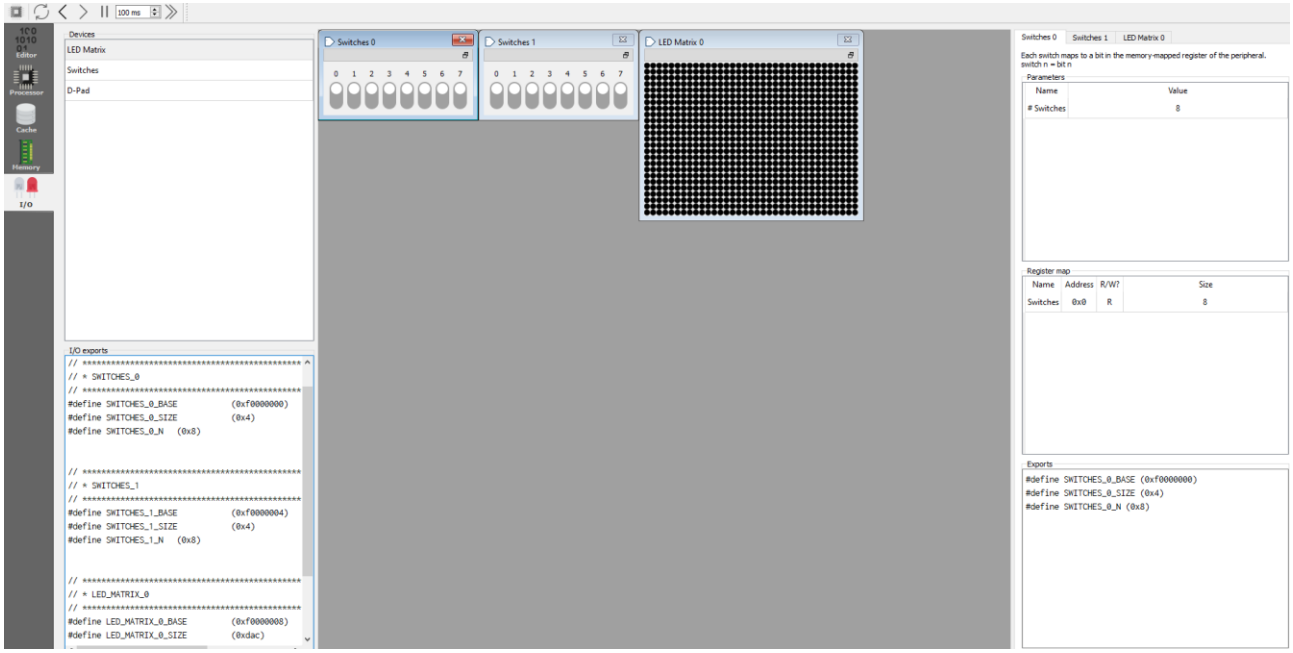
PRÁCTICA 9
ENTRADA/SALIDA II

Objetivos:

- Aplicar los conceptos de entrada/salida por consulta de estado.

Desarrollo / Comentario:
Crear Switch 0 , switch 1 y Matriz de Led 0.



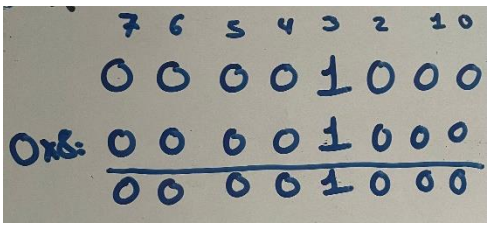
En esta práctica vamos a trabajar con switch0 como registro de estado.

a) Escribir un programa que cuando el bit 3 del switch 0 valga 1 se muestre en la matriz de led en amarillo tantos leds como indique switch 1.

```
.text
li a0, LED_MATRIX_0_BASE        #Carga la dirección del primer LED de la matriz
                                #Que va de 4 en 4 la dirección de memoria
li s0, 0xffff00 #amarillo en RGB #Carga el amarillo

poll1:
lw s1, SWITCHES_0_BASE          #Carga el valor de BITS encendidos en Switch0
andi s1,s1,0x8                  #Filtra para ver si está encendido el bit 3
beq s1,zero,poll1               #Si no, salta otra vez y espera
```

"Si quisiésemos ver el bit 4, habría que hacer un andi con 10, porque va de 2 en 2"



```
        lw s2, SWITCHES_1_BASE          #Leemos cantidad de switch(2^n) y lo dejamos
                                        #en entero para hacer un contador
bucle:
        beq s2,zero,final               #Si no quedan enteros por poner, se acaba

        sw s0, 0(a0)                    #Cargamos el amarillo
        addi a0,a0,4                    #Pasamos al siguiente LED
        addi s2,s2,-1                   #Quitamos uno del contador porque ya se puso
        j bucle                         #Repetimos proceso
final:
        li a7,10
        ecall
```

b) Escribir un programa que cuando el bit 3 del switch 0 valga 1 se muestre en la matriz de led en amarillo tantos leds como indique switch 1 y cuando el bit 4 del switch0 valga 1 se ponga en blanco toda la matriz.

```
        .text
        li s0, 0xffff00 #amarillo en RGB    #Cargamos amarillo
        li s3,0xffffff #blanco en RGB       #Cargamos blanco

poll1:                                      #Problema1
        lw s1, SWITCHES_0_BASE              #Cargamos el valor que haya en Switch0
        andi s1,s1,0x8     #bit 3           #Filtramos en busca del bit3 de Switch0
        bne s1,zero,amarillo                #Si no es 0 (es 1), salta a amarillo


poll2:                                      #Problema2
        lw s1, SWITCHES_0_BASE              #Cargamos el valor que haya en Switch0
        andi s1,s1,0x10   #bit 4            #Filtramos en busca del bit4 de Switch0
        bne s1,zero,blanco                  #Si no es 0 (es 1), salta a blanco
        j poll1                             #Si no ocurre nada, salta arriba y repite

amarillo:                                   #Pinta los bits que le digas de amarillo
        li a0, LED_MATRIX_0_BASE            #Carga la matriz en a0
        lw s2, SWITCHES_1_BASE              #Carga el valor de Switch1 en s2
bucle:
        beq s2,zero,poll2                   #Si no queda en el contador, va a comprobar
                                            #está encendido aún
        sw s0, 0(a0)                        #Guarda el amarillo
        addi a0,a0,4                        #Pasa al siguiente
        addi s2,s2,-1                       #Quita uno del contador
        j bucle                             #Salta a bucle y repite proceso

blanco:                                     #Lo pinta entero de blanco
        li a0, LED_MATRIX_0_BASE            #Carga matriz
        li a1, LED_MATRIX_0_SIZE            #es el tamaño en bytes de la matriz
        srli a1, a1,2                       #divido por 4 para tener número de leds a pintar
```

```
bucle2:
        beq a1,zero,final               #Si el tamaño que queda es cero, se acaba
        sw s3, 0(a0)                    #Guarda el blanco
        addi a0,a0,4                    #Pasa al siguiente LED
        addi a1,a1,-1                   #Resta uno a la cantidad que queda
        j bucle2
final:
        j poll1

        li a7,10
        ecall
```

Otra forma con SUBRUTINAS

```
        .text
main:
poll1:
        lw s1, SWITCHES_0_BASE
        andi s1,s1,0x8 #bit 3
        bne s1,zero,amarillo

poll2:
        lw s1, SWITCHES_0_BASE
        andi s1,s1,0x10 #bit 4
        bne s1,zero,blanco
        j poll1

amarillo:
        la a0, 0xffff00 #amarillo en RGB
        lw a1, SWITCHES_1_BASE #numero de leds a pintar
        call pintar

        j poll2

blanco:
        li a0, 0xffffff #blanco en RGB
        li a1, LED_MATRIX_0_SIZE #es el tamaño en bytes de la matriz
        srli a1, a1,2  #divido por 4 para tener el número de leds a pintar
        call pintar

        j poll1

final:

        li a7,10
        ecall
pintar:
        #recibe en a0 el color a pintar
        #recibe en a1 el número de leds pintar

        li t0, LED_MATRIX_0_BASE
buclepinta:
```

```
        beq a1,zero,finpinta
        sw a0, 0(t0)
        addi t0,t0,4
        addi a1,a1,-1
        j buclepinta

finpinta:
        ret
```