Estructura de Computadores

Tema 2. El lenguaje ensamblador Repertorio de instrucciones y lenguaje ensamblador de la arquitectura de 32 bits de Intel (IA32).

---

1. Ejercicios Resueltos

1.1.

Escribir una rutina en ensamblador de la arquitectura Intel (IA32) que convierta a un número entero, sin signo, de 32 bits a una cadena de caracteres ASCIIZ.

El número a convertir y la dirección del principio de la cadena de caracteres se la han de pasar a la rutina a través de la pila, según el siguiente prototipo de la rutina:

```
BinToAsciiz proto near pascal, NUM:DWORD, CADENA:FAR PTR BYTE
```

A la salida, en `AX` se devolverá el nº de cifras decimales (1..10) del número transformado, no debiéndose modificar ningún otro registro.

Solución

```
;======================================== BinToAsciiz ============
; Convierte un número entero, sin signo, de 32 bits a ASCIIZ
; El resultado es desde '0',0 hasta '4294967296',0
;-------------------------------------------------------------------
; Entrada:  parámetros por la pila (NUM,CADENA)
;           NUM = número de 32 bits sin signo a convertir
;           CADENA = puntero al destino de la cadena ASCIIZ
; Salida:   AX = nº de cifras decimales (1..10)
;===================================================================
BinToAsciiz proc    near pascal   USES    ds ebx cx edx si,
                NUM:DWORD, CADENA:FAR PTR BYTE
        LOCAL   CifrDec:WORD
        mov     ebx,10
        mov        eax,NUM
        xor     ecx,ecx         ; CX = nº de cifras decimales
        .REPEAT
            xor     edx,edx     ; se divide EDX:EAX entre EBX
            div     ebx         ; y el cociente queda en EAX
            push    dx          ; y el resto (de 0 a 9) en EDX
            inc     cx          ; una cifra decimal más
        .UNTIL  !EAX            ; si EAX = 0, se termina

        mov     CifrDec,cx

        lds        si,CADENA
        .REPEAT
            pop     ax          ; ahora se extraen las cifras en orden
                                ; opuesto a como se obtuvieron
```

```
            or      al,'0'      ; se convierten a ASCII
            mov     BYTE PTR [si],al ; y se pasan a la cadena
            inc     si
        .UNTILCXZ               ; de izquierda a derecha

        xor     al,al
        mov     BYTE PTR [si],al ; termina la cadena
        inc     si
        mov     ax,CifrDec      ; devuelve en AX el nº de cifras

        ret
Bin4Asc endp
```

1.2.

Dado un procesador de 32 bits con un espacio direccionable de memoria de 4 GB, direccionada a nivel de byte. El banco de registros consta de 128 registros mientras que la unidad aritmético-lógica es capaz de realizar 16 operaciones aritméticas y lógicas. El repertorio de instrucciones se compone de 64 instrucciones distintas.

Considérese el siguiente fragmento de programa, que utiliza instrucciones del juego del MIPS32.

```
lw      $t1, 0x0010($t0)
lw      $t2, 0x1008($t0)
add     $t3, $t1, $t2
sw      $t4, 0x0014($t0)
```

Considerar la instrucción de máquina `SWAPD dir1, dir2` que intercambia el contenido de los datos almacenados en las posiciones de memoria principal `dir1` y `dir2`. Se pide:

1. Indicar el formato que se necesita para codificar la instrucción anterior.
2. El juego de instrucciones del MIPS32 no incluye la instrucción de máquina anterior, indíquese el conjunto de instrucciones del juego del MIPS32 que habría que ejecutar para obtener el mismo resultado.

Respuesta.-

1. La instrucción consta de tres campos: el primero es el código de operación y los dos restantes son las direcciones de memoria cuyo contenido se intercambia. Dado que la instrucción emplea un direccionamiento absoluto, el número de bits necesario para almacenar una dirección de memoria es de $\log_2(4\text{ GB}) = 32$. De este modo, la instrucción necesita tres palabras para su codificación. La primera palabra tendrá el campo del código de operación, cuyo tamaño es de $\log_2(64) = 6$ bits. Los 26 bits restantes de la primera palabra quedarían sin uso. La segunda palabra está ocupada por el campo de la primera dirección de memoria, de 32 bits de extensión. De forma análoga, la tercera palabra de la instrucción está ocupada por la segunda dirección de memoria, de 32 bits de extensión.
2. El juego de instrucciones equivalente para el MIPS32 es:

```
la      $t1, dir1
la      $t2, dir2
sw      $t1, dir2
sw      $t2, dir1
```
