



Arquitecturas y Organización de Computadoras I  
2° Cuatrimestre 2020

TP N° 6 – Programación en lenguaje ensamblador MIPS  
con MIPSX



**Objetivo:** Comprender la estructura de un programa en lenguaje ensamblador MIPS básico. Introducción a la interfaz `mipsx` y al conjunto de instrucción MIPS.

**Recursos y Bibliografía:**

*Arq. MIPS Vol I, II and III.*

Programa `mipsx` desarrollado por la cátedra.

**1.** Cree un programa que dado un número flotante de precisión simple extraiga los valores de signo, exponente y mantisa, guardando 1 si es positivo y -1 si es negativo en la variable **signo**, guardando el exponente traducido de notación en exceso a complemento a dos (es decir, habiéndose restado 128) en la variable **exponente**, y la mantisa sin modificar en la variable **mantisa**.

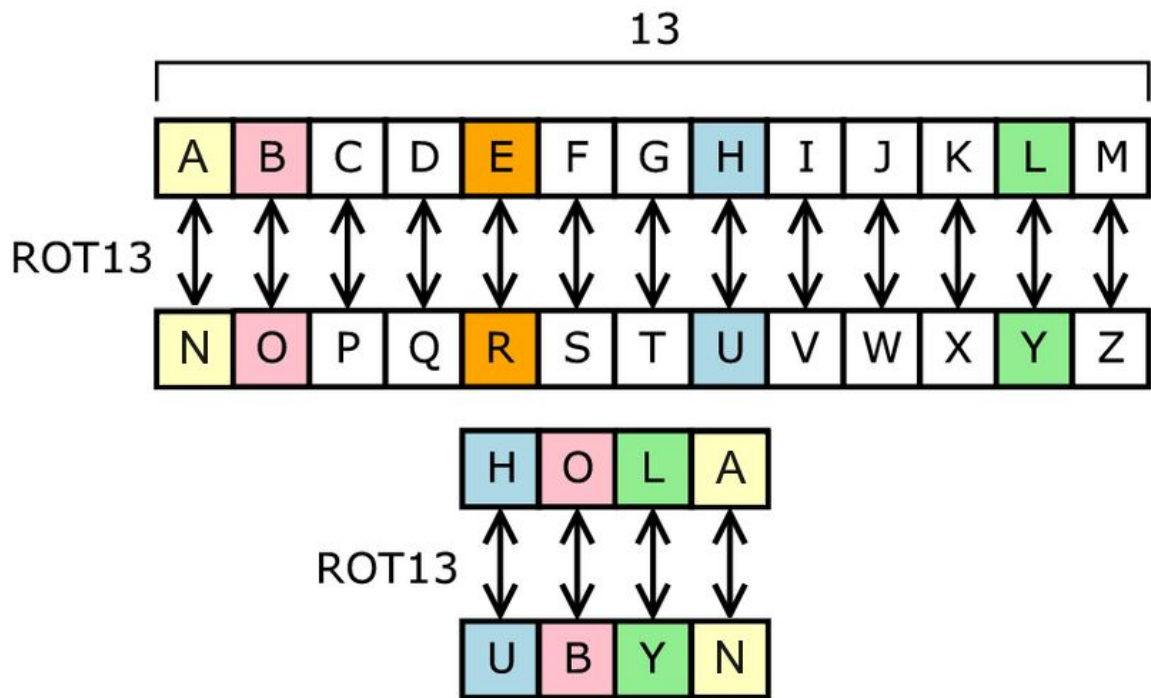
```
.data
memoria:
flotante:
    .float 3.14
signo:
    .byte 0 # -1 si es negativo, 1 si es positivo
exponente:
    .byte 0 # En C2
mantisa:
    .word 0 # Sin modificar del punto flotante,
            # es decir sin agregar el cero.
```

**2.** Dado el siguiente segmento de datos donde “**Imagen**” representa una imagen de 9 pixeles con el formato RGB donde cada byte es la intensidad de color que compone cada pixel. Realizar un programa que haga una conversión de colores a escala de grises, guardando el resultado en **destino**.

Para pasar a escala de grises deben realizar el promedio entre las tres componentes que forman el píxel, donde cada componente es un entero sin signo que va de 0..255.

```
.data
memoria:
Imagen:
    .byte 123,0,33,44,55,2,56,78,66
    .byte 0,33,44,55,2,56,78,66,123
    .byte 33,44,55,2,56,78,66,23,55
destino: .space 27
```

**3.** Cree un programa que aplique el cifrado ROT13 a una cadena de texto ASCII. Este cifrado sustituye las letras de la A a la M por aquellas de la N a la Z, y viceversa.



4. Dado el siguiente programa escrito en código ensamblador de mips, y suponiendo que el compilador traduce cada pseudo instrucción a 3 instrucciones máquina ¿Cuántos bytes ocupa el segmento de texto? Si la dirección de la primera instrucción es 0x400B0 ¿Cuál es la dirección de la última instrucción? ¿Cuál es la dirección del último byte del segmento de texto?

```
li $t0, 5
add $t3, $t0, $t0
move $t4, $t0
la $t0, memoria
lw $t3, 11($t0)
ori $t3, $t4, 1
j start
blt $t3, $7, menor
lw $t3, valor
```

5. Traducir a lenguaje ensamblador de **MIPS** el siguiente fragmento de código en alto nivel, sabiendo que el vector V es un vector de halves, y el resto de las variables enteras.

```
for (i=0; i<n-1; i++)
{
    for (j=i+1; j<n; j++)
    {
        if(V[i]>V[j])
        {
            aux = V[i];
            V[i] = V[j];
            V[j] = aux;
        }
    }
}
```