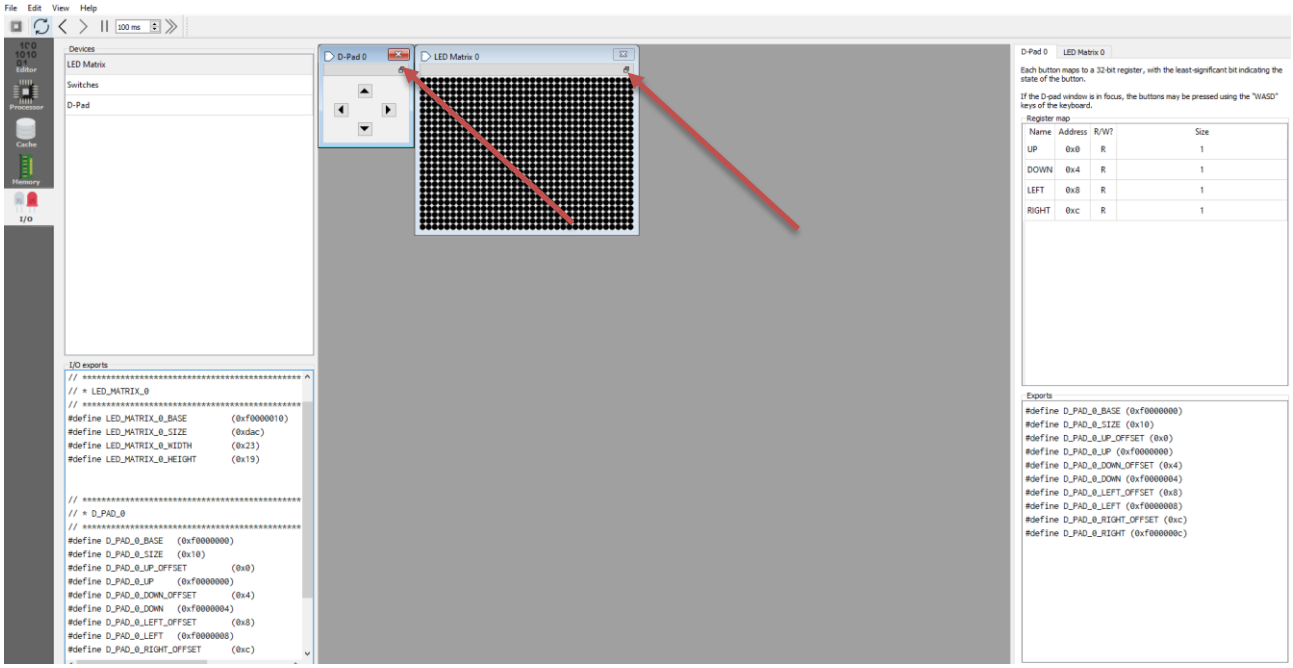
PRÁCTICA 10
ENTRADA/SALIDA III

Objetivos:

- Aplicar los conceptos de entrada/salida.

Desarrollo / Comentario:

Crear D-Pad 0 y Matriz de Led 0. Para poder visualizarlos mientras se ejecuta debéis hacer un click debajo del aspa de D-Pad0 y LEDmatriz0.



Realizar un programa que vaya pintando un punto sobre la matriz de Led 0 en las direcciones indicadas en el D-Pad0 respetando el tamaño de la matriz. Es decir cada vez que se active D_PAD_0_UP se pinte un led hacia arriba, cuando se active D_PAD_0_DOWN se pinte un led hacia abajo, cuando se active D_PAD_0_LEFT se pinte un led hacia la izquierda y cuando se active D_PAD_0_RIGHT se pinte un led hacia la derecha. Como ayuda se os proporciona la función pixel que pinta un led cuyo color está en a2 en la columna a1 y fila a0 de la matriz cuya dirección base es LED_MATRIX_0_BASE y su anchura es LED_MATRIX_0_WIDTH.

```
pixel:
        li t2, LED_MATRIX_0_BASE
        li t3 ,LED_MATRIX_0_WIDTH
        mul t1,a0,t3
        add t1,t1,a1
        slli t1,t1,2
        add t1,t1,t2
        sw a2, 0(t1)
        ret
```

NOTA: LAS BARRAS – SON BARRABAJAS (NO ME VAN)

```
        li s1, LED_MATRIX_0_BASE        #Cargamos las definiciones de los I/0s en
        li s2, LED_MATRIX_0_WIDTH       #variables, guardan su tamaño
        addi s2, s2, -1                 #Restamos 1 a la anchura de matriz para no salir
        li s3, LED_MATRIX_0_HEIGHT      #Restamos 1 a la altura de matriz para no salir
        addi s3, s3, -1                 #Empieza en (0,0), por lo que si el limite es 25,
                                        #hay que parar en 24 -> s3 - 1. Igual para width

main:
        li a0, 5                        #Pintamos en la fila 5 (6 en realidad)
        li a1, 5                        #Pintamos en la columna 5 (6 en realidad)
        li a2, 0xff000                  #Elegimos el color rojo
        call pixel                      #Llamamos a la función pixel para pintarlo

up:
        lw t0, D_PAD_0_UP               #Cargamos el valor en t0 (1 o 0)
        beq t0, zero, down              #Si es 0, no se pulsó y salta al siguiente botón
bucle1:
        lw t0, D_PAD_0_UP               #Volvemos a cargar el valor (por si acaso)
        bne t0, zero, bucle1            #si t0 no es 0 vuelve a esperar a que pueda pasar
        beq a0, zero, up                #Si a0=0,controla para que no salga de la matriz
        addi a0, a0, -1                 #Se resta 1 a la fila para que suba
        call pixel                      #Llamada para pintar el pixel

down:
        lw t0, D_PAD_0_DOWN             #Cargamos el valor en t0 (1 o 0)
        beq t0, zero, down              #Si es 0, no se pulsó y salta al siguiente botón
bucle2:
        lw t0, D_PAD_0_DOWN             #Volvemos a cargar el valor (por si acaso)
        bne t0, zero, bucle2            #si t0 no es 0 vuelve a esperar a que pueda pasar
        beq a0, s3, left                #Si a0=24,controla para que no se salga
        addi a0, a0, 1                  #sumamos 1 para que baje
        call pixel                      #Llamada para pintar el pixel

left:
        lw t0, D_PAD_0_LEFT             #Cargamos el valor en t0 (1 o 0)
        beq t0, zero, right             #Si es 0, no se pulsó y salta al siguiente botón
bucle3:
        lw t0, D_PAD_0_LEFT             #Volvemos a cargar el valor (por si acaso)
        bne t0, zero, bucle3            #si t0 no es 0 vuelve a esperar a que pueda pasar
        beq a1, zero, left              #Si a1=0,controla para que no salga de la matriz
        addi a1, a1, -1                 #restamos 1 para que vaya a la izquierda
        call pixel                      #Llamada para pintar el pixel
```

```
right:
        lw t0, D_PAD_0_RIGHT            #Cargamos el valor en t0 (1 o 0)
        beq t0, zero, up                #Si es 0, no se pulsó y salta al siguiente botón
bucle4:
        lw t0, D_PAD_0_RIGHT            #Volvemos a cargar el valor (por si acaso)
        bne t0, zero, bucle4            #si t0 no es 0 vuelve a esperar a que pueda pasar
        beq a1, s2, left                #Si a1=34, controla para que no se salga
        addi a1, a1, 1                  #sumamos 1 para que vaya a la derecha
        call pixel                      #Llamada para pintar pixel y que vuelva aquí
        j up                            #Salto a up para repetir


pixel:
        li t2, LED_MATRIX_0_BASE
        li t3 ,LED_MATRIX_0_WIDTH
        mul t1,a0,t3
        add t1,t1,a1
        slli t1,t1,2
        add t1,t1,t2
        sw a2, 0(t1)
        ret
```