Manual Tecnico*

* Practica 1: Arquitectura y Ensambladores Sección A
Douglas Daniel Aguilar Cuque 201503935

*

Resumen– En este documento se describe el ensamblador utilizado en la practica 1, tambien se describe las partes relevantes del codigo, el modo de video utilizado y las interrupciones utilizadas en la practica.

Palabra clave-INT 21H, INT 10H, Modo video, registro.

***Abstract*—This document describes the assembler used in practice 1, also describes the relevant parts of the code, the video mode used and the interruptions used in practice.**

***Index Terms*—INT 21H, INT 10H, Video mode, registration**

I. Marco Teórico

A. NASM: el ensamblador de Netwide

El ensamblador Netwide, NASM, es un ensamblador 80x86 y x86-64 diseñado para portabilidad y modularidad. Es compatible con una amplia gama de formatos de archivo de objetos, incluyendo Linux y *BSD a.out, ELF, COFF, Mach-O, 16 bits y 32 bits OBJ de formato (OMF), Win32y Win64. También generará archivos binarios simples, formatos Intel He y S-Record de Motorola. Su sintaxis está diseñada para ser simple y fácil de entender, similar a la sintaxis en el Manual del desarrollador de software Intel con una complejidad mínima. Es compatible con todas las extensiones de arquitectura x86 actualmente conocidas y tiene un gran soporte para macros.

NASM también viene con un conjunto de utilidades para manejar el RDOFFformato de archivo de objeto personalizado.

B. Explicación de partes relevantes del código

1) Macro Lectura de archivo: Primero se limpian los registros ax, cx y dx con la operacion xor entre ellos mismos. Despues se intenta abrir el archivo, si el archivo no existe muestra un mensaje de error, luego lee un total de 255 bites. De ultimo mueve la cantidad de caracteres leidos a una variable. De ultimo cierra el archivo.

2) Macro escritura de archivo: Primero crea el archivo, si no lo crea regresa al menu. Despues de crear el archivo, se abre el archivo en modo escritura. Luego se escribe el numero de caracteres que se ingresaron. Al final se cierra el archivo.

```
%macro leerarchivo 3
    ; 1 ruta archivo, 2 contenedor de los datos archivo
    ; 3 se pondra el numero de caracteres leidos
    ;limpiar ax,cx,dx
    xor ax, ax
    xor cx, cx
    xor dx, dx
    ;abrir archivo
    mov ah, 3dh
    mov cx, 00
    mov dx, %1
    int 21h
    ;archivo no existe
    jc noarchivo
    ;archivo si existe se lee
    mov bx, ax
    mov ah, 3fh
    mov cx, 255
    mov dx, [%2]
    int 21h

    ;numero de caracteres
    mov [%3], ax

    mov ah, 3eh
    int 21h
%endmacro
```

Fig. 1. Codigo Macro lectura de archivo

```
%macro escribir_reporte 3
    ; 1 nombre archivo, 2 numero de caracters,
    ; 3 texto a escribir
    ; crear archivo
    mov ah, 3ch
    mov cx, 0
    mov dx, %1
    int 21h
    jc menu
    mov bx, ax
    mov ah, 3eh
    int 21h

    ;abrir el archivo
    mov ah,3dh
    mov al,1h ;Abrimos el archivo en solo escritura.
    mov dx, %1
    int 21h
    jc menu ;Si hubo error

    ;Escritura de archivo
    mov bx,ax ; mover hadfile
    mov cx,[%2] ;num de caracteres a grabar
    mov dx,%3
    mov ah,40h
    int 21h

    cmp cx,ax
    jne menu ;error salir
    mov ah,3eh  ;Cierre de archivo
    int 21h
%endmacro
```

Fig. 2. Codigo Macro Escritura de archivo

3) Comprobacion de caracteres: Primero se mueve alregistro esi el contenido del archivo leido. Se utiliza lodsb para mover un byte al registro al, para luego compararlo con los caracteres permitidos. Si es un caracter permitido se continua con el analisis, si no es un caracter permitido lo imprime y continua con el analisis.

```
    mov esi, [textoleido]
    jmp comporbarcaracter


comporbarcaracter:
    lodsb
    cmp al, 32
    je repetircomp
    cmp al, 43
    je repetircomp
    cmp al, 42
    je repetircomp
    cmp al, 45
    je repetircomp
```

Fig. 3. Parte de la comprobacion

4) Operar archivo: Se realiza la operacion pop ax la cual representa el signo del segundo numero, se hace un pop eax el cual es el numero dos. Limpia los registros, luego realiza otro pop ax, el cual es el signo del primer numero, despues realiza otro pop eax el cual es el numero uno. Pasa el signo de la operacion al registro al, compara el registro con los numeros ascii de cada operacion y realiza la operacion correspondeinte.

```
cargaoperdor:
    mov [operacionseigno], al
    xor ax, ax
    xor al, al
    pop ax
    mov [signum2], al

    xor eax, eax
    pop eax
    mov [num2], eax


    xor ax, ax
    xor al, al
    pop ax
    mov [signum1], al

    xor eax, eax
    pop eax
    mov [num1], eax


    mov al, [operacionseigno]

    cmp al, 43; +
    je suma2
    cmp al, 45
    je resta2
    cmp al, 42
    je multiplicacion2
    cmp al, 47
    je division2
```

Fig. 4. Colocar los numeros y mandar a operar

5) Fin de una operacion: Al terminar una operacion hace un salto a la parte cargapila, la cual mueve el resultado al registro eax y realiza un push eax. Despues pasa el signo del resultado al registro al y hace un push al registro ax.

```
cargapila:
    xor eax, eax
    mov eax, [resultado]
    push eax
    xor al, al
    xor ax, ax
    mov al, [sigres]
    push ax
    jmp cargasiguiente
```

Fig. 5. Cargar a pila resultado

6) Factorial: Es un loop en el cual se multiplica el resultado con el registro cx, luego se guarda el resultado, se imprime el signo de multiplicacion y el numero en el que esta.

```
operacionfactorial:
    xor ax, ax
    xor bx, bx
    mov ax, [facresul]
    mov bx, cx
    mul bx
    mov [facresul], ax

    mov dl, [digfac]
    add dl, [uno]
    mov [digfac], dl

    mov ah, 02h
    mov dx, 42
    int 21h

    mov ah, 02h
    mov dx, [digfac]
    add dx, 30h
    int 21h

loop operacionfactorial
    ret
```

Fig. 6. loop factorial

C. Modo video

1) INT 10H Función 06H: Esta funcion se utilizo para limpiar la pantalla.

Desplazar líneas de texto hacia arriba

- AH = 06H
- AL = Número de líneas a desplazar. Si AL=0, se borra toda la ventana seleccionada mediante los registros CX y DX
- BH = Atributo a usar en las líneas borradas.

- CH = Línea donde comienza la ventana de texto.
- CL = Columna donde comienza la ventana de texto.
- DH = Línea donde acaba la ventana de texto.
- DL = Columna donde acaba la ventana de texto.

.

```
mov ax,0600h
mov bh,0fh ;0
mov cx,0000h
mov dx,184Fh
int 10h
```

Fig. 7. INT 06H

2) INT 10H Función 02H: Esta funcion se utilizo para posicionar el cursor despues de limpiar la pantalla

- AH = 02H
- BH = Página de vídeo
- DH = Línea donde situar el cursor
- DL = Columna donde situar el cursor

.

```
mov ah,02h
mov bh,00
mov dh,00
mov dl,00
int 10h
ret
```

Fig. 8. INT 02H

D. Interrupciones utilizadas

1) INT 21H Función 01H: Leer un caracter con eco

- AH = 01H
- AL = Código ASCII del Carácter leído y Echo a pantalla

.

2) INT 21H Función 02H: imprimir un caracter

- AH = 02H
- DL = Código ASCII a enviar al dispositivo de salida.

.

3) INT 21H Función 08H: Leer un caracter sin eco

- AH = 08H
- AL = Código ASCII del Carácter leído

.

4) INT 21H Función 3CH: Crea un fichero

- AH = 3CH
- CX = Atributos del fichero: 00H Fichero Normal. 01H Fichero de Sólo Lectura. 02H Fichero Oculto. 03H Fichero de Sistema. DS:DX = Segmento: Desplazamiento de una cadena ASCII con el nombre de fichero.
- Si se ejecutó correctamente: Flag de acarreo (Cf) = 0 AX = Handle o manejador de fichero. Si NO se ejecutó correctamente: Flag de acarreo (Cf) = 1
- AX = Código de error.

.

5) INT 21H Función 3DH: Abre un fichero

- AH = 3DH
- AL = Modo de acceso. Bits 0-2: Permiso de lectura/escritura. DS:DX = Segmento: Desplazamiento de una cadena ASCII con el nombre de fichero.
- Si se ejecutó correctamente Flag de acarreo (Cf) = 0 AX = Handle o manejador de fichero
- Si NO se ejecutó correctamente: Flag de acarreo (Cf) = 1 AX = Código de error.

.

6) INT 21H Función 40H: Escribe en un fichero

- AH = 40H
- BX = Handle.
- CX = Número de bytes a escribir.
- DS:DX = Segmento: Desplazamiento del buffer desde donde se van a tomar los caracteres a escribir.
- Si se ejecutó correctamente: Flag de acarreo (Cf) = 0 AX = Bytes transferidos.
- Si NO se ejecutó correctamente: Flag de acarreo (Cf) = 1 AX = Código de error.

.

7) INT 21H Función 3EH: Cierra un fichero

- H = 3EH
- BX = Handle.
- Si se ejecutó correctamente: Flag de acarreo (Cf) = 0
- Si NO se ejecutó correctamente: Flag de acarreo (Cf) = 1 AX = Código de error.

.

REFERENCES

[1] INT 10H, Modo video, INT 10H Recuperado de: http://ict.udlap.mx/people/oleg/docencia/Assembler/asm$_{i}nterrup_{1}0.html$

[2] INT 21H, interrupción, INT 21H Recuperado de: http://ict.udlap.mx/people/oleg/docencia/ASSEMBLER/asm$_{i}nterrup_{2}1.html$

[3] Manejo de archivos, Ensamblador básico, archivo Recuperado de: http://miensamblador.blogspot.com/2013/07/manejo-de-archivos.html

[4] NASM, NASM - The Netwide Assembler, NASM Recuperado de: https://www.nasm.us/doc/nasmdoc0.html