Лабораторная работа №5

«Исследование методов ввода-вывода данных в персональных компьютерах»

5.1 Цель работы:

Изучить способы функционирования клавиатуры и подключения ее к процессору, принципы отображения цифровой информации в жидкокристаллических дисплеях, методы программирования ввода-вывода данных. Исследовать особенности функционирования микропроцессора при реализации ассемблерных функций ввода данных с клавиатуры и вывода их на экран монитора. Приобрести практические навыки программирования на языке ассемблера МП 8086 процедур ввода-вывода с использованием функций BIOS.

5.2 Постановка задачи

Вариант – 8

Изучить принцип устройства компьютерной клавиатуры и кодирования формируемых символов, а также основные функции BIOS, позволяющие обрабатывать состояния клавиатуры. Изучить принцип устройства жидкокристаллических мониторов и управления пикселами. Изучить основные функции BIOS, позволяющие упрощать программировать задачи работы с клавиатурой и дисплеем.

Запустить в отладчике emu8086 программу вывода на экран VGAмонитора прямоугольника (emu8086\examples\0_sample_vga_graphics.asm) и исследовать работу процессора при выполнении этой программы. Составить подробный алгоритм работы этой программы.

Модифицировать приведенную в примере программу, позволяющей а) изменять размер отображаемого прямоугольника; б) изменение цвета фигуры.

Произвести отладку разработанных программ в пошаговом режиме и проследить за изменениями содержимого регистров.

5.3 Ход работы

Был изучен принцип устройства компьютерной клавиатуры и кодирования формируемых символов, а также основные функции BIOS, позволяющие обрабатывать состояния клавиатуры. Изучен принцип устройства жидкокристаллических мониторов и управления пикселами. Изучены основные функции BIOS, позволяющие упрощать программировать задачи работы с клавиатурой и дисплеем.

Была составлена программа согласно заданию представленная в листинге 1. Данная программа на ассемблере рисует прямоугольник в графическом режиме (видеорежим 13h, разрешение 320x200). Пользователь вводит координаты второй точки прямоугольника (x2 и y2). Цвет прямоугольника указан в переменной color. В коде есть два цикла, в каждом из которых вызывается прерывание 10h для установки точек прямоугольника. В первом цикле координаты увеличиваются, во втором - уменьшаются. В конце программы есть вызов прерывания 21h с кодом 1, чтобы ждать нажатия клавиши перед выходом из программы. В программе также используется процедура InputInt для ввода числовых значений с клавиатуры.

Листинг 1 – Программа обработки строк

```
.data
chr db 'F'
x1 dw 10  ; col
y1 dw 10  ; row
x2 dw 50
y2 dw 20
color db 6

.code
begin:
    mov ax, @data
    mov ds, ax
    mov es, ax

    call InputInt
    mov x2, ax
    call InputInt
    mov y2, ax

    mov ah, 0       ; 0 - установить видеорежим
    mov al, 13h     ; Видеорежим = 13h (графика, 320x200)
    int 10h         ; Прерывание.

    mov cx, x1      ; устанавливаем координату X
    mov dx, y1      ; устанавливаем координату Y
    mov ah, 0Ch     ; Номер функции установки точки
                    ; CX - строка (Y) ; DX - столбец (X)
    xor bh, bh      ; видеостраница - 0
    mov al, color   ; устанавливаем цвет

c1:
    int 10h         ; вызываем прерывание и ставим точку
    cmp dx, y2      ; сравниваем со значением y2
    jne lp          ; если не равно - goto LP
    cmp cx, x2      ; если равно - сравниваем с X2
    jne lp2         ; не равно - goto lp2
```

```
    jmp ex          ; иначе - выходим из цикла (т.к. половину прямоугольника мы нарисовали)
lp:
    inc dx          ; увеличиваем координату
    jmp c1
lp2:
    inc cx
    jmp c1

ex:
; аналогичный цикл на достроение 2 части прямоугольника
c2:
    int 10h
    cmp dx, y1
    jne lp3
    cmp cx, x1
    jne lp4
    jmp ex2
lp3:
    dec dx
    jmp c2
lp4:
    dec cx
    jmp c2
ex2:
    mov ah, 1
    int 21h

    mov ax, 4c00h
    int 21h
 InputInt proc

    mov ah,0ah
    xor di,di
    mov dx,offset buff ; адрес буфера
    int 21h ; принимаем строку
    mov dl,0ah
    mov ah,02
    int 21h ; выводим перевода строки

; обрабатываем содержимое буфера
    mov si,offset buff+2 ; берем адрес начала строки
    cmp byte ptr [si],"-" ; если первый символ минус
    jnz ii1
    mov di,1  ; устанавливаем флаг
    inc si    ; и пропускаем его
ii1:
    xor ax,ax
    mov bx,10  ; основание сс
ii2:
    mov cl,[si] ; берем символ из буфера
    cmp cl,0dh  ; проверяем не последний ли он
    jz endin

; если символ не последний, то проверяем его на правильность
    cmp cl,'0'  ; если введен неверный символ <0
    jb er
    cmp cl,'9'  ; если введен неверный символ >9
    ja er

    sub cl,'0' ; делаем из символа число
    mul bx     ; умножаем на 10
    add ax,cx  ; прибавляем к остальным
    inc si     ; указатель на следующий символ
    jmp ii2     ; повторяем

er:   ; если была ошибка, то выводим сообщение об этом и выходим
    mov dx, offset error
    mov ah,09
    int 21h
    int 20h

; все символы из буфера обработаны число находится в ax
endin:
    cmp di,1 ; если установлен флаг, то
    jnz ii3
    neg ax   ; делаем число отрицательным
ii3:
    ret

error db "incorrect number$"
buff    db 6,7 Dup(?)
InputInt endp
end begin
```

После написания кода программы программа была запущена в среде для эмуляции 16-разрядного процессора. Рисунок 1 содержит содержимое экрана буфера – результат работы программы при вводе значений 20 и 100.

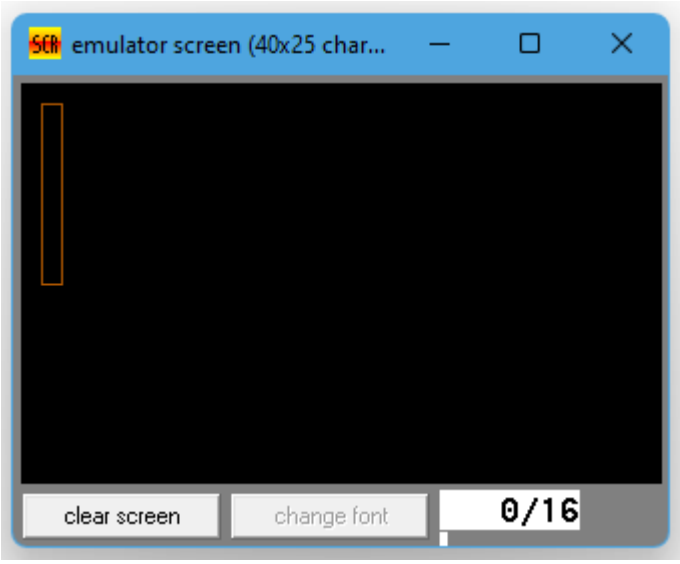

Рисунок 1 – Результат выполнения программы при вводе значений 20 и 100

Результаты тестирования полностью соответствуют ожиданиям.

Выводы

В ходе выполнения данной лабораторной работы были изучены способы функционирования клавиатуры и подключения ее к процессору, принципы отображения цифровой информации в жидкокристаллических дисплеях, методы программирования ввода-вывода данных. Были исследованы особенности функционирования микропроцессора при реализации ассемблерных функций ввода данных с клавиатуры и вывода их на экран монитора. Были приобретены практические навыки программирования на языке ассемблера МП 8086 процедур ввода вывода с использованием функций BIOS.