

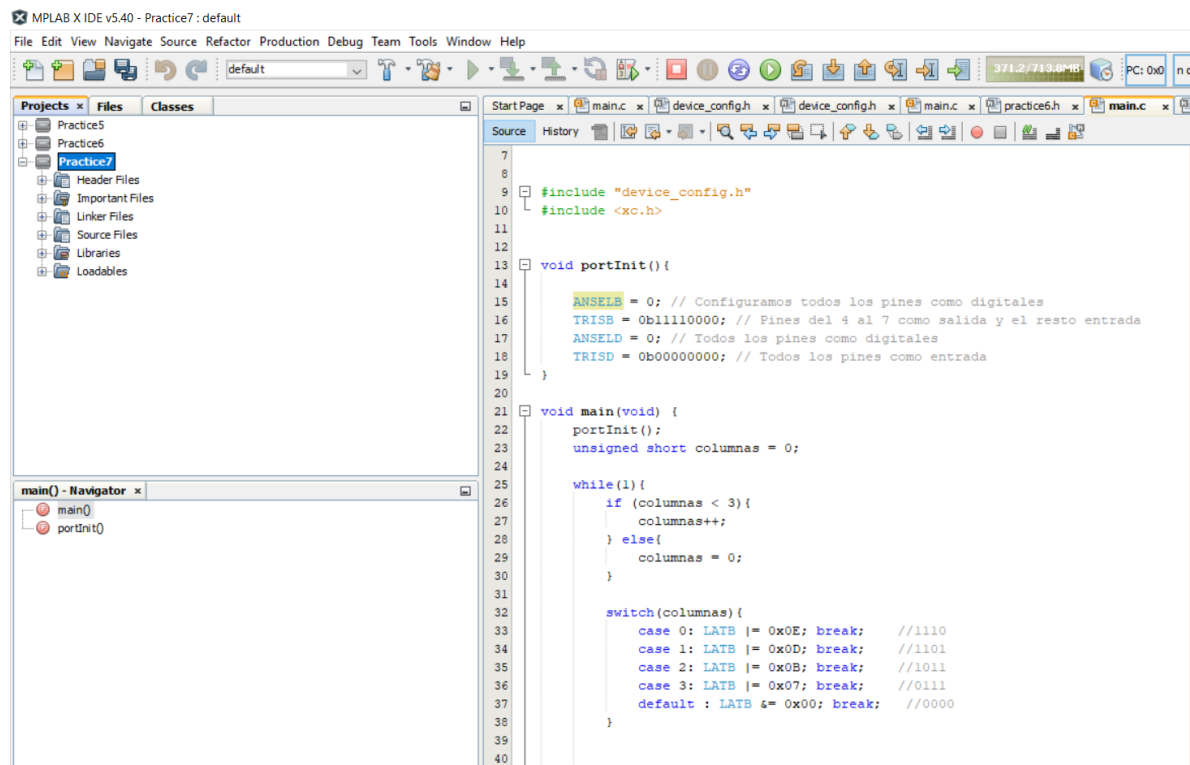
Jesus Ramiro Garza Hernandez

A01410925

Reporte Práctica 7

Ejercicio: Construir un teclado matricial 4x4

Código:



```
7
8
9  #include "device_config.h"
10 #include <xc.h>
11
12
13 void portInit() {
14
15     ANSELB = 0; // Configuramos todos los pines como digitales
16     TRISEB = 0b11110000; // Pines del 4 al 7 como salida y el resto entrada
17     ANSELB = 0; // Todos los pines como digitales
18     TRISD = 0b00000000; // Todos los pines como entrada
19 }
20
21 void main(void) {
22     portInit();
23     unsigned short columns = 0;
24
25     while(1){
26         if (columns < 3){
27             columns++;
28         } else{
29             columns = 0;
30         }
31
32         switch(columns){
33             case 0: LATB |= 0x0E; break; //1110
34             case 1: LATB |= 0x0D; break; //1101
35             case 2: LATB |= 0x0B; break; //1011
36             case 3: LATB |= 0x07; break; //0111
37             default : LATB &= 0x00; break; //0000
38         }
39
40     }
```

Para construir el código se siguió el procedimiento visto en clase, en donde primeramente se inicializaron los puertos como entradas o salidas digitales según cada pin.

Posteriormente se creó el main (void) del programa que incluye las columnas con las que se construye la matriz del teclado. Utilizando la función de “switch case” se declararon los distintos casos que se nos mencionan en el problema para distinguir las columnas y filas activadas.

MPLAB X IDE v5.40 - Practice7 : default

File Edit View Navigate Source Refactor Production Debug Team Tools Window Help

default

Projects Files Classes

Practice5
Practice6
Practice7
Header Files
Important Files
Linker Files
Source Files
Libraries
Loadables

main() - Navigator x

- main()
- portinit()

Source History

```
40  
41     if (PORTB == 0xEE) {  
42         LATD <= 4;  
43         LATD |= 0x01;  
44     } else if (PORTB == 0xDE) {  
45         LATD <= 4;  
46         LATD |= 0x04;  
47     } else if (PORTB == 0xBE) {  
48         LATD <= 4;  
49         LATD |= 0x07;  
50     } else if (PORTB == 0x7E) {  
51         LATD <= 4;  
52         LATD |= 0x0F;  
53     } else if (PORTB == 0xED) {  
54         LATD <= 4;  
55         LATD |= 0x02;  
56     } else if (PORTB == 0xDD) {  
57         LATD <= 4;  
58         LATD |= 0x05;  
59     } else if (PORTB == 0xBD) {  
60         LATD <= 4;  
61         LATD |= 0x08;  
62     } else if (PORTB == 0x7D) {  
63         LATD <= 4;  
64         LATD |= 0x00;  
65     } else if (PORTB == 0xEB) {  
66         LATD <= 4;  
67         LATD |= 0x03;  
68     } else if (PORTB == 0xDB) {  
69         LATD <= 4;  
70         LATD |= 0x06;  
71     } else if (PORTB == 0xBB) {  
72         LATD <= 4;  
73         LATD |= 0x09;
```

MPLAB X IDE v5.40 - Practice7 : default

File Edit View Navigate Source Refactor Production Debug Team Tools Window Help

default

Projects Files Classes

Practice5
Practice6
Practice7
Header Files
Important Files
Linker Files
Source Files
Libraries
Loadables

main() - Navigator x

- main()
- portinit()

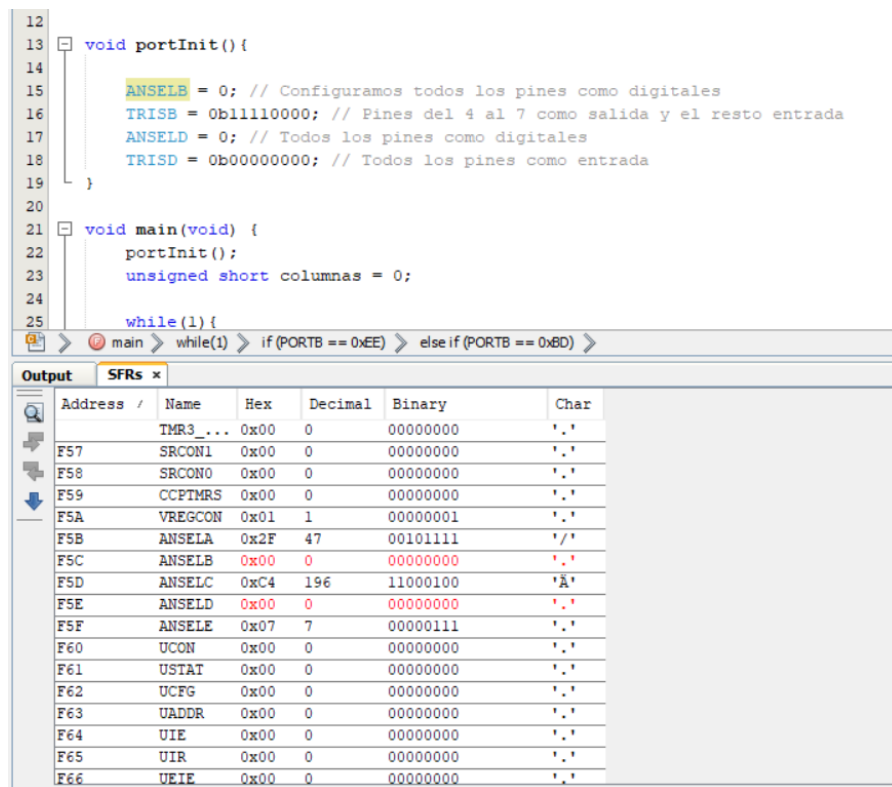
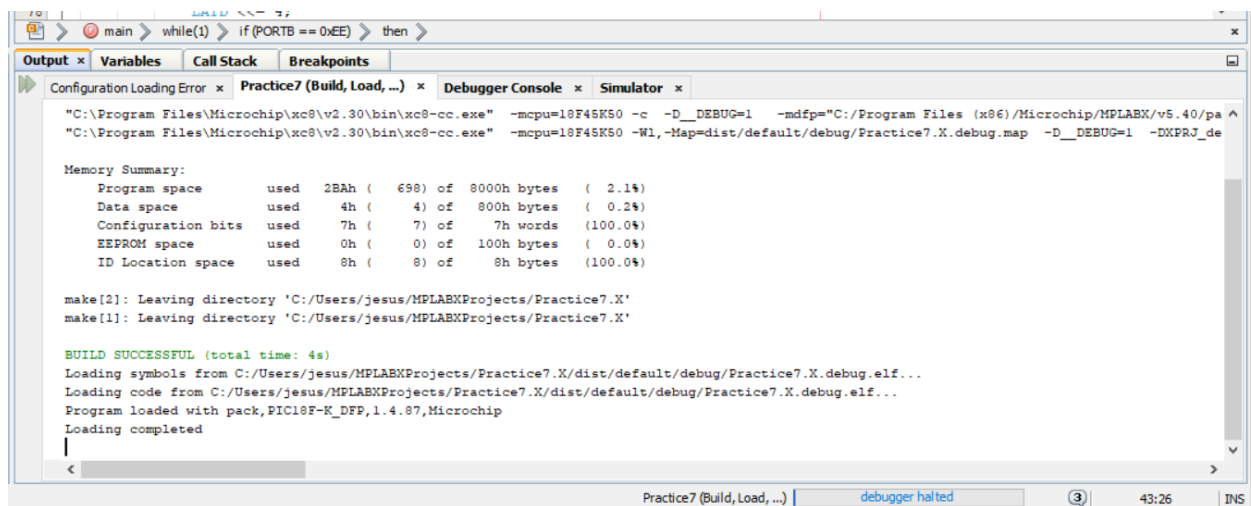
Source History

```
62     } else if (PORTB == 0x7D) {  
63         LATD <= 4;  
64         LATD |= 0x00;  
65     } else if (PORTB == 0xEB) {  
66         LATD <= 4;  
67         LATD |= 0x03;  
68     } else if (PORTB == 0xDB) {  
69         LATD <= 4;  
70         LATD |= 0x06;  
71     } else if (PORTB == 0xBB) {  
72         LATD <= 4;  
73         LATD |= 0x09;  
74     } else if (PORTB == 0x7B) {  
75         LATD <= 4;  
76         LATD |= 0x0E;  
77     } else if (PORTB == 0xE7) {  
78         LATD <= 4;  
79         LATD |= 0x0A;  
80     } else if (PORTB == 0xD7) {  
81         LATD <= 4;  
82         LATD |= 0x0B;  
83     } else if (PORTB == 0xB7) {  
84         LATD <= 4;  
85         LATD |= 0x0C;  
86     } else if (PORTB == 0x77) {  
87         LATD <= 4;  
88         LATD |= 0x0D;  
89     }  
90  
91 }  
92  
93  
94 return;  
95 }
```

Posteriormente ya con los casos identificados mandamos llamar distintas acciones según el caso con una serie de ifs y else ifs, que son los que ejecutan el cambio según lo que se introduce al teclado matricial. (cada uno lleva un corrimiento debido a como se nos especifica que deben ir identificados los bits del programa).

Funcionamiento:

Por desgracia sin un PIC18 ni un simulador como Proteus para poder probar el funcionamiento correcto del programa, lo único que podemos hacer para verificarlo es comprobar que el programa haga un debug exitoso sin presentar errores y que los puertos iniciales se configuren correctamente.



```

12
13 void portInit(){
14
15     ANSELB = 0; // Configuramos todos los pines como digitales
16     TRISB = 0b11110000; // Pines del 4 al 7 como salida y el resto entrada
17     ANSELB = 0; // Todos los pines como digitales
18     TRISD = 0b00000000; // Todos los pines como entrada
19 }
20
21 void main(void) {
22     portInit();
23     unsigned short columnas = 0;
24
25     while(1){

```

main > while(1) > if(PORTB == 0xEE) > else if(PORTB == 0xBD) >

Output		SFRs x				
Address /	Name	Hex	Decimal	Binary	Char	
F90	CCPR2	0x0000	0	00000000 00000000	'..'	
F90	CCPR2L	0x00	0	00000000	','	
F91	CCPR2H	0x00	0	00000000	','	
F92	TRISA	0xFF	255	11111111	'ÿ'	
F93	TRISB	0xF0	240	11110000	'ð'	
F94	TRISC	0xF7	247	11110111	'÷'	
F95	TRISD	0x00	0	00000000	','	
F96	TRISE	0x87	135	10000111	'ï'	
F97	CCP2CON	0x00	0	00000000	','	
F98	CM1CON0	0x08	8	00001000	','	
F99	CM2CON0	0x08	8	00001000	','	
F9A	CM2CON1	0x00	0	00000000	','	
F9B	OSCTUNE	0x00	0	00000000	','	
F9C	HLVDCON	0x00	0	00000000	','	
F9D	PIEL	0x00	0	00000000	','	
F9E	PIR1	0x00	0	00000000	','	
F9F	IPR1	0xFF	255	11111111	'ÿ'	

Memory SFRs Format Individual