

Microcontroladores

Semana 2

1

Preguntas previas

- Con respecto a los materiales
 - PIC18F45K50 (S/35 – S/45)
 - Software: MPLAB X v6.05 / XC8 Assembler v2.36
- Datasheet PIC18F45K50
- En el PIC18F45K50
 - Los puertos son entradas analógicas en un PoR (Power on Reset) para lo cual hay que configurarlos como digitales usando los registros ANSELx
 - Colocar una primera instrucción: movlb OFH para que se graben los valores en los registros SFR

2

Agenda

- Arquitectura del CPU de los PIC18xxx
- Manipulación de la memoria de programa y de la memoria de datos
 - El contador de programa
 - El puntero de tabla
 - Punteros de acceso a la memoria de datos
- Manejo de los puertos de E/S
- Soporte del lenguaje XC8 PIC Assembler
- Interface a displays de siete segmentos

3

Resolución de la asignación

1. Desarrollar una aplicación con el microcontrolador PIC18F45K50 el cual tenga 8 LEDs y tres pulsadores activos en alto, la acción de cada pulsador hará un ciclo de visualización de una sola vez según la siguiente tabla, el tiempo de transición será de un segundo aproximadamente.

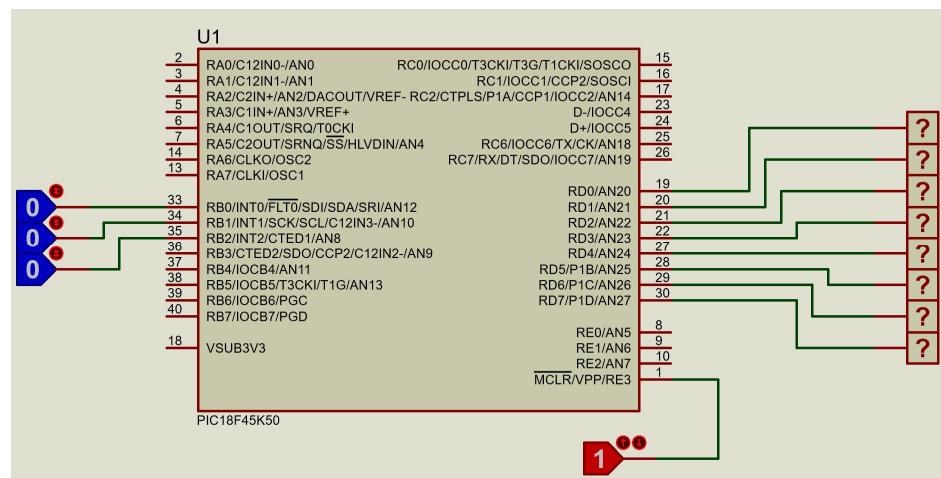
Pulsador 1 accionado								
	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LEDO
t0								
t1								
t2								
t3								
t4								
t5								
t6								

Pulsador 2 accionado								
	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LEDO
t0								
t1								
t2								
t3								

4

Resolución de la asignación

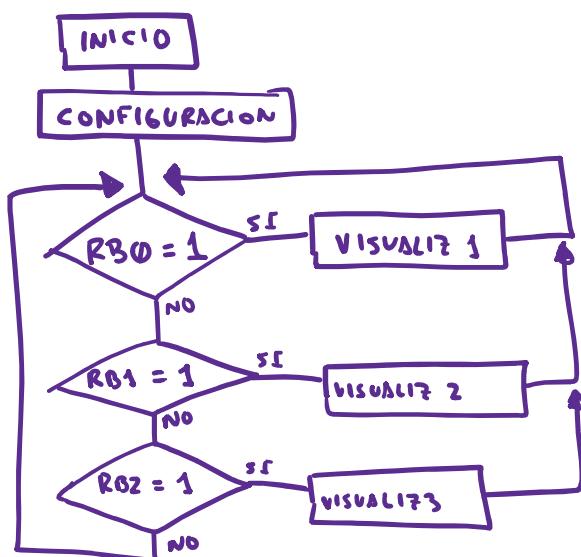
- Circuito



5

Resolución de la asignación

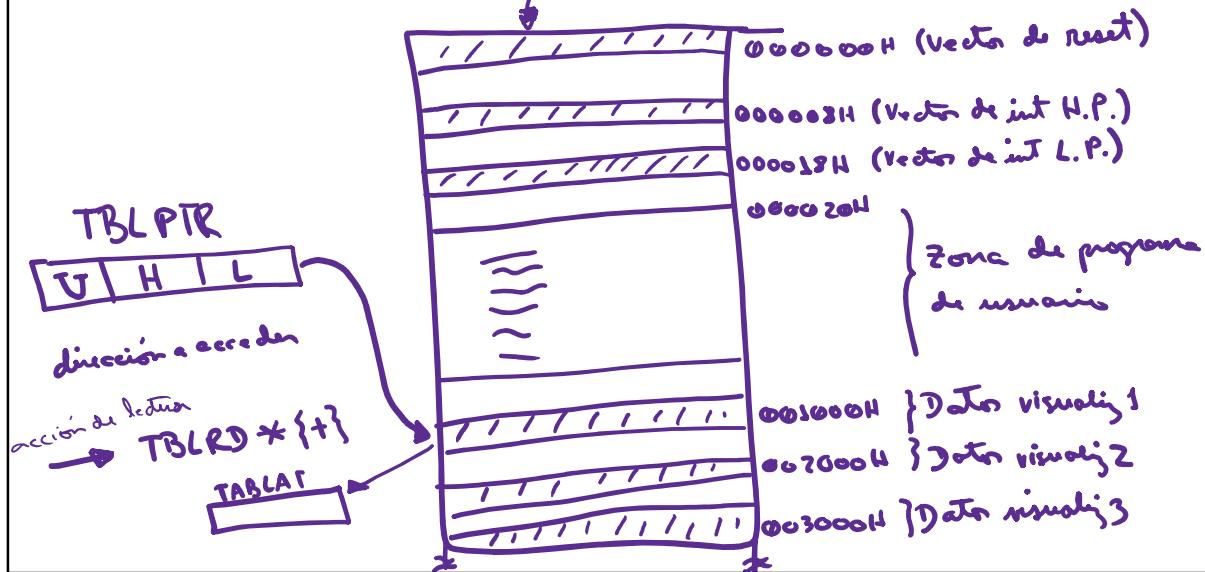
- Diagrama de flujo



6

Resolución de la asignación

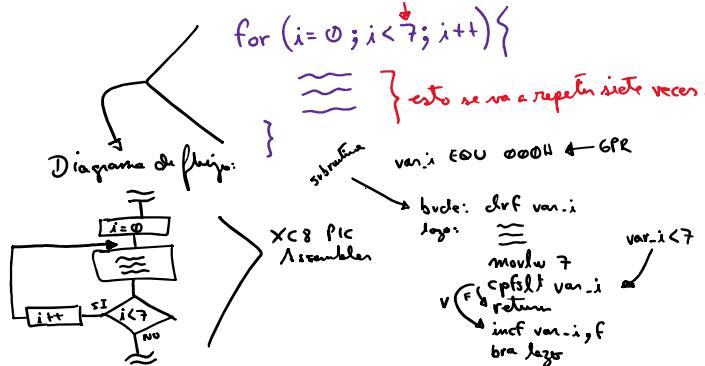
- Estructura de la memoria de programa según programa realizado



7

Resolución de la asignación

- Notas: Estamos apreciando luego de pasar al uso del TBLPTR que hay mucho código redundante (que se repite varias) haciendo que ocupe mas espacio en la memoria de programa.
- Recordando la estructura de repetición controlada FOR:



CPFSLT	Compare f with W, skip if f < W
Syntax:	CPFSLT f {,a}
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	(f) – (W). skip if (f) < (W) (unsigned comparison)
Status Affected:	None
Encoding:	0110 000a ffff ffff
Description:	Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction. If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
Words:	1
Cycles:	1(2)
Note:	Three cycles if skip and followed by a 2-word instruction

8

Resolución de la asignación

- Bits de configuración

Configuration Bits *					
	Address	Name	Value	Field	Option
		CFUDIV	0	NOCCLKDIV	CPU System Clock Postscaler
		LS48MHZ	0	SYS24X4	Low Speed USB mode with 48 MHz system clock
300001	CONFIG1R 25	-	-	-	-
		FOSC	9	INTOSCCLWD	Oscillator Selection
		FCLEN	1	ON	Primary Oscillator Shutdown
		FCMER	0	OFF	Fail-Safe Clock Monitor
		IESO	0	OFF	Internal/External Oscillator Switchover
300002	CONFIG2L 55	-	-	-	-
		nPWREN	0	ON	Power-up Timer Enable
		BOREN	0	OFF	Brown-out Reset Enable
		BORV	3	190	Brown-out Reset Voltage
		nLPBOR	1	OFF	Low-Power Brown-out Reset
300003	CONFIG2H 3C	-	-	-	-
		WDTEN	0	OFF	Watchdog Timer Enable bits
		WDTPS	F	32768	Watchdog Timer Postscaler
300005	CONFIG3H D1	-	-	-	-
		CCP2MX	1	RC1	CCP2 MUX bit
		PBADDEN	0	OFF	PORB A/D Enable bit
		T3CMX	1	RC0	Timer3 Clock Input MUX bit
		SDOMX	1	RB3	SDO Output MUX bit
		MCLR	1	ON	Master Clear Reset Pin Enable
300006	CONFIG4L 81	-	-	-	-
		STVREN	1	ON	Stack Full/Underflow Reset
		LVP	0	OFF	Single-Supply ICSP Enable bit
		ICPRT	0	OFF	Dedicated In-Circuit Debug/Programming Port Enable
		XINST	0	OFF	Extended Instruction Set Enable bit

9

Resolución de la asignación

- 4MHz desde el oscilador interno:

REGISTER 3-1: OSCCON: OSCILLATOR CONTROL REGISTER						
R/W-0	R/W-0	R/W-1	R/W-1	R-q	R-0	R/W-0
IDLEN		IRCF<2:0>		OSTS ⁽¹⁾	HFIOPS	SCS<1:0>
						bit 0
bit 7						bit 0
Legend:						
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'		q = depends on condition		
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared		x = Bit is unknown		
bit 6-4	IRCF<2:0>: Internal RC Oscillator Frequency Select bits					
	111	= HFINTOSC – (16 MHz)				
	110	= HFINTOSC/2 – (8 MHz)				
	101	= HFINTOSC/4 – (4 MHz)				
	100	= HFINTOSC/8 – (2 MHz)				
	011	= HFINTOSC/16 – (1 MHz) ⁽²⁾				
	010	= HFINTOSC/32 – (500 kHz)				
	001	= HFINTOSC/64 – (250 kHz)				
	If INTSRC = '1':					
	000	= HFINTOSC/512 – (31.25 kHz)				
	If INTSRC = '0':					
	000	= INTRC – (31.25 kHz)				
bit 3	OSTS: Oscillator Start-up Time-out Status bit					
	1	= Device is running from the clock defined by FOSC<3:0> of the CONFIG1H register				
	0	= Device is running from the internal oscillator (HFINTOSC or INTRC)				
bit 2	HFIOPS: HFINTOSC Frequency Stable bit					
	1	= HFINTOSC frequency is stable				
	0	= HFINTOSC frequency is not stable				
bit 1-0	SCS<1:0>: System Clock Select bit					
	1x	= Internal oscillator block				
	01	= Secondary (SOSC) oscillator				
	00	= Primary clock (determined by FOSC<3:0> in CONFIG1H).				

10

Resolución de la asignación

- Valores en hexadecimal de lo que se va a enviar hacia el Puerto con LEDs

VISUALIZ1

Pulsador 1 accionado							
03H	06H	0CH	18H	30H	60H	0COH	
t0							
t1							
t2							
t3							
t4							
t5							
t6							
03H	06H	0CH	18H	30H	60H	0COH	
03H	0CH	30H	C0H				

Pulsador 2 accionado							
03H	0CH	30H	C0H				
t0							
t1							
t2							
t3							
03H	0CH	30H	C0H				

VISUALIZ2

```

ORG 001000H ;Zona de visualiz1
datos1 DB 03H, 06H, 0CH, 18H, 30H, 60H, 0COH

ORG 002000H ;Zona de visualizz
datos1 DB 03H, 0CH, 30H, 0COH

```

11

Resolución de la asignación

- Código en XC8 PIC Assembler

```

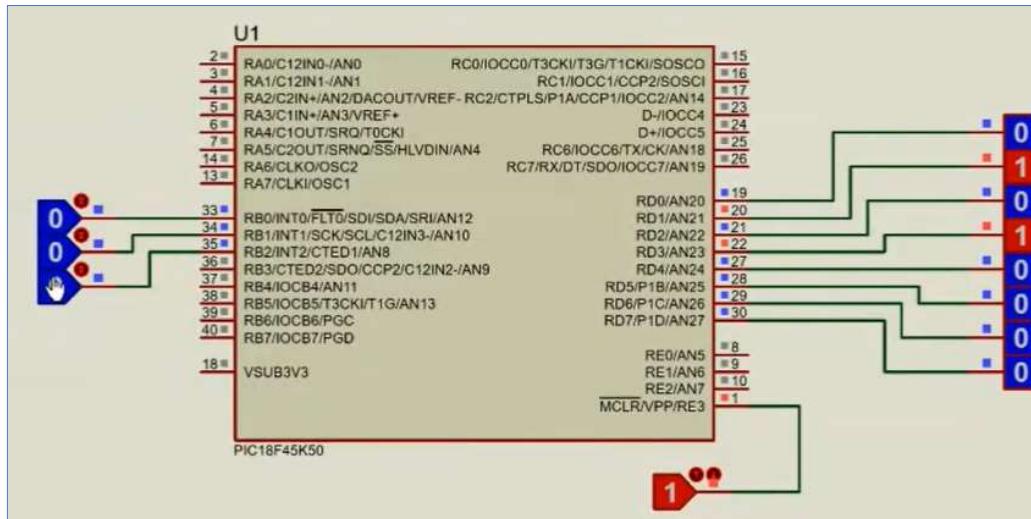
1  PROCESSOR 18F45K50
2  #include "cabecera.inc"
3
4  PSECT principal, class=CODE, reloc=2, abs
5  principal:
6
7  var_a EQU 000H      ;GPR 000H con label var_a
8  var_b EQU 001H      ;GPR 001H con label var_b
9  var_c EQU 002H      ;GPR 002H con label var_c
10 var_for EQU 003H    ;GPR 003H con label var_for
11 var_vis EQU 004H    ;GPR 004 con label var_vis
12
13 ORG 000000H        ;vector de reset
14 bra configuro
15
16 ORG 001000H        ;Zona de visualiz1
17 datos1: DB 03H, 06H, 0CH, 18H, 30H, 60H, 0COH, 00H
18 ORG 002000H        ;Zona de visualizz
19 datos2: DB 03H, 0CH, 30H, 0COH, 00H
20 ORG 003000H        ;Zona de visualiz3
21 datos3: DB 05H, 0AH, 14H, 28H, 50H, 0AOH, 00H
22
23 ORG 000020H        ;zona de programa de usuario
24 configuro:          ;configuraciones de modulos
25  movlb 0FH           ;para acceder a los SFR
26  movlw 52H
27  movwf OSCCON        ;HFINTOSC a 4MHz / Internal oscillator
28  clrf OSCCON2
29  movlw 0FH
30  movwf ANSELB        ;RB2:RB0 como digitales
31  clrf ANSELD        ;todos los pines de RD como digital
32  clrf TRISD          ;todos los pines de RD como salida
33
34         loop:          ;loop principal
35     btfss PORTB, 0
36     bra salto1
37     clrf TBLPTRU
38     movlw 10H
39     movwf TBLPTRH
40     clrf TBLPTRL
41     movlw 7
42     movwf var_vis
43     call sub_visual
44     bra loop
45
46         salto1:        ;salto para el caso de pulsador 1
47     btfss PORTB, 1
48     bra salto2
49     clrf TBLPTRU
50     movlw 20H
51     movwf TBLPTRH
52     clrf TBLPTRL
53     movlw 4
54     movwf var_vis
55     call sub_visual
56     bra loop
57
58         salto2:        ;salto para el caso de pulsador 2
59     btfss PORTB, 2
60     bra loop
61     clrf TBLPTRU
62     movlw 30H
63     movwf TBLPTRH
64     clrf TBLPTRL
65     movlw 6
66
67     movwf var_vis
68     call sub_visual
69     bra loop
70
71         sub_visual:   ;subrutina para visualizar
72     clrf var_for
73     movlw 7
74     movwf var_vis
75     call retardo
76     movf var_vis, w
77     cpfslt var_for
78     return
79
80     incf var_for, f
81     bra lazo
82
83         lazo:          ;lazo principal
84     TBLRD*+
85     movff TABLAT, LATD
86     call retardo
87     movf var_vis, w
88     cpfslt var_for
89     return
90
91     incf var_for, f
92     bra lazo
93
94         retardo:       ;retardo
95     movlw 100
96     movwf var_a
97
98         bucle1:        ;bucle 1
99     movlw 50
100    movwf var_c
101
102    otro1:          ;otro1
103    decfsz var_a, 1
104    bra otro1
105    return
106
107    bucle2:        ;bucle 2
108    movlw 50
109    movwf var_b
110
111    otro2:          ;otro2
112    decfsz var_b, 1
113    bra otro2
114
115    end principal

```

12

Resolución de la asignación

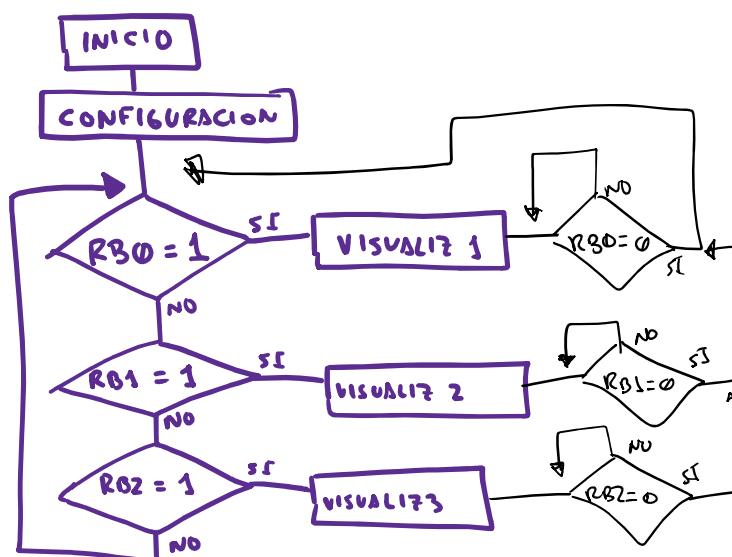
- Ejecución de la simulación



13

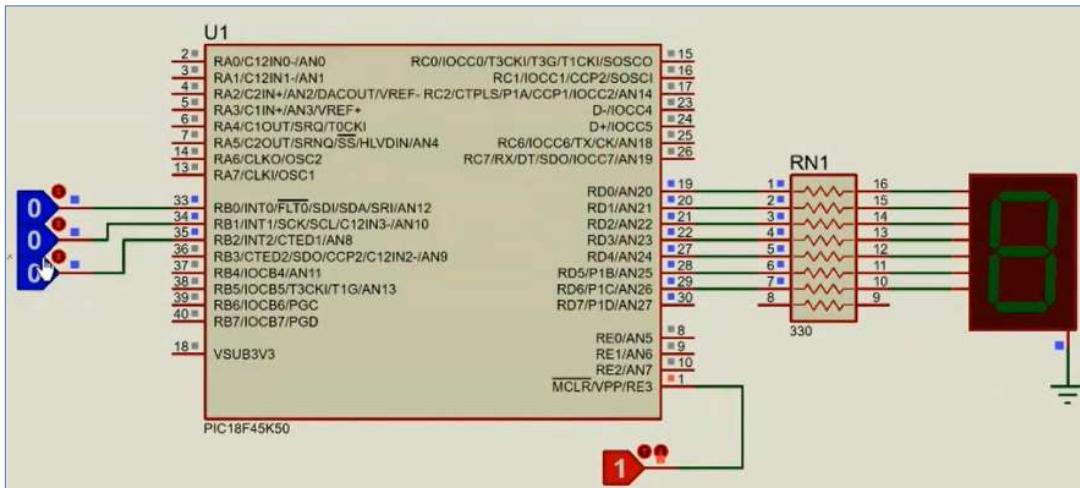
Resolución de la asignación

- Diagrama de flujo con confirmación de soltada de botones



14

Interface con display de siete segmentos



15

Interface con display de siete segmentos

- Obtención de los valores hexadecimales decodificador para el display

The screenshot shows a web-based application for generating seven-segment display patterns. The interface includes:

- DISPLAY:** Shows a green seven-segment display with the character 'P'.
- SEGMENTS STATE:** A table showing the state of each segment (A-G) for the character 'P'. The table is as follows:

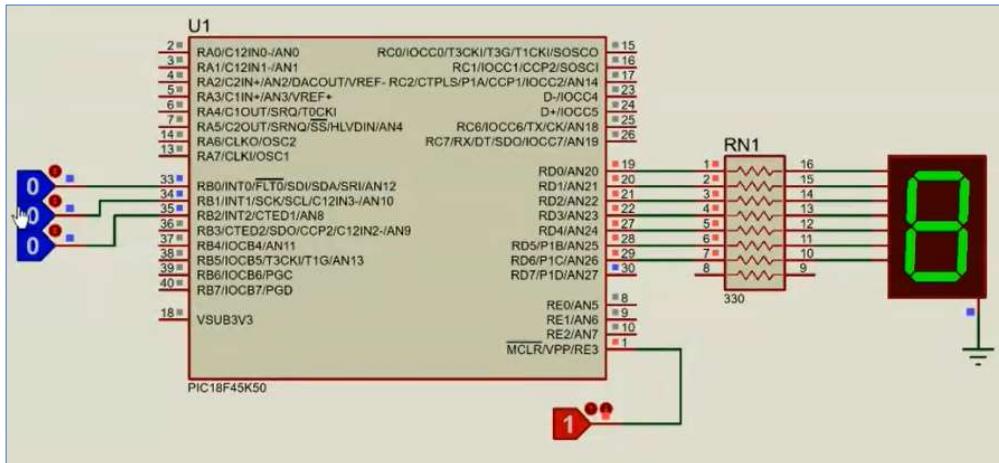
	A	B	C	D	E	F	G
capital: P	1	1	0	0	1	1	1
HEX VALUE:	0x67						
HEX VALUE:	0x73						

- ANIMATE VALUE:** A section with buttons for animating the display, such as "COUNT UP FROM 0 TO 9", "COUNT DOWN FROM 9 TO 0", etc.

16

Interface con display de siete segmentos

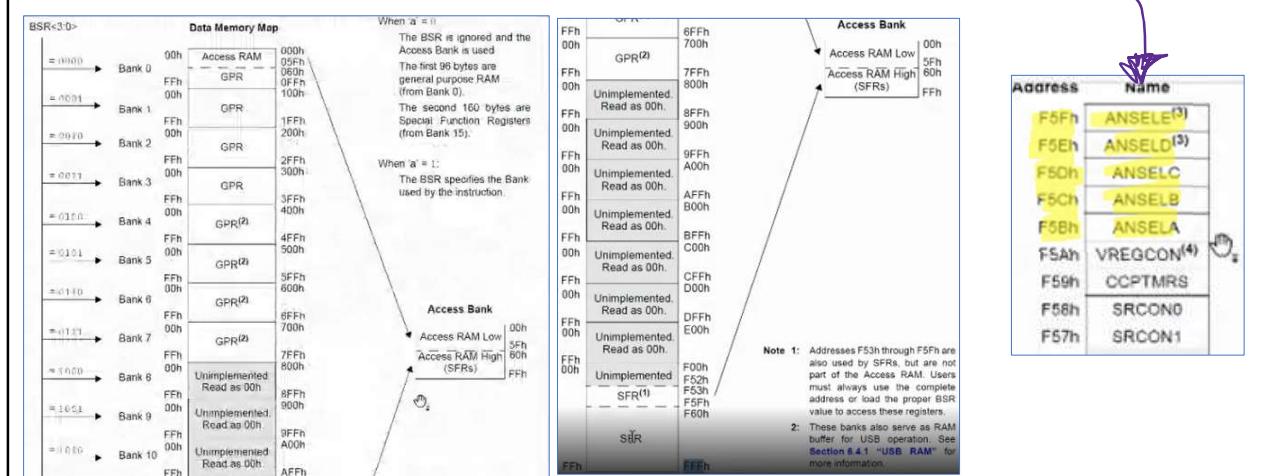
- Ejecución de la simulación



17

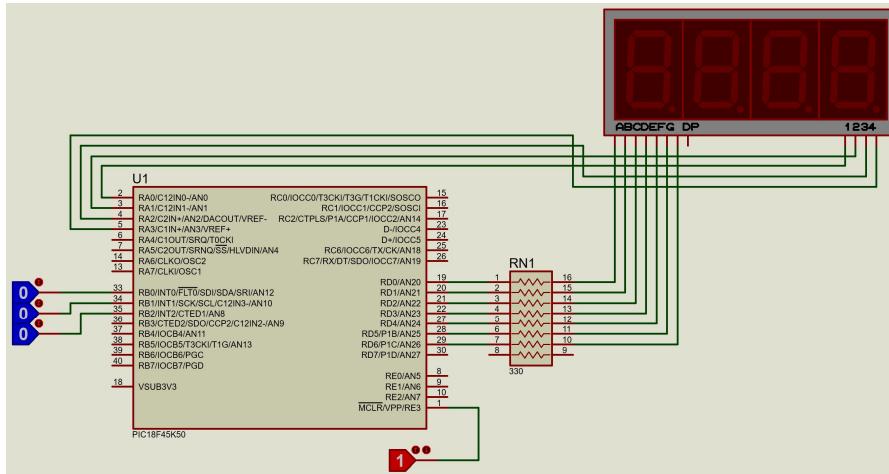
Modo Access Bank

- En este modo de acceso a la memoria de datos toma la mitad de la RAM del Bank 0 comprendido entre 00H-5FH y la parte del SFR comprendido entre F60H y FFFH.
- Tener en cuenta que los registros ANSELx no pueden ser accedidos por Access Bank dado que se encuentran debajo de la dirección F60H



18

Multiplexación de displays de siete segmentos



19

Multiplexación de cuatro dígitos de siete segmentos

```

1      PROCESSOR 18F45K50
2      #include "cabecera.inc"
3
4      PSECT principal, class=CODE, reloc=2, abs
5      principal:
6
7      var_a EQU 000H ;GPR 000H con label var_a
8      var_b EQU 001H ;GPR 001H con label var_b
9      var_c EQU 002H ;GPR 002H con label var_c
10     var_for EQU 003H ;GPR 003H con label var_for
11     var_vis EQU 004H ;GPR 004 con label var_vis
12     data1 EQU 005H
13     data2 EQU 006H
14     data3 EQU 007H
15     data4 EQU 008H
16
17     ORG 000000H ;vector de reset
18     bra configuro
19
20     ORG 001000H ;zona de visualiz1
21     datos1: DB 73H,79H,50H,3EH
22     ORG 002000H ;zona de visualiz2
23     datos2: DB 1EH,3FH,6DH,79H
24     ORG 003000H ;Zona de visualiz3
25     datos3: DB 38H,30H,15H,77H
26
27     ORG 000020H ;zona de programa de usuario
28     configuro: ;configuraciones de modulos
29     movlb OFH ;para acceder a los SFR
30     movlw 52H
31     movwf OSCCON ;HFINTOSC a 4MHz / Internal
32     clrf OSCCON2
33     movlw 0FH
34     movwf ANSEL0 ;RA3:RA0 como digitales
35
36     movwf TRISA ;RA3:RA0 como salidas
37     movlw 0FH
38     movwf LATA ;RA3:RA0 en uno lógico (en alto)
39     movlw 0FSH
40     movwf ANSEL0 ;RB2:RB0 como digitales
41     clrf TRISD ;todos los pines de RD como digitales
42     clrf TBLPTRU ;todos los pines de RD como salidas
43     movlw 10H
44     movwf TBLPTRH
45
46     loop: ;prog1
47     clrf TBLPTRL
48     TBLRD+*
49     movwf TABLAT, data1
50     TBLRD+*
51     movff TABLAT, data2
52     TBLRD+*
53     movwf TABLAT, data3
54     TBLRD+*
55     movwf TABLAT, data4
56     call multiplex
57     bra loop
58
59     multiplex:
60     movwf data1,w
61     movwf LATD
62     bcf LATD,0
63     call nopes
64     bcf LATD,0
65     movwf data2,w
66     movwf LATD
67     bcf LATD,1
68     call nopes
69
70     bsf LATA,1
71     movwf data3,w
72     movwf LATD
73     bcf LATA,2
74     call nopes
75     bsf LATA,2
76     movwf LATD
77     bcf LATA,3
78     call nopes
79     bsf LATA,3
80     return
81
82     nopes:
83     nop
84     nop
85     nop
86     nop
87     nop
88     nop
89     return
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116 end principal

```

20

Multiplexación de cuatro dígitos de siete segmentos

```

1      PROCESSOR 18F45K50          35      clrf TBLPTRU           69      bra loop
2      #include "cabecera.inc"    36
3
4      PSECT principal, class=CODE, reloc=2, abs   37      loop:                      ;programa de 1
5      principal:                38          btfss PORTB, 0        70      multiplex:
6
7      data1 EQU 005H             39          bra salt01           71      movwf data1,w
8      data2 EQU 006H             40          movlw 10H            72      movwf LATD
9      data3 EQU 007H             41          movwf TBLPTRH         73      bcf LATA,0
10     data4 EQU 008H             42          clrf TBLPTRL         74      call nopes
11
12     ORG 000000H              43          bra salida           75      bsf LATA,0
13     bra configuro            44          salt01:               76      movwf data2,w
14     ORG 001000H              45          btfss PORTB, 1        77      movwf LATD
15     datos1: DB 73H,79H,50H,3EH 46          bra salt02           78      bcf LATA,1
16     ORG 002000H              47          movlw 20H            79      call nopes
17     datos2: DB 1EH,3FH,6DH,79H 48          movwf TBLPTRH         80      bsf LATA,1
18     ORG 003000H              49          clrf TBLPTRL         81      movwf data3,w
19     datos3: DB 38H,30H,15H,77H 50          bra salida           82      movwf LATD
20     ORG 000020H              51          salt02:               83      bcf LATA,2
21     configuro:               52          btfss PORTB, 2        84      call nopes
22     movlw 0FH                 53          bra loop             85      bsf LATA,2
23     movlw 52H                 54          movlw 30H            86      movwf data4,w
24     movwf OSCCON              55          movwf TBLPTRH         87      movwf LATD
25     clrf OSCCON2             56          clrf TBLPTRL         88      bcf LATA,3
26     movlw 0FOH                57          bra salida           89      call nopes
27     movwf ANSELA              58          salida:               90      bsf LATA,3
28     movwf TRISA                59          TBLRD*+              91      return
29     movlw 0FH                 60          movff TABLAT, data1  92
30     movwf LATA                61          TBLRD*+              93      nopes:
31     movlw 0FBH                62          movff TABLAT, data2  94
32     movwf ANSELB              63          TBLRD*+              95      nop
33     clrf ANSELB              64          movff TABLAT, data3  96      nop
34     clrf TRI3D                65          TBLRD*+              97      nop
35     call multiplex           66          movff TABLAT, data4  98      nop
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102

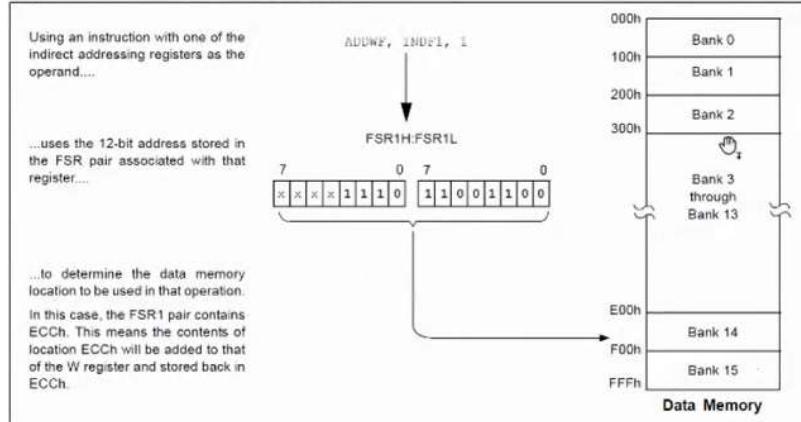
```

21

Arquitectura del CPU de los PIC18

- Direccionamiento indirecto a memoria de datos usando FSRx e INDFx

FIGURE 6-7: INDIRECT ADDRESSING



22

Manipulación de las memorias en el PIC18

- Tratamiento de datos:

- Direccionamiento directo:

movwf TRISB

movwf F93H

movwf dato1

;dato1 esta asociado con una posición en GPR

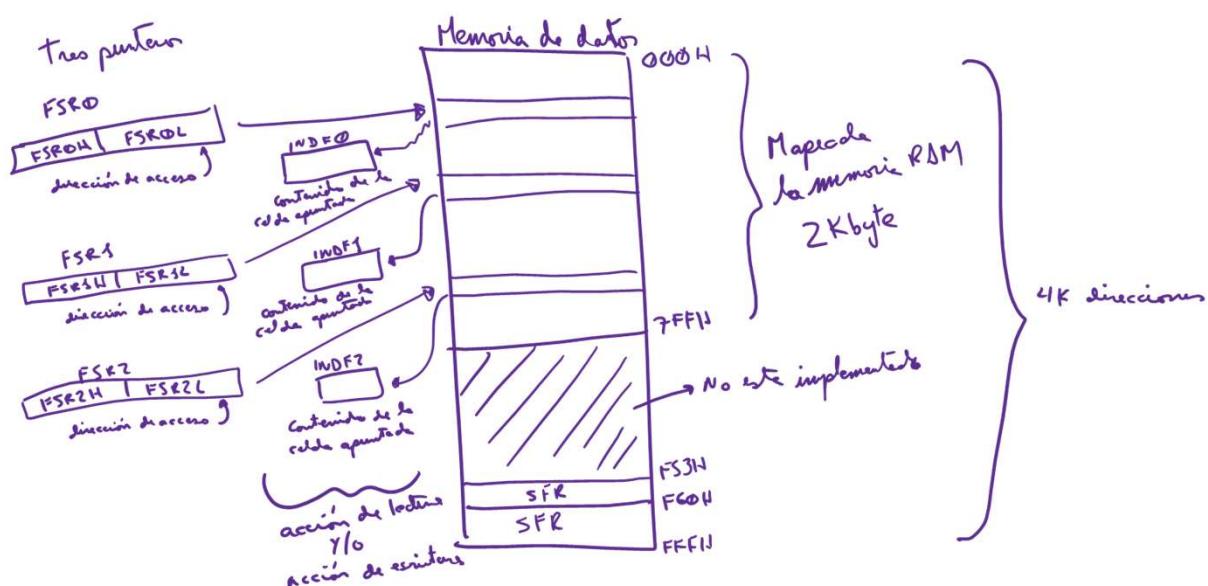
movwf 000H

dato1 EQU 000H

- Direccionamiento indirecto: uso de punteros

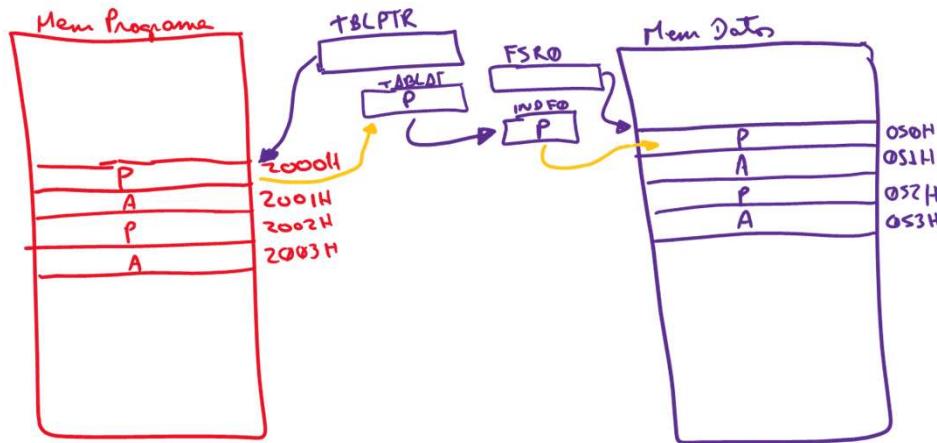
23

Direccionamiento indirecto detalle



24

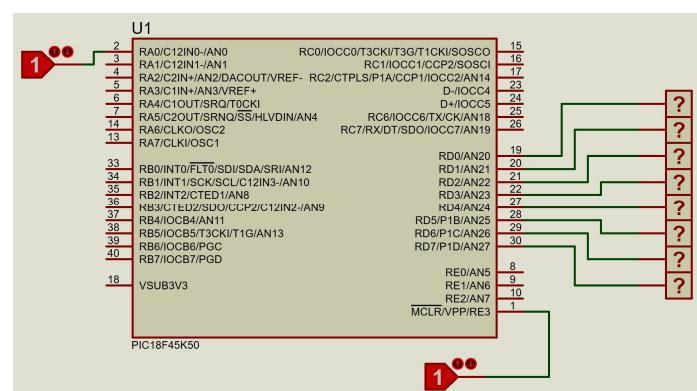
Transferir datos de la memoria de programa hacia la memoria de datos empleando punteros



25

Ejemplo de direccionamiento a memoria de datos

- Se verificará el funcionamiento de los modos directo e indirecto para acceder a la memoria de datos.
- La entrada RA0 servirá para seleccionar el dato a salir por RD, si es cero se empleará el direccionamiento directo para escribir 55H en RD y si es uno se empleará el direccionamiento indirecto para escribir AAH en RD



26

Ejemplo de direccionamiento a memoria de datos

```

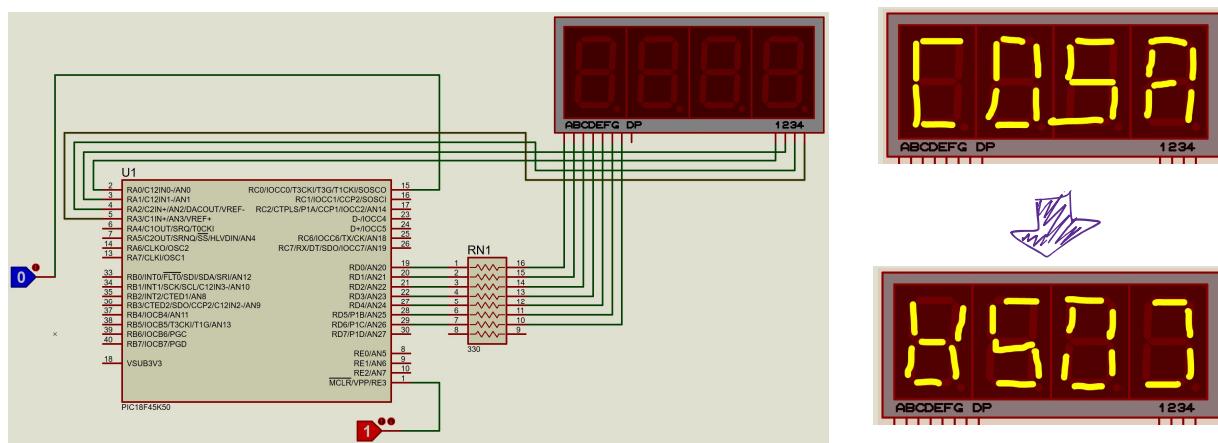
19    loop:
20        btfss PORTA, 0
21        bra escero
22        bra esuno
23        ;Ejercicio de direccionamiento directo
24        escero:
25            movlw 55H           ;cargo literal 55H a Wreg
26            movwf 000H          ;muevo contenido de Wreg a GPR 000H
27            movff 000H, LATD   ;muevo contenido de GPR 000H hacia registro LATD
28            bra loop
29
30        ;Ejercicio de direccionamiento indirecto
31        esuno:
32            lfsr 0, 000H         ;asignando dirección 000H hacia puntero FSR0
33            movlw OAAH          ;cargo literal 55H a Wreg
34            movwf INDF0          ;muevo contenido de Wreg hacia INDF0
35            movff INDF0, LATD   ;muevo contenido de INDF0 hacia LATD
36            bra loop
37        end principal
38
1     PROCESSOR 18F45K50
2     #include "cabecera.inc"
3
4     PSECT principal, class=CODE, reloc=2, abs
5     principal:
6
7     ORG 000000H      ;vector de reset
8     bra configuro
9
10    ORG 000020H     ;zona de programa de usuario
11    configuro:
12        movlb 0FH          ;configuraciones de modulos
13        movlw 52H          ;para acceder a los SFR
14        movwf OSCCON, 0    ;HFINTOSC a 4MHz / Internal oscillator block
15        clrf OSCCON2, 0
16        clrf TRISD         ;RD como salidas
17        bcf ANSELA, 0       ;RA0 como entrada digital

```

27

Ejercicio de manipulación de datos en 7 segmentos

- En un circuito basado en display multiplexado de cuatro dígitos cátodo común el cual permita visualizar una palabra tanto normal como de cabeza a partir de una sola cadena de datos.



28

Ejercicio de manipulación de datos en 7 segmentos

- Lo primero es obtener los datos hexadecimales de cada letra decodificada de la palabra en siete segmentos



$C = 39H$
 $O = 3FH$
 $S = 6DH$
 $A = 77H$

29

Ejercicio de manipulación de datos en 7 segmentos

```

1  PROCESSOR 18F45K50
2  #include "cabecera.inc"
3
4  PSECT principal, class=CODE, reloc=2, abs
5  principal:
6
7  data1 EQU 005H
8  data2 EQU 006H
9  data3 EQU 007H
10 data4 EQU 008H
11
12    ORG 000000H      ;vector de reset
13    bra configuro
14    ORG 001000H      ;Zona de palabra COSA
15  datos1: DB 39H,3FH,6DH,77H
16
17    ORG 000020H      ;zona de programa de usuario
18  configuro:          ;configuraciones de modulos
19    movlb 0FH           ;para acceder a los SFR
20    movlw 52H
21    movwf OSCCON, 0     ;HFINTOSC a 4MHz / Internal oscillator block
22    clrf OSCCON2, 0
23    movlw 0FOH
24    movwf ANSELA, 1     ;RA3:RA0 como digitales
25    movwf TRISA, 0       ;RA3:RA0 como salidas
26    movlw 0FH
27    movwf LATA, 0        ;RA3:RA0 en uno lógico (en alto)
28    bcf ANSEL0, 0, 1     ;RC0 como entrada digital
29    clrf ANSELD, 1       ;todos los pines de RD como digitales
30    clrf TRISD, 0        ;todos los pines de RD como salidas
31    clrf TBLPTRU, 0
32
33    loop:
34      movlw 10H
35      movwf TBLPTRH, 0
36      clrf TBLPTRL, 0
37      bra salida
38
39    salida:
40      TBLRD*+
41      movff TABLAT, data1
42      TBLRD*+
43      movff TABLAT, data2
44      TBLRD*+
45      movff TABLAT, data3
46      TBLRD*+
47      movff TABLAT, data4
48      call multiplex
49      bra loop
50
51    multiplex:
52      movf data1, 0, 0
53      movwf LATD, 0
54      bcf LATA, 0, 0
55      call nopes
56      bsf LATA, 0, 0
57      movf data2, 0, 0
58      movwf LATD, 0
59      bcf LATA, 1, 0
60      call nopes
61      bsf LATA, 1, 0
62      movf data3, 0, 0
63      movwf LATD, 0
64      bcf LATA, 2, 0
65      call nopes
66      bsf LATA, 2, 0
67      movf data4, 0, 0
68      movwf LATD, 0
69      bcf LATA, 3, 0
70      call nopes
71      bsf LATA, 3, 0
72      return
73
74    nopes:
75      nop
76      nop
77      nop
78      nop
79      nop
80      nop
81      return
82
83  end principal

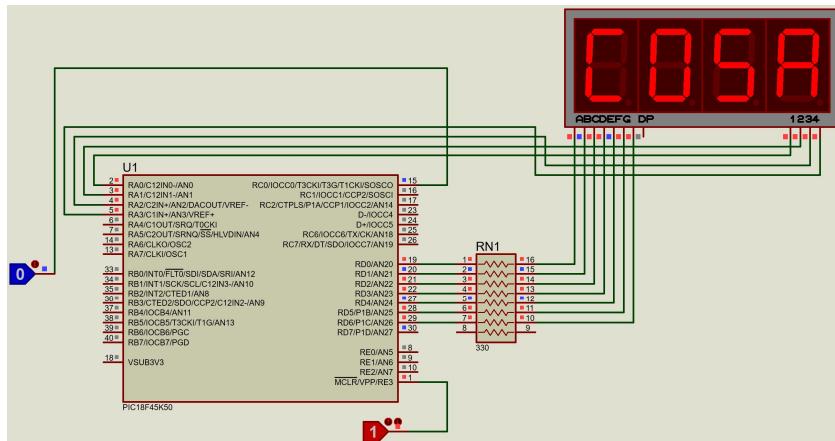
```

- Código fuente inicial que muestra COSA

30

Ejercicio de manipulación de datos en 7 segmentos

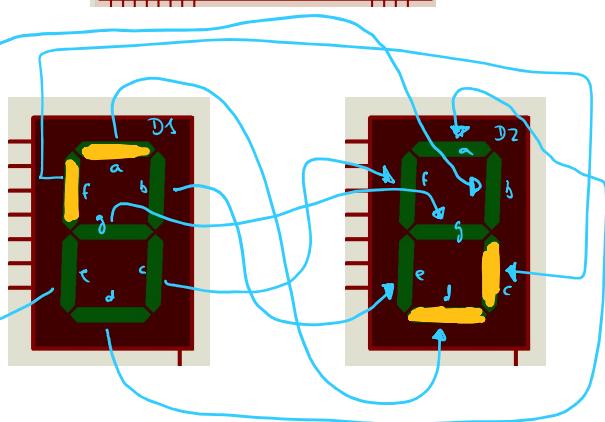
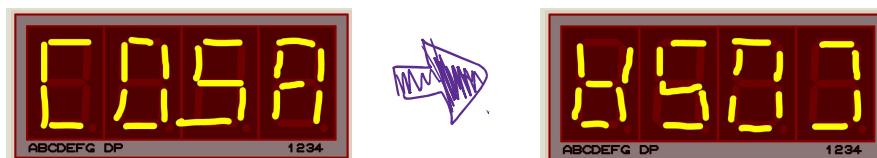
- Simulación inicial solo mostrando COSA



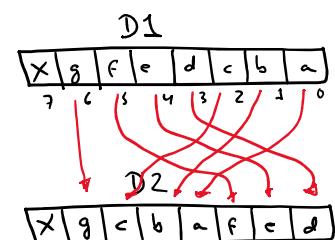
31

Ejercicio de manipulación de datos en 7 segmentos

- Algoritmo que me permita poner de cabeza el mensaje



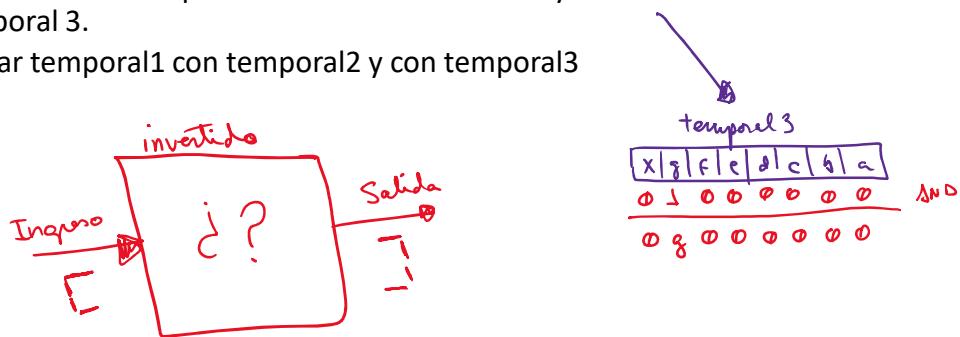
D1	D2
a	d
b	e
c	f
d	g
e	
f	
g	



32

Ejercicio de manipulación de datos en 7 segmentos

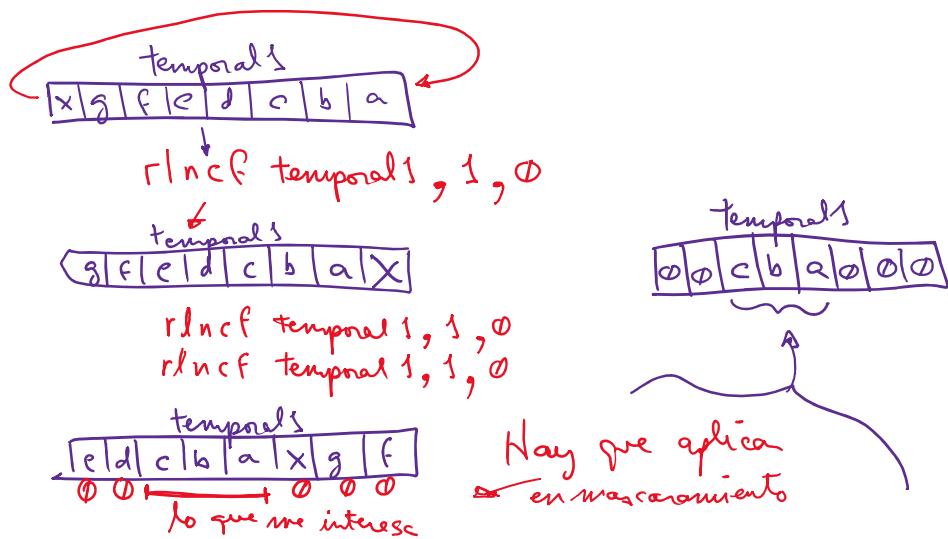
- Procedimiento de desarrollo del algoritmo:
 - Los tres bits menos significativos (bit2, bit1 y bit0) se desplazan tres bits a la izquierda y se almacena el resultado en un registro temporal1.
 - Bit5, bit4 y bit3 se desplazan tres bits a la derecha y se almacena el resultado en temporal2.
 - Enmascarar bit6: operación AND con valor 40H y almacenar el resultado en temporal 3.
 - Sumar temporal1 con temporal2 y con temporal3



33

Ejercicio de manipulación de datos en 7 segmentos

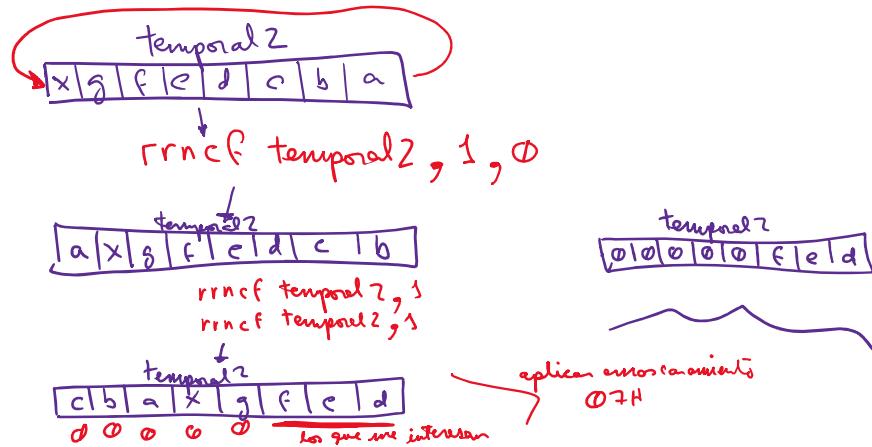
- Instrucción rlncf [registro], d, a



34

Ejercicio de manipulación de datos en 7 segmentos

- Instrucción rrncf [registro], d, a



35

Ejercicio de manipulación de datos en 7 segmentos

```

1      PROCESSOR 18F45K50          37      clrf TBLPTRU, 0           73      TBLRD+*
2      #include "cabecera.inc"    38      movff TABLAT, ingreso   74      movff TABLAT, ingreso
3      PSECT principal, class=CODE, reloc=2, abs 40      loop:                   75      call invertido          75      call invertido
4      principal:               41      movlw 10H                76      movff salida, datal   76      movff salida, datal
5      temporal EQU 000H          42      clrf TBLPTRL, 0          77      call multiplex        77      call multiplex
6      temporal2 EQU 001H         43      btfss PORTC, 0          78      bra loop               78      bra loop
7      temporal3 EQU 002H         44      bra noesuno             79      multiplex:             79      multiplex:
8      temporal4 EQU 003H         45      bra siesuno              80      movff datal, 0, 0     80      movff LATD, 0
9      ingreso EQU 003H          46      noesuno:                 81      movff LATD, 0           81      bcf LATA, 0
10     salida EQU 004H            47      nosesuno:                82      call nopes              82      call nopes
11     datal EQU 005H             48      TBLRD+*                 83      bcf LATA, 0             83      call nopes
12     data2 EQU 006H             49      movff TABLAT, datal    84      bsf LATA, 0             84      call nopes
13     data3 EQU 007H             50      TBLRD+*                 85      movff data2, 0, 0       85      movff LATD, 0
14     data4 EQU 008H             51      movff TABLAT, data2    86      movff LATD, 0           86      bcf LATA, 1, 0
15                           52      TBLRD+*                 87      call nopes              87      call nopes
16                           53      movff TABLAT, data3    88      bcf LATA, 1, 0          88      bcf LATA, 1, 0
17     ORG 000000H ;vector de reset 54      TBLRD+*                 89      call nopes              89      call nopes
18     bra configuro             55      movff TABLAT, data4    90      bsf LATA, 1, 0          90      bsf LATA, 1, 0
19     ORG 001000H ;Zona de palabra COSA 56      movff TABLAT, data4    91      movff data3, 0, 0       91      movff LATD, 0
20 ;datos1: DB 39H,3FH,6DH,77H      57      call multiplex        92      bcf LATA, 0             92      bcf LATA, 2, 0
21 datos1: DB 73H,79H,50H,3EH      58      bra loop               93      call nopes              93      call nopes
22                           59      siesuno:                 94      bsf LATA, 2, 0          94      bsf LATA, 2, 0
23     ORG 000020H ;zona de programa de 59      nosesuno:                95      movff data4, 0, 0       95      movff LATD, 0
24 configuro: ;configuraciones de m 60      TBLRD+*                 96      call nopes              96      call nopes
25     movlb OFH ;para acceder a los S 61      movff TABLAT, ingreso  97      bcf LATA, 3, 0          97      bcf LATA, 3, 0
26     movlw 52H                  62      call invertido          98      call nopes              98      call nopes
27     movwf OSCCON, 0 ;HFINTOSC a 4MHz / In 63      movff salida, data4   99      bsf LATA, 3, 0          99      bsf LATA, 3, 0
28     clrf OSCCON2, 0            64      TBLRD+*                 100     return                 100     return
29     movlw 0FOH                  65      movff TABLAT, ingreso  101     nop                     101     nop
30     movwf ANSELCA, 1 ;RA3:RA0 como digital 66      call invertido          102     nop                     102     nop
31     movwf TRISA, 0 ;RA3:RA0 como 67      movff TABLAT, data3    103     movff salida, data3   103     nopes:                 103     nopes:
32     movlw 0FH                  68      call invertido          104     nop                     104     nop
33     movwf LATA, 0 ;RA3:RA0 en u 69      TBLRD+*                 105     nop                     105     nop
34     bcf ANSELCA, 0, 1 ;RC0 como ent 70      movff TABLAT, ingreso  106     nop                     106     nop
35     clrf ANSELCD, 1 ;todos los pi 71      call invertido          107     movff salida, data2   107     nop                     107     nop
36     clrf TRISD, 0 ;todos los pi 72      movff LATD, 0           108     end principal          108     end principal

```

36

Tabla ASCII

- Sistema de codificación de caracteres alfanuméricos que asigna un número del 0 al 127 a cada letra, número o carácter especial recogidos.
- Ampliamente utilizado en sistemas basados en procesador

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0 000	NUL	(null)	32	20	040	 	Space	64	40	100	@		96	60	140	`	`
1	1 001	SOH	(start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2 002	STX	(start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3 003	ETX	(end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4 004	EOT	(end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5 005	ENQ	(enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6 006	ACK	(acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7 007	BEL	(bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8 010	BS	(backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9 011	TAB	(horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A 012	LF	(NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B 013	VT	(vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C 014	FF	(NF form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D 015	CR	(carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E 016	SO	(shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F 017	SI	(shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10 020	DLE	(data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11 021	DC1	(device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12 022	DC2	(device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13 023	DC3	(device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14 024	DC4	(device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15 025	NAK	(negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16 026	SYN	(synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17 027	ETB	(end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18 030	CAN	(cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19 031	EM	(end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A 032	SUB	(substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B 033	ESC	(escape)	59	3B	073	;	:	91	5B	133	[[123	7B	173	{	{
28	1C 034	FS	(file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D 035	GS	(group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E 036	RS	(record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F 037	US	(unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

Source: www.LookupTables.com

37

Caracteres ASCII en el XC8 PIC Assembler

- Podemos declarar una constante el cual contenga directamente caracteres, el compilador se encargará de trasladar dicho carácter a su correspondiente código en ASCII

```

21      datos1: DB 73H,79H,50H,3EH
22
23      mensaje: DB "MESA"

```

38