера, которая осуществляет опрос внешних устройств через порт ввода и записывает отправленные УСРР и соответствующие им данные о состоянии периферийных устройств в одномерные массивы заданной длины N. Реализацию задержки времени осуществлять при помощи подпрограммы DELAY.

Выполнение.

Согласно номеру варианта, время задержки должно составлять 145 мс. Тогда, для процессора с тактовой частотой 3,1 ГГц количество тактов задержки подпрограммы DELAY должно быть равно 145 000 000 * 3,1 = 449 500 000. Формула для подсчета количества тактов подпрограммы DELAY (из методички):

$$T = 29+4+N*(4+M*K*3+(M-1)*17+5+3+16) -16+4+20 ,$$

где N - кол-во итераций внешнего цикла, M - кол-во итераций внутреннего цикла, K - кол-во NOP во внутреннем цикле.

Для подбора неизвестных в цикле по N от 1 до 5000 и в цикле по M от 1 до 5000 и в цикле по K от 1 до 20 высчитывалось значение T. Лучшими значениями оказались N = 3325, M = 3297, K = 8. Для них T принимает значение 449 500 141, соответственно задержка DELAY в миллисекундах равна 449 500 141/3,1/1 000 000 = 145,00005 мс.

На рисунке 3 представлен код подпрограммы DELAY.

```
DELAY:
    push ecx
    push ebx
    push esi

    MOV BX, 3325
    D1:
        MOV CX, 3297
        D2: NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        LOOP D2 ; 5 без перехода, 17 с переходом
        DEC BX ; 3 т.
    JNZ D1 ; 4 т. без перехода, 16 т. с переходом

    pop esi
    pop ebx
    pop ecx

    RET ; 20 т.
```

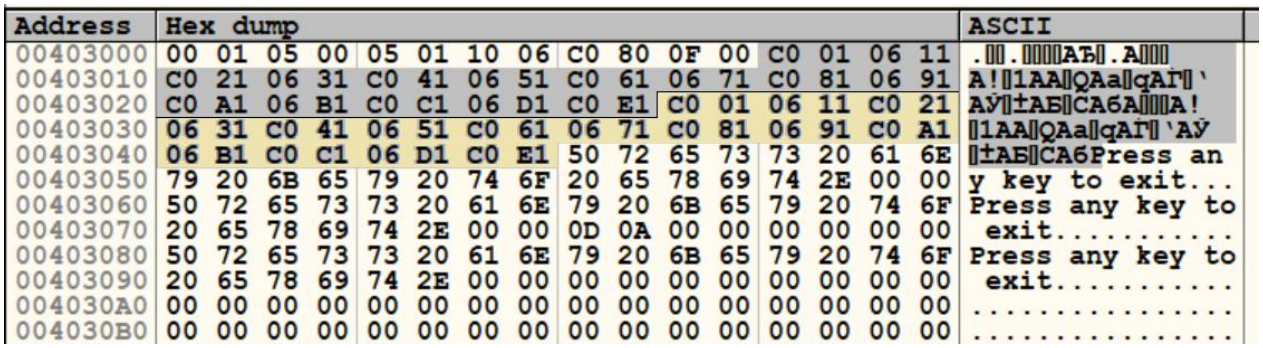
Рисунок 3 - Подпрограмма DELAY

```
.data
    Ain dw 501h
    Aout dw 500h
    U0 db 01h
    dU db 10h
    SR db 00000110b ; 1, 2 биты
    RS db 11000000b ; 6, 7 биты
    Rc db 10000000b
    N dw 0Fh

    arr_in dw 0Fh dup(?)
    arr_out dw 0Fh dup(?)
```

Рисунок 4 - Сегмент данных

```
.code
LStart:
    xor ecx, ecx
    xor esi, esi
    xor ebx, ebx
    xor eax, eax

    mov ah, U0
    mov al, RS
    mov cx, 0

    main:
        test_Rc:
            ;mov dx, Ain
            ;push ax
            ;in ax, dx
            mov bl, Rc
            test bl, Rc
            ;pop ax
            jz test_Rc

        test cx, 1; проверка на четность
        jz is_even ; четная итерация
        jnz is_odd ; нечетная итерация

        is_even:
            mov bl, SR
            xor bl, 0FFh
            and al, bl
            or al, RS
            jmp submain

        is_odd:
            mov bl, RS
            xor bl, 0FFh
            and al, bl
            or al, SR
            jmp submain

        submain:
            ;mov dx, Aout
            ;out dx, ax
            mov arr_out[esi], ax
            call DELAY
            ;mov dx, Ain
            ;push ax
            ;in ax, dx
            mov arr_in[esi], ax
            ;pop ax

            add esi, 2
            add ah, dU

            inc cx
            cmp cx, N
            jne main

    jmp LExit
```

Рисунок 5 - Код программы



Address	Hex dump																ASCII
00403000	00	01	05	00	05	01	10	06	C0	80	0F	00	C0	01	06	11	.▯▯.▯▯▯▯АЪ▯.А▯▯▯
00403010	C0	21	06	31	C0	41	06	51	C0	61	06	71	C0	81	06	91	А!▯1АА▯QAa▯qAЃ▯`
00403020	C0	A1	06	B1	C0	C1	06	D1	C0	E1	C0	01	06	11	C0	21	АЎ▯±АБ▯СА6А▯▯▯А!
00403030	06	31	C0	41	06	51	C0	61	06	71	C0	81	06	91	C0	A1	▯1АА▯QAa▯qAЃ▯`АЎ
00403040	06	B1	C0	C1	06	D1	C0	E1	50	72	65	73	73	20	61	6E	▯±АБ▯СА6Press an
00403050	79	20	6B	65	79	20	74	6F	20	65	78	69	74	2E	00	00	y key to exit...
00403060	50	72	65	73	73	20	61	6E	79	20	6B	65	79	20	74	6F	Press any key to
00403070	20	65	78	69	74	2E	00	00	0D	0A	00	00	00	00	00	00	exit.....
00403080	50	72	65	73	73	20	61	6E	79	20	6B	65	79	20	74	6F	Press any key to
00403090	20	65	78	69	74	2E	00	00	00	00	00	00	00	00	00	00	exit.....
004030A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
004030B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....

Рисунок 6 - Массивы входных и выходных значений в дампе памяти после выполнения программы

Выводы. В ходе выполнения работы были изучены принципы организации ввода информации извне в УВМ и вывода информации из УВМ вовне, организации временной задержки при обработке данных на языке ассемблера, а также приобретены практические навыки программирования указанных операций.