

Práctica 2 – Entrada y Salida en C

Jaime Arana
Manuel Ferrero
3ºA

Manejo de puertos:

```
1  #include <xc.h>
2
3  int main(void)
4  {
5      // se ponen los LEDs como salida
6      TRISC = ~0x0F;
7
8      // los LEDs son activos a nivel bajo
9      LATC = ~0x0F;
10
11     while(1)
12     {
13         return 0;
14     }
```

En este apartado se pide encender los 4 LEDs conectados a los bits menos significativos del puerto C, siendo los bits RCO – RC3.

En este primer apartado se ha optado por no utilizar máscaras y escribir el puerto C directamente. No obstante, para los siguientes apartados si se va a trabajar con máscaras.

Diseño 1:

Encender LED RC2

```
1  #include <xc.h>
2
3  #define BIT_LED0 0
4  #define BIT_LED1 1
5  #define BIT_LED2 2
6  #define BIT_LED3 3
7
8  int main(void)
9  {
10     // LEDs como salidas
11     TRISC = ~0x0F;
12
13     // variable de ayuda ya que no se puede escribir
14     // el puerto C varias veces seguidas
15     puerto = 0x0000;
16
17     //encendidos LED 2
18     puerto |= (1 << BIT_LED0);
19     puerto |= (1 << BIT_LED1);
20     puerto &= ~(1 << BIT_LED2);
21     puerto |= (1 << BIT_LED3);
22     LATC = puerto;
23
24     while(1)
25     ;
26     return 0;
27 }
```

Encender LED RC2 Y RC3

```
1  #include <xc.h>
2
3  #define BIT_LED0 0
4  #define BIT_LED1 1
5  #define BIT_LED2 2
6  #define BIT_LED3 3
7
8  int main(void)
9  {
10     // LEDs como salidas
11     TRISC = ~0x0F;
12
13     // variable de ayuda ya que no se puede escribir
14     // el puerto C varias veces seguidas
15     puerto = 0x0000;
16
17     //encendidos LED 2 y LED 3
18     puerto |= (1 << BIT_LED0);
19     puerto |= (1 << BIT_LED1);
20     puerto &= ~(1 << BIT_LED2);
21     puerto &= ~(1 << BIT_LED3);
22     LATC = puerto;
23
24     while(1)
25     ;
26     return 0;
27 }
```

Manejo del pulsador:

```
1  #include <xc.h>
2
3  #define PIN_PULSADOR 5
4
5  int main(void)
6  {
7      int pulsador;
8
9      // LEDs como salida
10     TRISC = ~0x0F;
11
12     // LEDs empiezan apagados
13     LATC = 0x0F;
14
15     // pulsador como entrada
16     TRISB = 0xFF;
17
18     while(1){
19         // Se lee el estado del pulsador
20         pulsador = ( PORTB >> PIN_PULSADOR ) & 1;
21         // Cuando se pulsa el pulsador se enciende el LED 0, al dejar de pulsar se apagan
22         if(pulsador == 0){
23             LATC &= ~1;
24         }else{
25             LATC |= 1;
26         }
27     }
28
29     return 0;
30 }
```

Detección de flancos:

```
1  #include <xc.h>
2
3  #define BIT_PULSADOR 5
4
5  int main(void)
6  {
7      int pulsador_ant, pulsador_act;
8
9      TRISC = ~0x0F;
10     LATC = 0x0F;
11     TRISB |= (1 << BIT_PULSADOR);
12
13     //leemos el estado del pulsador
14     //con PORTB --> leemos el valor del puerto
15     //>> BIT_PULSADOR --> leemos valor en el campo del pulsador
16     pulsador_ant = (PORTB >> BIT_PULSADOR) & 1;
17     int contador = 0;
18
19     while(1){
20         pulsador_act = (PORTB >> BIT_PULSADOR) & 1;
21
22         //si el pulsador == 0 --> está pulsado
23         if((pulsador_act != pulsador_ant) && (pulsador_act == 0)){
24             if(contador > 15) { // vuelve a empezar
25                 contador = 0;
26             }
27             contador = contador + 1;
28             //escribimos en el puerto el valor del contador para que se enciendan los LEDs
29             LATC = ~contador;
30         }
31         pulsador_ant = pulsador_act;
32     }
33
34     return 0;
35 }
```

Diseño 2:

```

1  #include <xc.h>
2
3  #define BIT_PULSADOR 5
4  #define BIT_LED0 0
5  #define BIT_LED1 1
6  #define BIT_LED2 2
7  #define BIT_LED3 3
8
9  int main(void)
10 {
11     int pulsador_ant, pulsador_act;
12
13     TRISC = ~0x0F;
14     LATC = 0x0F;
15     TRISB |= (1 << BIT_PULSADOR);
16
17     //leemos el estado del pulsador
18     //con PORTB --> leemos el valor del puerto
19     //>> BIT_PULSADOR --> leemos valor en el campo del pulsador
20     pulsador_ant = (PORTB >> BIT_PULSADOR) & 1;
21     int contador = 0;
22
23     int puerto = 0x0000;
24     //encendemos solo el LED 2
25     puerto &= ~(1 << BIT_LED0);
26     puerto |= (1 << BIT_LED1);
27     puerto |= (1 << BIT_LED2);
28     puerto |= (1 << BIT_LED3);
29     LATC = puerto;
30
31     while(1){
32         pulsador_act = (PORTB >> BIT_PULSADOR) & 1;
33
34         //si el pulsador == 0 --> está pulsado
35         if((pulsador_act != pulsador_ant) && (pulsador_act == 0)){
36             //ponemos todos los campos a 0 --> se encienden los LEDs
37             contador = contador + 1;
38
39             if(contador < 5){
40                 //desplazamos un 0 a la derecha y ponemos un 0 en la posicion anterior
41                 LATC &= ~(1 << contador);
42                 //tenemos que hacer un
43                 asm(" NOP");
44                 LATC |= (1 << (contador - 1));
45             }
46             //si llegamos al final de la secuencia resetear al valor inicial
47             if(contador == 4) {
48                 contador = 0;
49                 LATC = 14;
50             }
51         }
52
53         pulsador_ant = pulsador_act;
54     }
55
56     return 0;
57 }

```