МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Кафедра информационных технологий**

**Отчет о выполнении лабораторной работы №4**

**по дисциплине «Системы реального времени»**

Работу выполнил студент 45/2 группы А.А. Козин

Руководитель

доц. каф. ИТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.Н. Полетайкин

Краснодар

2023

**ЛАБОРАТОРНАЯ РАБОТА 4**

**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 программирование указанных операций.

Задание.

Имеется СРВ, включающая в себя некоторую аппаратную часть периферийных устройств (ПУ) и ядро в виде УВМ (рис. 4.2), которая осуществляет обмен с периферией через один 16-разрядный порт ввода с заданным адресом АIN и один 16-разрядный порт вывода с заданным адресом АOUT. Входные 8-разрядные данные поступают на младший байт порта ввода. 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 RC порта ввода. Начальное значение старшего байта УСРР (УСРР[1]) – заданная комбинация U0. При каждом следующем запросе данных значение УСРР[1] меняется на заданное приращение U. Вследствие аппаратных временных затрат периферийной части актуализация данных происходит через заданный интервал времени .

A close up of text

Description automatically generated

Рисунок 1 – Структурная схема СРВ

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. Блок-схема алгоритма управляющей программы показана на рис. 4.3. Заданные параметры СРВ представлены в таблице.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номер  варианта | Адреса портов | | Параметры УСРР, байт | | | | Параметры синхронизации | | |
| старший | | младший | | , мс | RC | N |
| AIN | AOUT | U0 | U | SR | RS |
| 24 | 07Fh | 080h | 40h | 11h | 6,7 | 0 | 44,8 | 10 | 11 |

**ХОД РАБОТЫ**

1. Определим более подробно поставленную задачу. Пусть на вход получаем код AIN, представленный 16-битным словом. Старший байт этого слова, обозначенный как RC (в данном случае равен 10), указывает на необходимость обработки сообщения. Младший байт содержит информацию о состоянии определенных кнопок, где 1 означает включенное состояние, а 0 – выключенное. Для удобства мы будем сохранять информацию парными элементами: в каждой успешной итерации первый массив будет содержать модифицированный RC, а соответствующая позиция второго массива будет хранить полезную информацию о состоянии кнопок.

Код программы:

%include "io.inc"

section .data

USRRS dw 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

AINS db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

RC db 10

AIN dw 0000001000101001b

USRR dw 0001110000000100b

Udelta db 11h

SR\_MASK db 11000000b

SR\_INV\_MASK db 00111111b

RS\_MASK db 00000001b

RS\_INV\_MASK db 11111110b

section .text

global CMAIN

; Подпрограмма для проверки необходимости обработки итерации

check\_rc\_success:

mov dx, [AIN]

test bx, 1 ; Проверка бита четности итерации

jne is\_odd ; Переход к обработке нечетного случая

jmp is\_even ; Переход к обработке четного случая

; Подпрограмма для обработки нечетного случая

is\_odd:

or al, [SR\_MASK] ; Устанавливаем биты SR

and al, [RS\_INV\_MASK] ;Сбрасываем биты RS

call save\_usrrt

ret

; Подпрограмма для обработки четного случая

is\_even:

or al, [RS\_MASK] ; Устанавливаем биты RS

and al, [SR\_INV\_MASK] ; Сбрасываем биты SR

call save\_usrrt

ret

; Сохранение модифицированного значения УСРР и вызов задержки

save\_usrrt:

mov [esi], ax ; Сохранение УСРР в массив

;out 080h, ax ; Вывод УСРР на устройство вывода

call delay ; Вызов задержки

ret

; Функция задержки

delay:

mov ECX, 44800 ; Установка счетчика цикла задержки 44,8 мс

start\_loop:

loop start\_loop ; Выполнение цикла задержки

; Сохранение информации о кнопках A\_in

save\_ain:

mov [edi], dl

; Увеличение старшего байта УСРР

incUSRR:

add ah, [Udelta] ; Добавление к старшему байту УСРР значения Udelta

ret

; Основная функция программы

CMAIN:

mov ebp, esp ; Установка базового указателя для отладки

mov ax, [USRR] ; Загрузка начального значения УСРР

mov dx, [AIN] ; Загрузка начального значения A\_in

mov bx, 11 ; Установка размера обрабатываемого массива

mov esi, USRRS ; Указатель на начало массива USRRS

mov edi, AINS ; Указатель на начало массива AINS

; Основной цикл программы

main\_loop:

;in dx, 07Fh ; Чтение данных с порта

mov dx, [AIN]

inc dx ; Инкремент A\_in

mov [AIN], dx

mov cl, [RC]

shr dx, cl ; Сдвиг A\_in на количество битов, указанное в RC

call check\_rc\_success ; Вызов подпрограммы проверки

add esi, 2 ; Переход к следующему элементу в массиве USRRS

add edi, 1 ; Переход к следующему элементу в массиве AINS

dec bx ; Уменьшение счетчика цикла

cmp bx, 0

je main\_loop\_end ; Завершение цикла, если счетчик равен нулю

jmp main\_loop ; Повтор цикла

; Завершение программы

main\_loop\_end:

xor eax, eax

xor ebx, ebx

int 0x80 ; Системный вызов для завершения программы

2. Для проверки функциональности программы мы проведем имитацию взаимодействия с внешним устройством, которое генерирует уникальную (не постоянную) последовательность состояний кнопок для обработки. После компиляции программы мы ее запустим и получим следующие результаты:

A number of numbers in a row

Description automatically generated with medium confidence

Рис. 1 – память на момент начала работы программы.

A group of numbers and letters

Description automatically generated

Рис. 2 – память на момент окончания работы программы.

**РЕЗУЛЬТАТЫ**

В первом этапе был разработан ассемблерный код для управления взаимодействием между ПУ и центральным процессором, включая функцию задержки.

На втором этапе программа была собрана и испытана через симуляцию общения, что позволило продемонстрировать её функционирование как было задумано в первом этапе.

**ВЫВОДЫ**

Изучены и применены методы ввода-вывода данных между ПУ и центральным процессором с использованием ассемблерного кода, а также освоены техники программирования задержек обработки данных. Это дало практический опыт в реализации ввода-вывода и задержки в ассемблерных программах.