

# Práctica 7



## Sistemas numéricos

### Objetivo

El alumno se familiarizará con el desarrollo de procedimientos para el manejo de sistemas numéricos en el procesador 8088.

### Equipo

Computadora personal con el software TASM y TLINK.

### Teoría

Resumen sobre conversiones numéricas.

### Desarrollo

1. Ensamble, encadene y ejecute el programa **Prac7.asm** que se muestra en el Listado 1. El programa realiza un desplegado en pantalla del valor del registro AL en formato binario y hexadecimal. Ensamble y encadene el programa para diferentes valores de AL.
2. En un archivo diferente y siguiendo la plantilla formato.asm, diseñe e implemente el procedimiento **printNumBase**, el cual imprime la palabra dada en el registro **AX** en el formato según la base dada en el registro **BX**.

Por ejemplo:

```
mov ax,94Ah          ; valor a imprimir (equivalente a 2378 decimal)
mov bx,8              ; base seleccionada
call printNumBase     ; imprime 4512

mov ax,10h            ; valor a imprimir (equivalente a 16 decimal)
mov bx,17              ; base seleccionada
call printNumBase     ; imprime G
```

En el código anterior se ejemplifica la impresión del registro AX en base octal y en base 17, sin embargo, el procedimiento debe ser **funcional para cualquier base solicitada** que sea imprimible de acuerdo al límite de caracteres en la tabla ASCII. El procedimiento debe ser **genérico**, no realice invocaciones a **printBin**, **printDec** o **printHex**, haga la conversión por medio de divisiones.

3. Programe el procedimiento **atoi**, el cual recibe en **BX** un apuntador a una cadena terminada en null, que contiene una serie de números ASCII en formato decimal, la cadena es convertida a su correspondiente valor numérico regresándolo en el registro **AX**.

Ejemplo:

```
mov bx, offset cadena      ; si cadena es "1234" regresa:
call atoi                  ; AX=1234 decimal, o lo que es
                           ; equivalente AX=04D2 hexadecimal
```

4. Programe el procedimiento **esAutomorfico**, el cual recibe una palabra en el registro **DX** y retorna un **1** en **AX** si el número es automórfico, caso contrario retorna **0**.

Un número  $n$  es automórfico si los últimos dígitos de  $n^2$  son  $n$ .

Ejemplos:

$$6^2 = 3\mathbf{6}$$

$$25^2 = 6\mathbf{25}$$

$$76^2 = 57\mathbf{76}$$

Algunos números automórficos son los siguientes: 0, 1, 5, 6, 25, 76, 376, 625, 9376

## Conclusiones y comentarios

## Dificultades en el desarrollo

## Referencias

## Listado 1.

```
MODEL small
.STACK 100h

;----- Insert INCLUDE "filename" directives here
;----- Insert EQU and = equates here

INCLUDE procs.inc

LOCALS

.DATA
mens_asci db "AL desplegado en ASCII:",0
mens_bin  db "AL desplegado en Binario:",0
mens_dec  db "AL desplegado en Decimal:",0
mens_hex  db "AL desplegado en Hexadecimal:",0
new_line db 13,10,0

.CODE ;----- Insert program, subroutine call, etc., here

Principal PROC
mov ax,@data ;Inicializar DS al la direccion
mov ds,ax ; del segmento de datos (.DATA)
call clrscr

mov al,7Bh ; dato a desplegar
mov dx, offset mens_asci
call puts
call putchar ; imprime AL en ASCII

mov dx, offset new_line
call puts

mov dx, offset mens_bin
call puts
call printBin ; desplegar AL en binario

mov dx, offset new_line
call puts

mov dx, offset mens_dec
call puts
call printDec ; desplegar AL en decimal

mov dx, offset new_line
call puts

mov dx, offset mens_hex
call puts
call printHex ; desplegar AL en decimal

mov ah,04ch ; fin de programa
mov al,0
int 21h
ret
ENDP
```

```

; --- procedimientos ----

printBin PROC
    push ax          ; salvar registros a utilizar
    push cx
    mov cx,8         ; inicializar conteo a 8
    mov ah,al        ; AH sera el registro a desplegar
@@nxt: mov al,'0'    ; prerar a AL para imprimir ASCII
    shl ah,1         ; pasar el MSB de AH a la bandera de acarreo
    adc al,0         ; sumar a AL el valor del acarreo
    call putchar
    loop @@nxt       ; continuar con el proximo bit
    pop cx           ; recuperar registros utilizados
    pop ax
    ret
ENDP

;*****

printDec PROC
    push ax          ; salvar registro a utilizar
    push bx
    push cx
    push dx
    mov cx,3         ; inicializar conteo a 3 (cent-dec-unida)
    mov bx,100       ; iniciar con centenas
    mov ah,0         ; asegurar AX = AL
@@nxt: mov dx,0      ; asegurar DX=0 para usar div reg16
    div bx           ; dividir DX:AX entre BX
    add al,'0'       ; convertir cociente a ASCII
    call putchar     ; desplegar digito en pantalla
    mov ax,dx        ; pasar residuo (DX) a AX
    push ax          ; salvar temporalmente AX
    mov dx,0         ; ajustar divisor para nuevo digito
    mov ax,bx        ; la idea es:
    mov bx,10        ; BX = BX/10
    div bx           ;
    mov bx,ax        ; pasar cociente al BX para nuevo digito
    pop ax           ; recupera AX
    loop @@nxt       ; proximo digito
    pop dx
    pop cx
    pop bx
    pop ax
    ret
ENDP

;*****

printHex PROC
    push ax          ; salvar registros a utilizar
    push bx
    push cx
    mov ah,0         ; asegurar AX = AL
    mov bl,16        ;
    div bl           ; dividir AX/16 --> cociente en AL y residuo AH
    mov cx,2         ; para imprimir dos digitos hex
@@nxt: cmp al,10     ; verifica si cociente AL es menor a 10
    jb @@print
    add al,7
@@print: add al,30h   ; si es menos a 10 sumar 30h de lo contrario 37h
    call putchar
    mov al,ah        ; pasa residuo (AH) a AL para imprimirlo
    loop @@nxt       ; proximo digito
    pop cx
    pop bx
    pop ax           ; recupera registros utilizados
    ret
ENDP

END

```