

Microcontroladores

Laboratorio Semana 2

Semestre: 2021-1

Profesor: Kalun José Lau Gan

Empiezamos en breve

1

Preguntas previas:

- ¿Por qué estudiamos Assembler?
 - Manipulamos directamente los recursos disponibles en el microcontrolador
 - Mejora la eficiencia en diferentes aspectos: velocidad, consumo energético, disipación de calor, costos reducidos, tamaño final del PCB
- ¿Cuál es el lenguaje mas usado en microcontroladores?
 - El lenguaje C y variantes
- Había leído el capítulo 7 del datasheet sobre el Data EEPROM. ¿Qué es y como se usa?
 - Es una memoria no-volátil que es considerado como un periférico en la arquitectura del microcontrolador PIC18F4550
 - Para acceder a dicha memoria se emplean cuatro registros SFR según se detalla en la hoja técnica:

- EECON1
- EECON2
- EEDATA
- EEADR

2

Agenda

- El flujo de datos en el microcontrolador PIC18F4550
- Instrucciones básicas en MPASM
- Ejemplos básicos con el microcontrolador PIC18F4550 en lenguaje Assembler

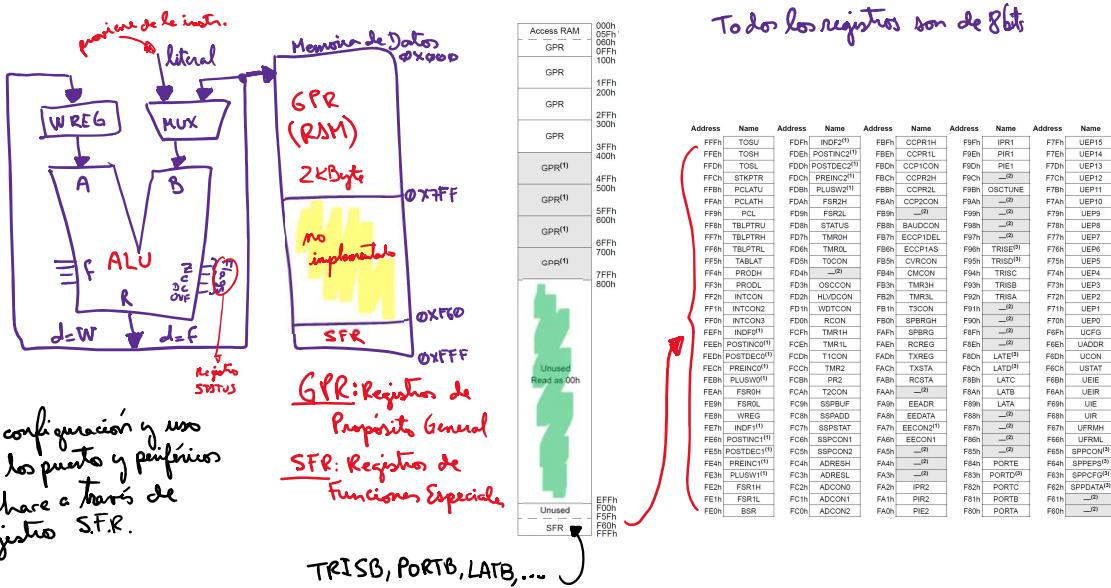
3

Aspectos relacionados con el MPASM (Microchip PIC Assembler)

- El año 2020 Microchip dejó de lado el MPASM para dar lugar al XC8 Assembler (PIC-AS)
- El XC8 Assembler tiene bugs reportados por lo que no se usará.
- MPASM es un lenguaje de bajo nivel (orientado a la máquina), nosotros debemos de conocer primero cómo funciona la máquina para luego hacer que funcione mediante la codificación de un programa en Assembler

4

El flujo de datos en el microcontrolador PIC18F4550



5

Partes de un programa en MPASM

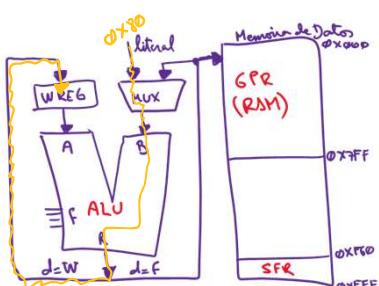
Instrucciones básicas en MPASM

- Instrucciones de movimiento de datos

movlw [literal]

- mover un literal hacia WREG

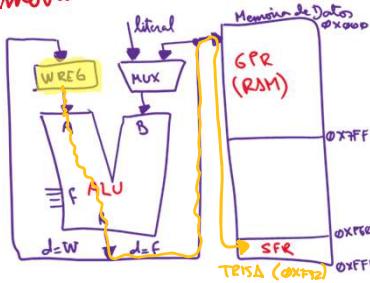
Ej: `movlw 0x80`



movwf [registro]

- mover el contenido de WREG hacia [registro]

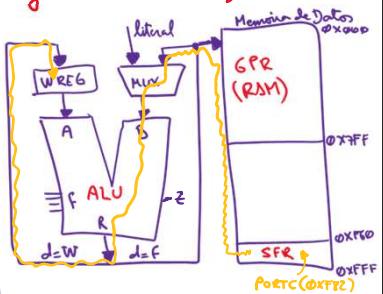
Ej: `movwf TRISA`



movf [registro], d

mover el contenido de [registro] y lo muesre según d.

Ej: `movf PORTC, W`

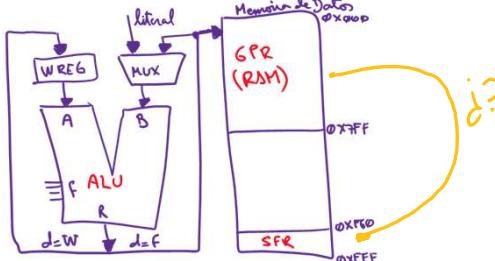


Note: `movf [registro], F` sirve para act. flag

7

Instrucción movff [registro1], [registro2]

- Mueve el contenido de registro1 hacia registro2

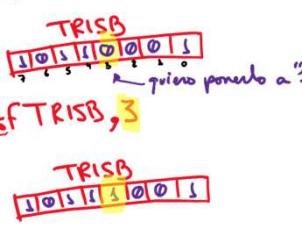


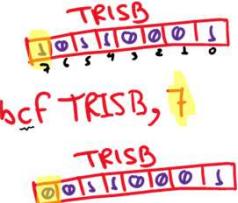
movff PORTB, LATD
instrucción compuesta
movf PORTB, W
movwf LATD

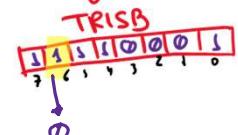
8

Instrucciones básicas en MPASM

- Instrucciones de manipulación de bits en un registro

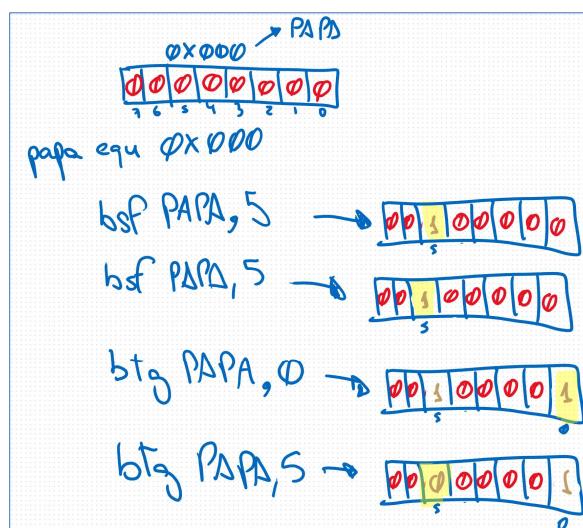
bsf [registro], #bit *posición*
coloca a "1" el #bit de [registro]
Ej:

 bsf TRISB, 3


bcf [registro], #bit
coloca a "0" el #bit de [registro]
Ej:

 bcf TRISB, 7


btg [registro], #bit
aplica complemento al #bit de [registro]
Ej: btg TRISB, 6

 btg TRISB, 6


9

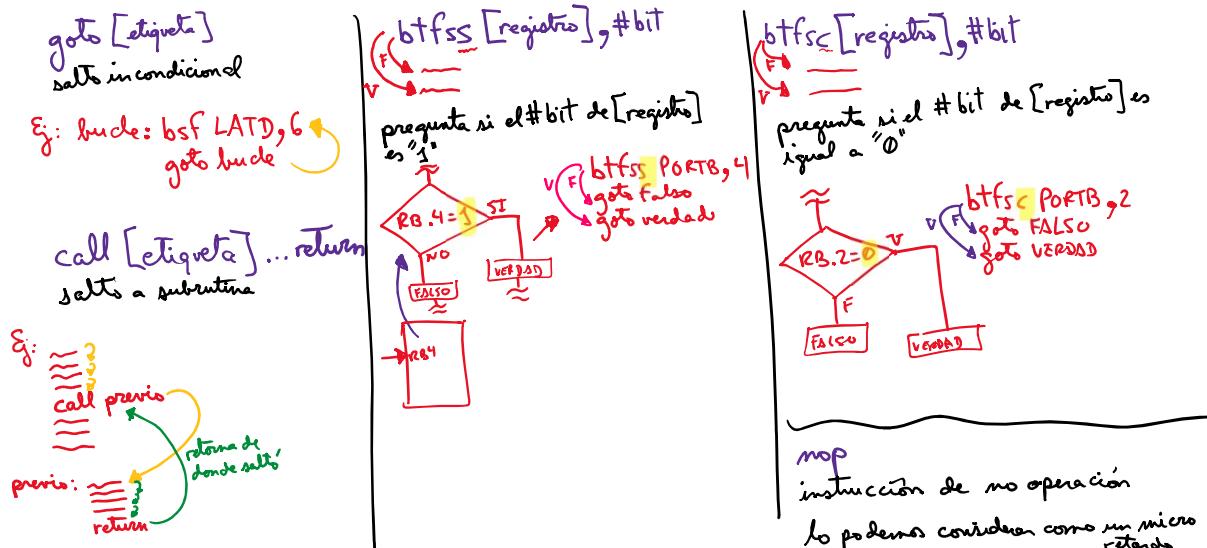
Ejemplo de uso de instrucciones de manipulación de bits en un registro



Access RAM	000h
GPR	00Fh
GPR	060h
GPR	0FFh
GPR	100h
GPR	1FFh
GPR	200h
GPR	2FFh
GPR	300h
GPR(1)	3FFh
GPR(1)	400h
GPR(1)	4FFh
GPR(1)	500h
GPR(1)	5FFh
GPR(1)	600h
GPR(1)	6FFh
GPR(1)	700h
GPR(1)	7FFh
GPR(1)	800h
Unused	
Read as 00h	
Unused	EFFh
SFR	F00h
SFR	F5Fh
SFR	F60h
SFR	FFFh

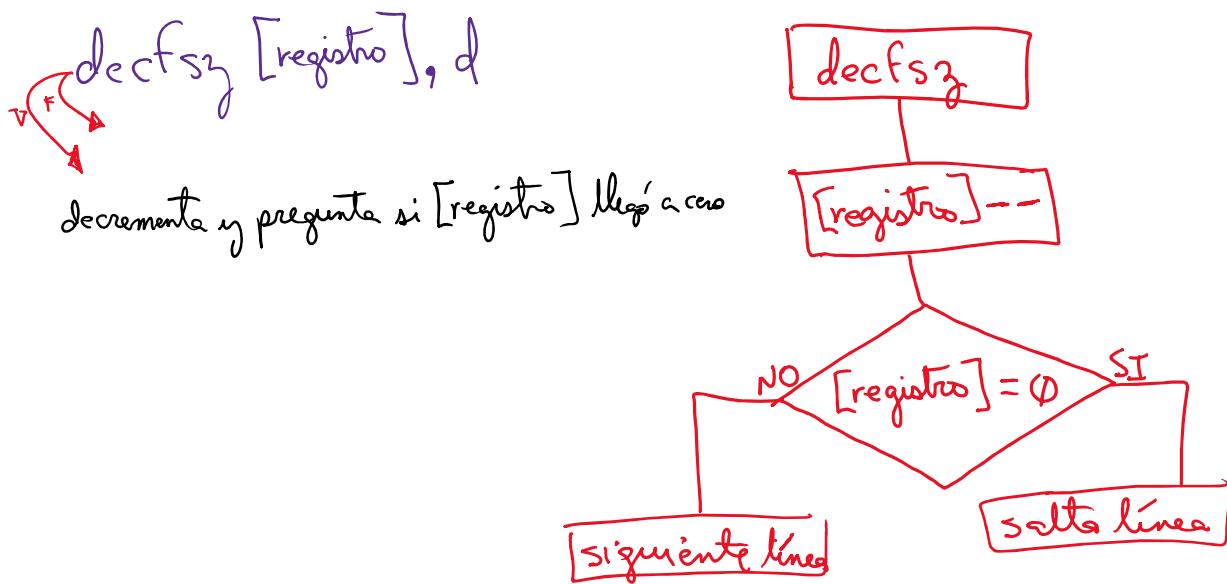
10

Instrucciones básicas en MPASM



11

Instrucciones básicas en MPASM



12

Tiempo de ejecución de las instrucciones en MPASM

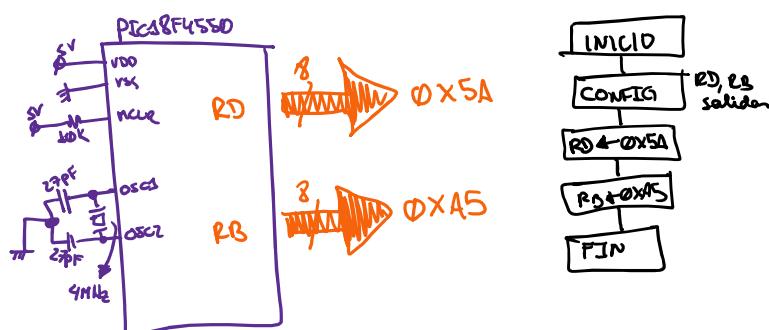
- Se utiliza la siguiente fórmula:

$$t_{opc} = \left(\frac{f_{osc}}{4} \right)^{-1}$$

- Hay instrucciones simples, dobles y especiales (revisar 26.0 de la datasheet)

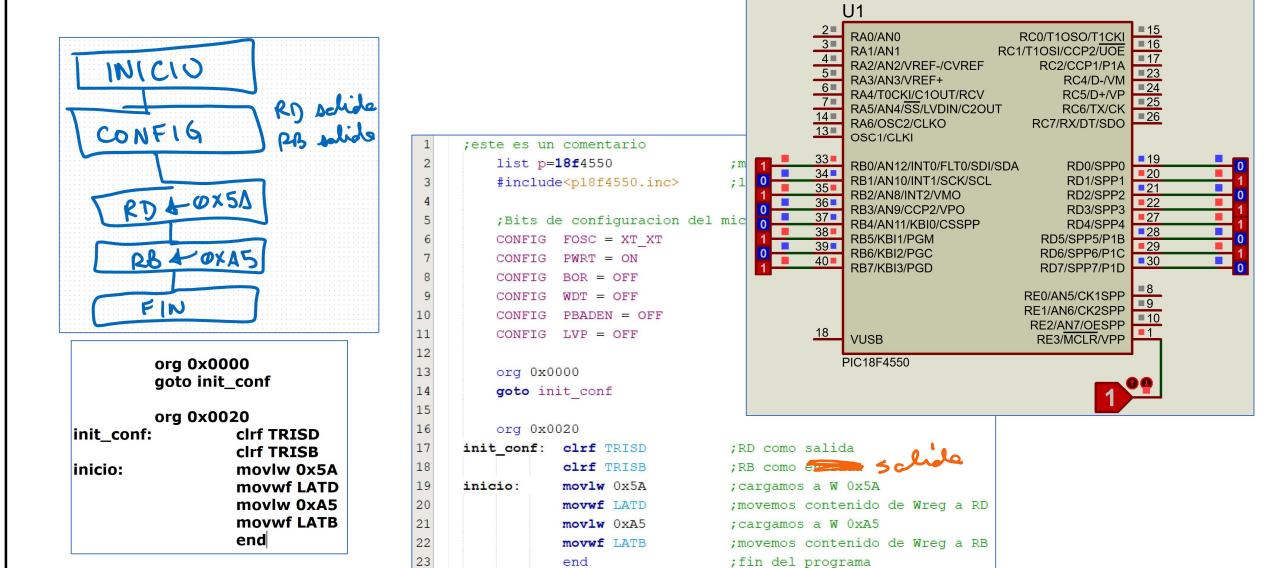
13

Ejemplo: Escribir 0x5A en RD y 0xA5 en RB



14

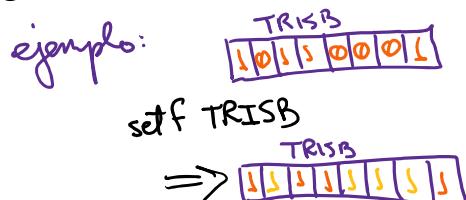
Ejemplo: Enviar un dato 0x5A al puerto RD y un dato 0xA5 al puerto RB



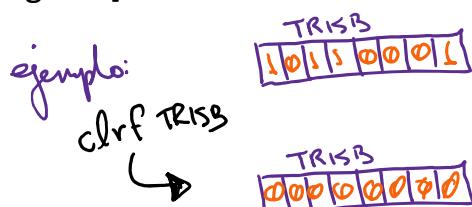
15

Instrucciones setf [registro] y clrf [registro]

- setf [registro] : Coloca todos los bits del registro indicado a uno

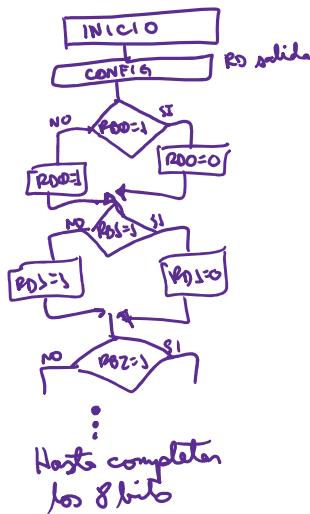
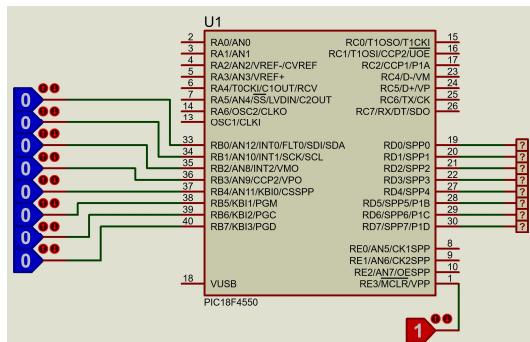


- clrf [registro] : Coloca todos los bits del registro indicado a cero



16

Ejemplo: Recibir un dato en RB y replicarlo en complemento a RD



```

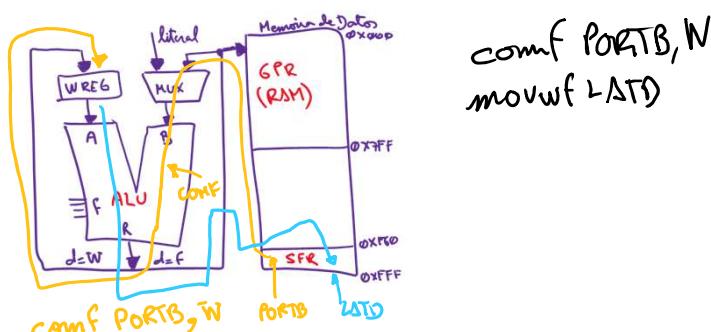
13      org 0x0000
14      goto init_conf
15
16      org 0x0020
17      init_conf: clrf TRISD
18      inicio: btfss PORTE, 0
19      goto falso0
20      bcf LATD, 0
21      goto siguiente0
22      falso0: bcf LATD, 0
23      siguiente0: btfss PORTE, 1
24      goto falsol
25      bcf LATD, 1
26      goto siguiente1
27      falsol: bcf LATD, 1
28      siguiente1: btfss PORTE, 2
29      goto falso2
30      bcf LATD, 2
31      goto siguiente2
32      falso2: bcf LATD, 2
33      siguiente2: btfss PORTE, 3
34      goto falso3
35      bcf LATD, 3
36      goto siguiente3
37      falso3: bcf LATD, 3
38      siguiente3: btfss PORTE, 4
39      goto falso4
40      bcf LATD, 4
41      goto siguiente4
42      falso4: bcf LATD, 4
43      siguiente4: btfss PORTE, 5
44      goto falso5
45      bcf LATD, 5

```

17

Instrucción COMF [registro], d

- Aplica complemento al registro indicado y el resultado lo puede almacenar en Wreg o en el mismo registro



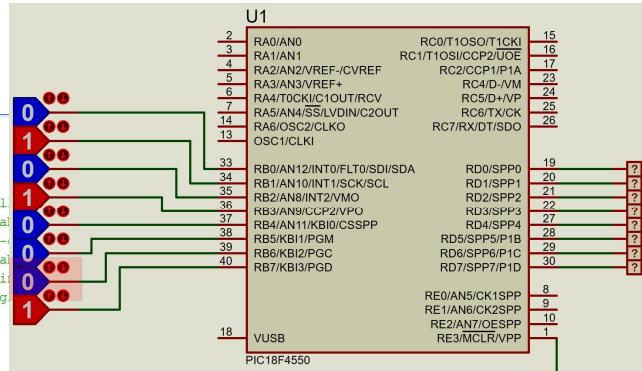
18

Optimización del ejemplo 2 empleando comf

```

2      list p=18f4550      ;modelo de procesador
3      #include<p18f4550.inc> ;libreria de nombre de registros
4
5      ;Bits de configuracion del microcontrolador
6      CONFIG FOSC = XT_XT      ; Oscillator Selection bits (XT oscil
7      CONFIG PWRT = ON        ; Power-up Timer Enable bit (PWRT enal
8      CONFIG BOR = OFF        ; Brown-out Reset Enable bits (Brown-
9      CONFIG WDT = OFF        ; Watchdog Timer Enable bit (WDT disal
10     CONFIG PBADEN = OFF     ; PORTB A/D Enable bit (PORTB4:0) pi
11     CONFIG LVP = OFF        ; Single-Supply ICSP Enable bit (Sing
12
13     org 0x0000
14     goto init_conf
15
16     org 0x0020
17     init_conf: clrf TRISD
18     inicio:  comf PORTB, W      ;RD como salida
19     movwf LATD
20     goto inicio
21     end                      ;fin del programa

```



19

Ejemplo: Titilar un LED conectado en RD2

Hardware:

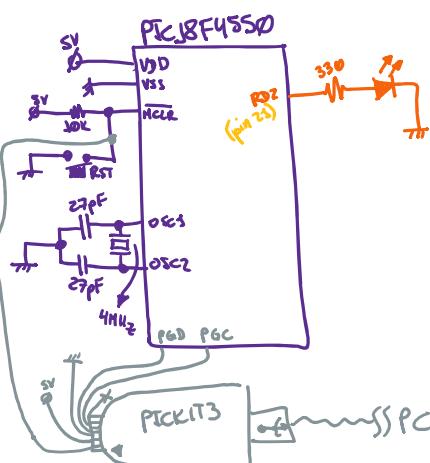
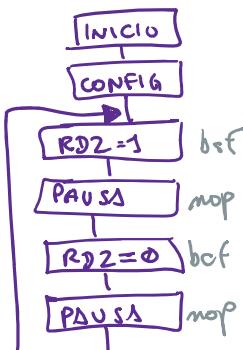


Diagrama de Flujos:



Nota:

$$\text{Tejec instr.} = \left(\frac{\text{Fosc}}{4}\right)^{-1}$$

$$\text{Fosc} = 4\text{MHz}$$

$$\Rightarrow \text{Tejec instr.} = 1\text{ }\mu\text{s}$$

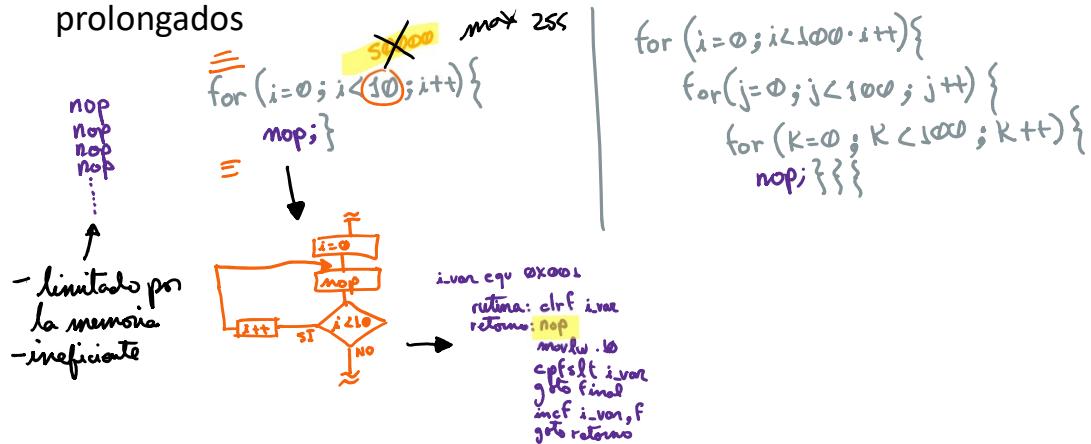
Tenemos que alargar el tiempo
¿Puedo colocar 50000 'nop'?

NO!

20

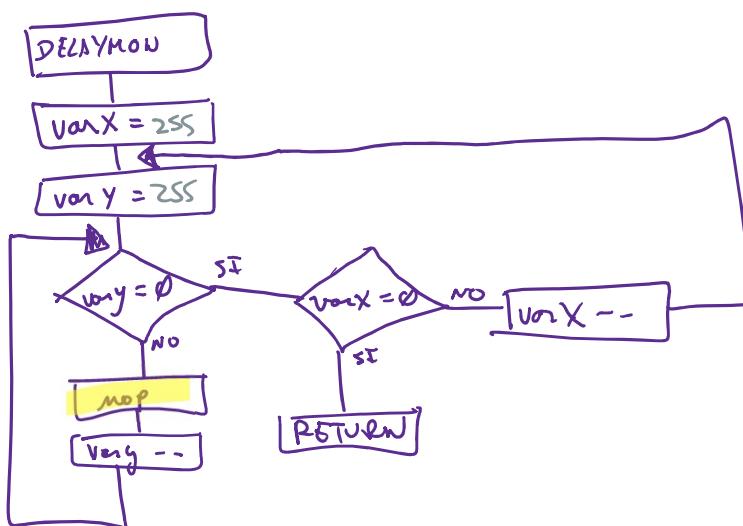
Ejemplo: Titilar un LED conectado en RD2

- Debemos de crear un bucle de repetición para generar retardos más prolongados



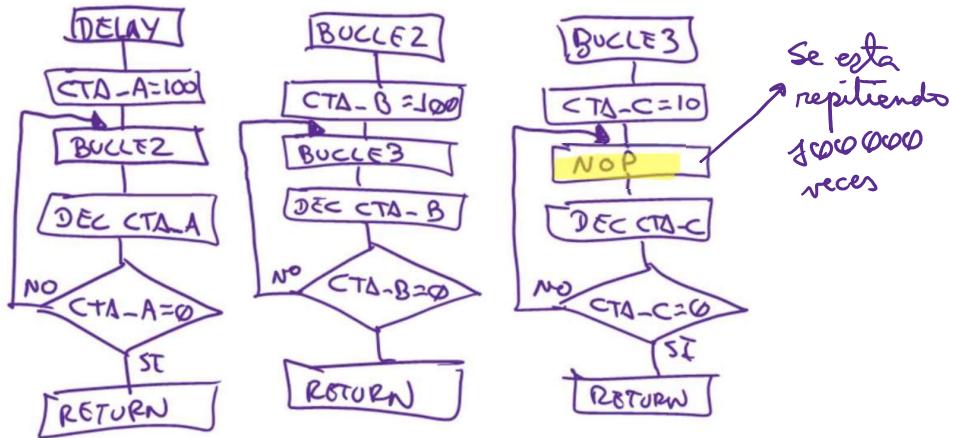
21

Propuesta de algoritmo con dos anillos de repetición



22

Propuesta de algoritmo con tres anillos de repetición



23

Código del titilador de LED en MPASM:

<pre> 1 list p=18f4550 2 #include <p18f4550.inc> ;libreria de nombre de los 3 4 CONFIG FOSC = XT_XT 5 CONFIG CPUDIV = OSC1_PLL2 6 CONFIG PWRT = ON ; Power-up Timer Enabled 7 CONFIG BOR = OFF ; Brown-out Reset Enabled 8 CONFIG WDT = OFF ; Watchdog Timer Enabled 9 CONFIG PBADEN = OFF ; PORTB A/D Enable bit 10 CONFIG LVP = OFF ; Single-Supply ICSP Enabled 11 12 cblock 0x000 13 var_i 14 var_j 15 var_k 16 endc 17 18 org 0x0000 ;Vector de reset 19 goto configuro 20 21 org 0x0020 ;Zona de programa de usuario 22 configuro: 23 bcf TRISD, 2 ;RD0 como salida 24 25 inicio: 26 bsf LATD, 2 ;prendo el led 27 call delaymon 28 bcf LATD, 2 ;apago el led 29 call delaymon 30 goto inicio </pre>	<pre> 32 delaymon: 33 movlw .50 34 movwf var_i 35 otro1: 36 call bucle1 ;Salto a subrutina 37 decfsz var_i,f 38 goto otro1 39 return 40 41 bucle1: 42 movlw .55 43 movwf var_j 44 otro2: 45 nop 46 nop 47 call bucle2 ;Salto a subrutina 48 decfsz var_j,f 49 goto otro2 50 return 51 52 bucle2: 53 movlw .20 54 movwf var_k 55 otro3: 56 nop 57 decfsz var_k,f 58 goto otro3 59 return 60 end </pre>
---	--

24

Prototipo físico del circuito:



25

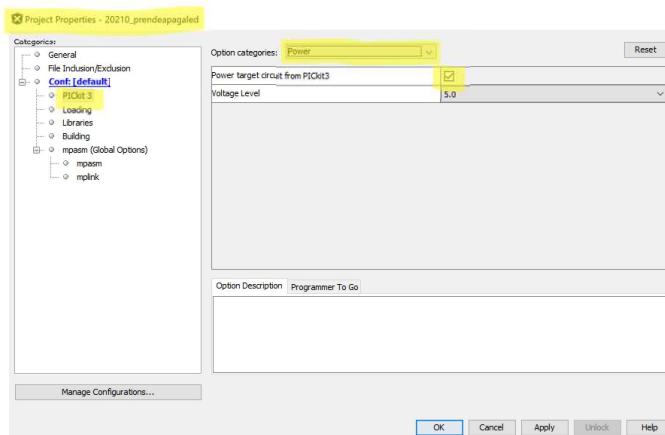
Recomendaciones al momento de implementar el circuito en físico:

- Verificar continuidad en los cables jumper utilizados en el circuito
- Verificar que la resistencia en pin MCLR hacia 5V sea de 10K
- Verificar que la PC haya detectado correctamente el PICKIT3
- Revisar siempre los mensajes en la ventana de “output” por posibles fallos en el evento de compilación y evento de programación.
- Tener a la mano un multímetro para verificar voltajes y continuidad en el prototipo.

26

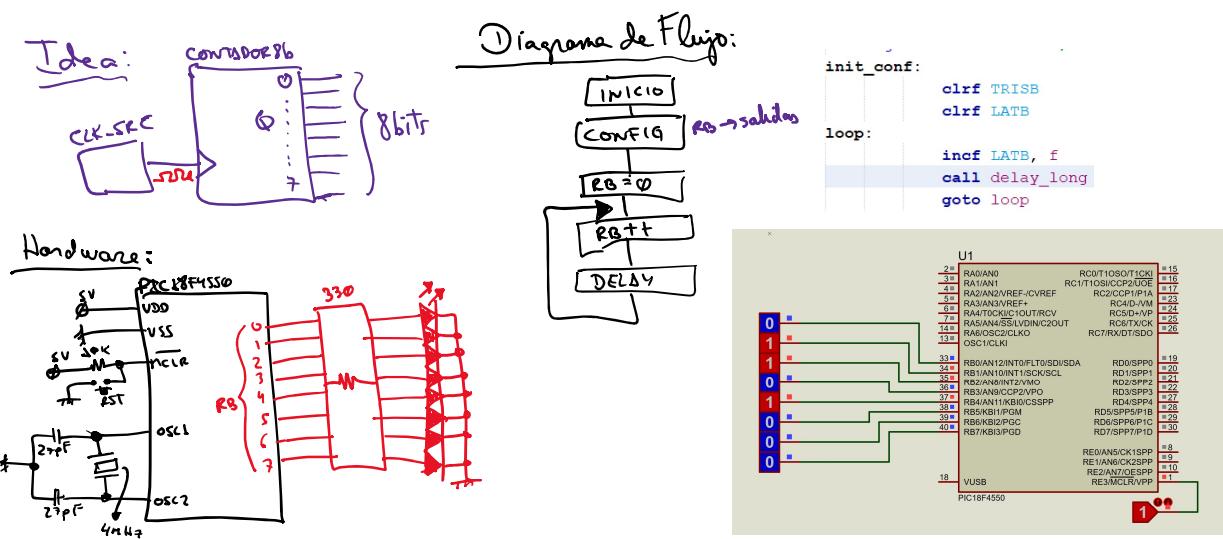
Recomendaciones al momento de implementar el circuito en físico:

- Si se está empleando el PICKIT3 como programador y deseas que éste administre el voltaje de alimentación al circuito de prueba:



27

Modificar el titileo de LED para que se visualice un contador autoincremental de 8 bits



28

Ejercicios adicionales:

Tener en cuenta que se debe de seguir el procedimiento visto en clase (idea, desarrollo del circuito, diagrama de flujo, código en MPASM, simulación, prototipo físico)

- Desarrollar un titilador de un LED conectado en RE0 en el cual su periodo de parpadeo dependerá del estado de un switch conectado en RB0, si RB0=1 el periodo será de 500ms, si RB0=0 el periodo será de 100ms.
- Desarrollar una señal de cruce de tren con entrada de activación.
- Desarrollar una “vela electrónica” con entrada de activación, dicha entrada tendrá como sensor de luz a un L.D.R.

Fin de la sesión!