

Microcontroladores

Laboratorio Semana 1

Semestre: 2021-1

Profesor: Kalun José Lau Gan

Empiezamos en breve

1

Agenda

- Requerimientos de software:
 - El MPLAB X IDE
 - El Proteus VSM
- Requerimientos de hardware:
 - Lista de materiales
 - Computadora
 - Instrumentos de laboratorio
- Requerimientos de documentos:
 - Hoja técnica del microcontrolador PIC18F4550 rev.E
 - <https://ww1.microchip.com/downloads/en/devicedoc/39632e.pdf>

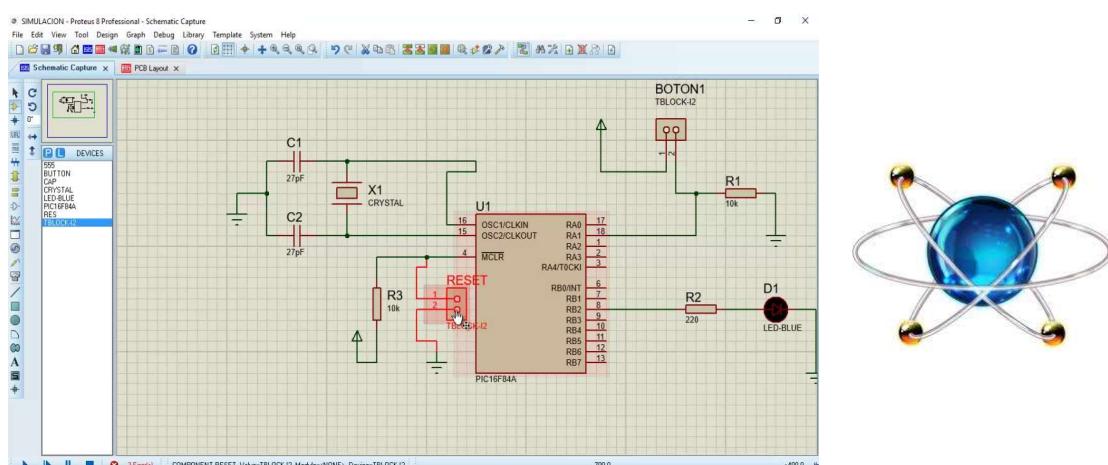
2

Verificación del funcionamiento de los softwares:

- ¿Se instaló correctamente el MPLAB X IDE?
- ¿Instalaste la versión v5.35 del MPLAB X IDE?
 - No instalar versión 5.45 debido a que usa XC8 Assembler en lugar de MPASM
 - Versión 5.35 ya no soporta PICKit2, si vas a usar este último tendrás que emplear la 5.30
- El Proteus. ¿Funciona correctamente?
- Verificar si el Proteus instalado tiene la librería de simulación para el microcontrolador PIC18F4550

3

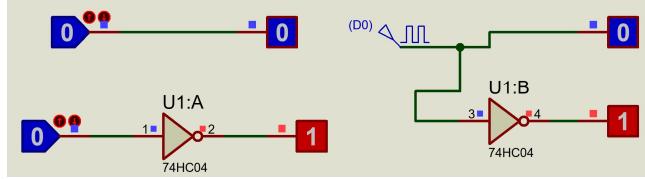
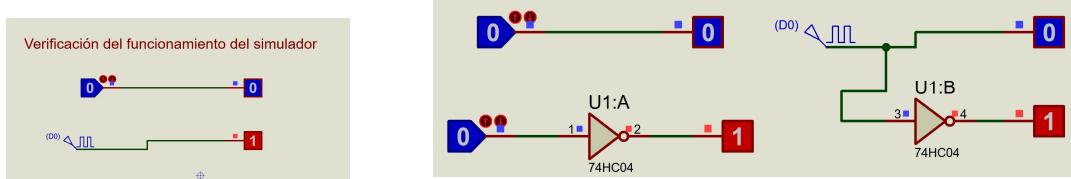
El Proteus VSM



- Simulador de circuitos

4

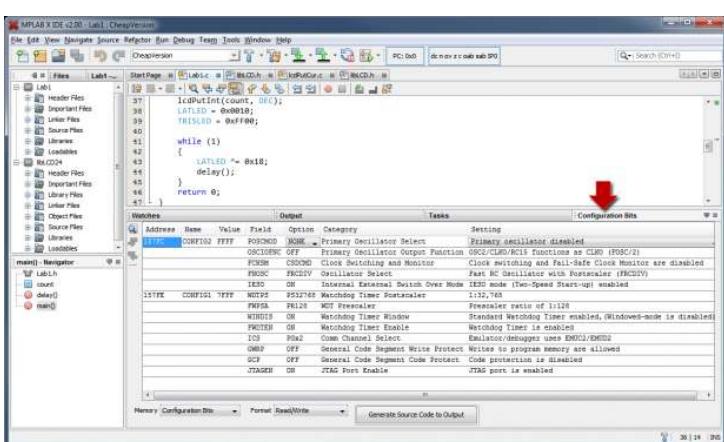
Verificación: Simulación en Proteus



5 minutos

5

El MPLAB X IDE

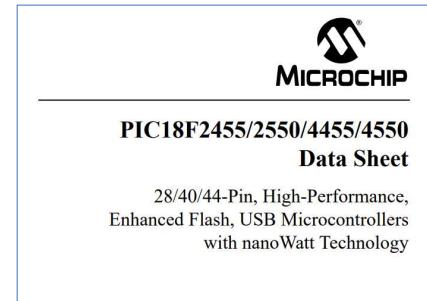


- Descargable desde el siguiente link:
<https://www.microchip.com/mplab/mplab-x-ide>
<https://www.microchip.com/development-tools/pic-and-dspic-downloads-archive>

6

Importancia de tener las hojas técnicas de los IC's a usar:

- Las hojas técnicas (datasheet) son proporcionadas por el fabricante del IC's y se detallan todas las funcionalidades, capacidades, configuraciones, limitaciones, etc de dicho dispositivo, es la información mas fiel.
- En nuestro caso tendremos siempre presente la hoja técnica del microcontrolador PIC18F4550 en su revisión E.



7

Procedimiento para desarrollar una aplicación con el microcontrolador PIC18F4550

1. Análisis del problema y ver los requerimientos (puertos E/S, tipo de señales, velocidad, consumo energético, etc)
2. Desarrollamos el hardware (el circuito)
3. Elaboramos el algoritmo en diagrama de flujo
4. Redactamos el código en un lenguaje de programación
5. Compilar y realizar la pruebas (simulación, emulación, programación)

8

Importancia de los comentarios en un código fuente

- Cuando uno desarrolla un programa, en cualquier lenguaje de programación, es fundamental colocar comentarios.
- Los comentarios no añaden espacio de memoria luego de la compilación.
- Los comentarios sirven para recordar ideas, configuraciones, procesos, algoritmos, etc que le permitan al programador en un tiempo después ver lo que hizo en dicho momento.
- En MPASM los comentarios van antecedidos por un punto y coma (;)

9

Plantilla de código en MPASM

```
;Los comentarios van antecedidos de un punto y coma

list p=18f4550          ;Modelo del microcontrolador
#include <p18f4550.inc> ;Librería de nombres de los regis

;Zona de declaración de los bits de configuración del mic
CONFIG PLLDIV = 1        ; PLL Prescaler Selection b
CONFIG CPUDIV = OSC1_PLL2 ; System Clock Postscaler S
CONFIG FOSC = XT_XT      ; Oscillator Selection bits
CONFIG PWRT = ON          ; Power-up Timer Enable bit
CONFIG BOR = OFF          ; Brown-out Reset Enable bi
CONFIG WDT = OFF          ; Watchdog Timer Enable bit
CONFIG CCP2MX = ON         ; CCP2 MUX bit (CCP2 input/
CONFIG PBADEN = OFF       ; PORTB A/D Enable bit (POR
CONFIG MCLRE = ON          ; MCLR Pin Enable bit (MCLR
CONFIG LVP = OFF          ; Single-Supply ICSP Enable

org 0x0000                ;Vector de RESET
goto init_conf

org 0x0008                ;Vector de interrupcion

org 0x0020                ;Zona de programa de usuario
init_conf:
|
|
end                      ;Fin del programa
```

10

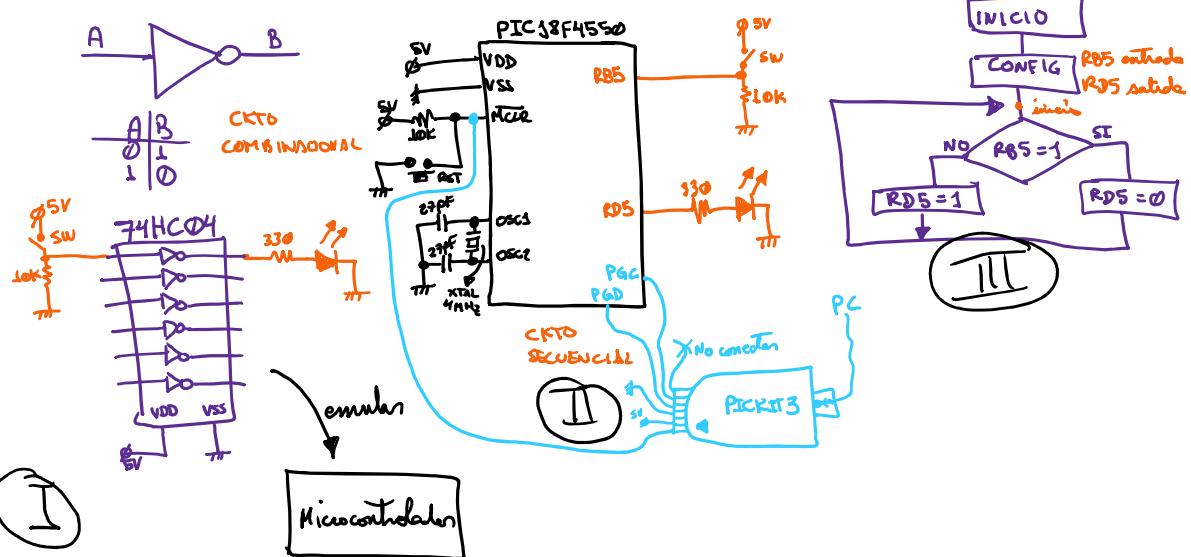
Directivas de pre-procesador (bits de configuración)

Los esenciales:

| | |
|---------------------|--|
| CONFIG FOSC = XT_XT | → Tipo de fuente de reloj: crystal externo hasta 4MHz |
| CONFIG PWRT = ON | → Establece un Tiempo de pausa al energizarse el uC |
| CONFIG BOR = OFF | → Detecta un límite mínimo en VDD, debajo del límite el uC estará en RST |
| CONFIG WDT = OFF | → Período guardián deshabilitado |
| CONFIG PBADEN = OFF | → Puerto RB como digitales |
| CONFIG LVP = OFF | → Programación con 5V deshabilitado |

11

Ejemplo: Desarrollar una NOT de un bit en el microcontrolador PIC18F4550

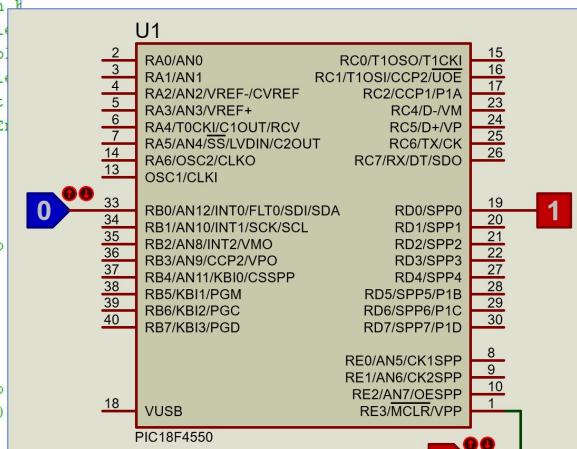


12

```

1      list p=18f4550          ;modelo de procesador
2      #include<p18f4550.inc> ;declaración de librería de n
3
4      ;Declaración de bits de configuración
5      CONFIG FOSC = XT_XT          ; Oscillator Selection, 1
6      CONFIG PWRT = ON            ; Power-up Timer Enable
7      CONFIG BOR = OFF             ; Brown-out Reset Enable
8      CONFIG WDT = OFF             ; Watchdog Timer Enable
9      CONFIG PBADEN = OFF          ; PORTB A/D Enable bit
10     CONFIG LVP = OFF             ; Single-Supply ICSP Enable
11
12     org 0x0000                  ;vector de reset
13     goto pre_conf
14
15     org 0x0020                  ;zona de programa de usuario
16
17     pre_conf:
18         bcf TRISD, 0           ;defino a RD0 como salida
19         bsf TRISB, 0           ;defino a RB0 como entrada
20
21     inicio:
22         btfss PORTB, 0          ;pregunto si RB0 es uno
23         goto noes              ;cuando lo anterior es falso
24         bcf LATD, 0             ;cuando fue verdadero, RD0=0
25         goto inicio             ;salto a inicio
26
27     noes:
28         bsf LATD, 0             ;RD0=1
29         goto inicio             ;salto a inicio
30
31     end

```



13

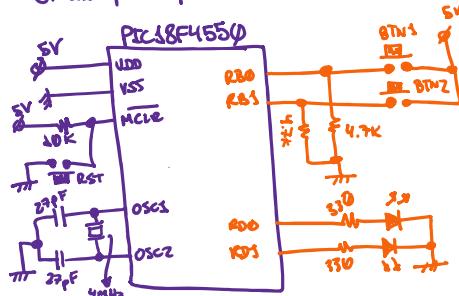
Ejemplo: Desarrollar un negador lógico de 2 bits empleando el PIC18F4550

Negación lógica:



$$\begin{array}{c|c} A & B \\ \hline \emptyset & 1 \\ \downarrow & \emptyset \end{array}$$

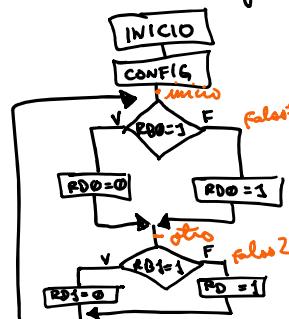
Circuitos prototípicos:



¿Qué es emular?

- Imitar el funcionamiento de un dispositivo en otra plataforma

Diagrama de flujo:



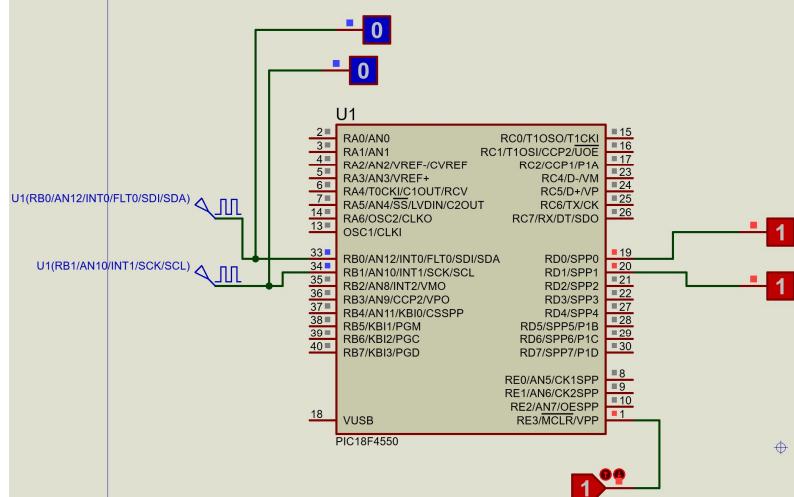
14

Ejemplo: Desarrollar un negador lógico de 2 bits empleando el PIC18F4550

```

1 ;Este es un comentario
2 list p=18f4550
3 #include <p18f4550.inc> ;libreria de nombre de los
4
5 CONFIG FOSC = XT_XT ; Oscillator Selection
6 CONFIG PWRT = ON ; Power-up Timer Enable
7 CONFIG BOR = OFF ; Brown-out Reset Enable
8 CONFIG WDT = OFF ; Watchdog Timer Enable
9 CONFIG PBADEN = OFF ; PORTB A/D Enable bit
10 CONFIG LVP = OFF ; Single-Supply ICSF Enable
11
12 org 0x0000 ;Vector de reset
13 goto configuro
14
15 org 0x0020 ;Zona de programa de usuario
configuro:
16 bcf TRISD, 0 ;RD0 como salida
17 bcf TRISD, 1 ;RD1 como salida
18
19 inicio:
20 btfss PORTB, 0 ;pregunta si rb0 es uno
21 goto falso1 ;viene cuando es falso
22 bcf LATD, 0 ;viene aqui cuando es verdadero
23 goto otro
24
falso1:
25 bcf LATD, 0
26 otro:
27 btfss PORTB, 1
28 goto falso2
29 bcf LATD, 1
30 goto inicio
31
falso2:
32 bcf LATD, 1
33 goto inicio
34
35

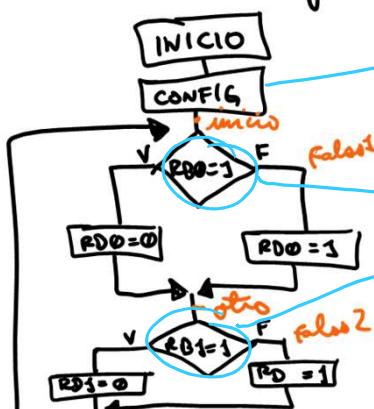
```



15

Ejemplo: Desarrollar un negador lógico de 2 bits empleando el PIC18F4550

Diagrama de flujo:



```

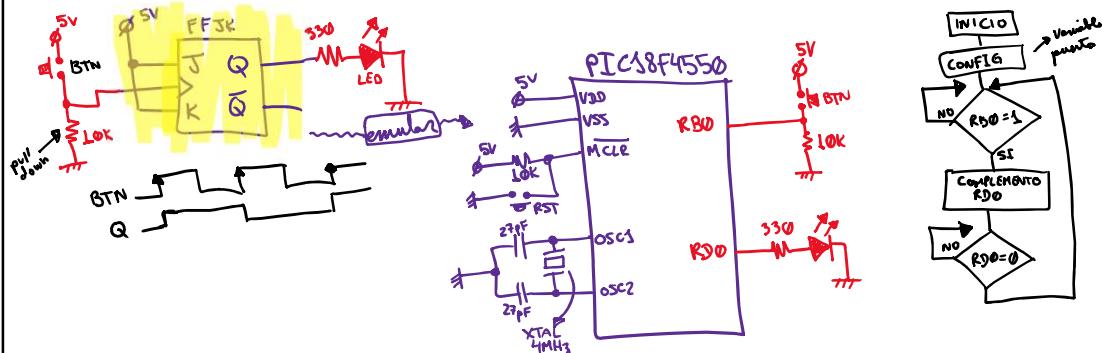
1 ;Este es un comentario
2 list p=18f4550
3 #include <p18f4550.inc> ;libreria de nombre de los
4
5 CONFIG FOSC = XT_XT ; Oscillator Selection
6 CONFIG PWRT = ON ; Power-up Timer Enable
7 CONFIG BOR = OFF ; Brown-out Reset Enable
8 CONFIG WDT = OFF ; Watchdog Timer Enable
9 CONFIG PBADEN = OFF ; PORTB A/D Enable bit
10 CONFIG LVP = OFF ; Single-Supply ICSF Enable
11
12 org 0x0000 ;Vector de reset
13 goto configuro
14
15 org 0x0020 ;Zona de programa de usuario
configuro:
16 bcf TRISD, 0 ;RD0 como salida
17 bcf TRISD, 1 ;RD1 como salida
18
19 inicio:
20 btfss PORTB, 0 ;pregunta si rb0 es uno
21 goto falso1 ;viene cuando es falso
22 bcf LATD, 0 ;viene aqui cuando es verdadero
23 goto otro
24
falso1:
25 bcf LATD, 0
26 otro:
27 btfss PORTB, 1
28 goto falso2
29 bcf LATD, 1
30 goto inicio
31
falso2:
32 bcf LATD, 1
33 goto inicio
34
35

```

16

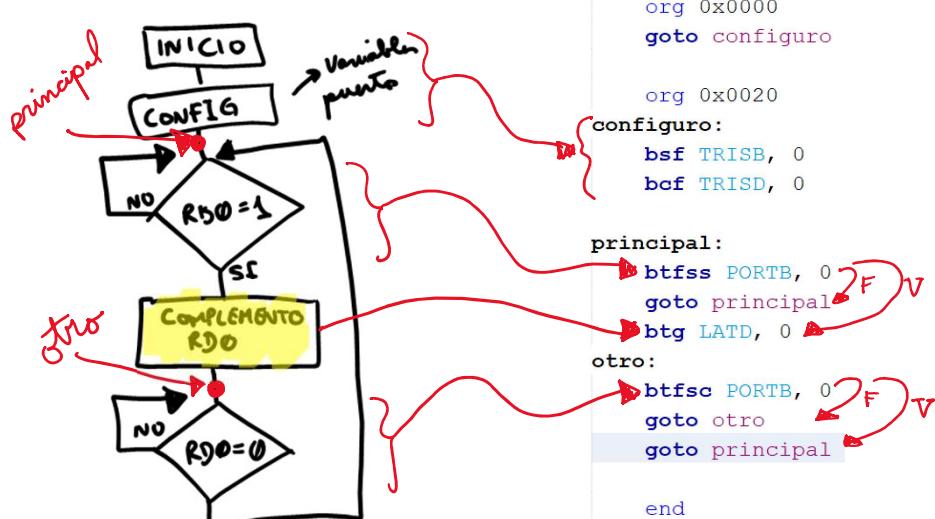
Ejemplo: Desarrollar lo siguiente en el microcontrolador PIC18F4550

- Un circuito latch para un botón y un LED



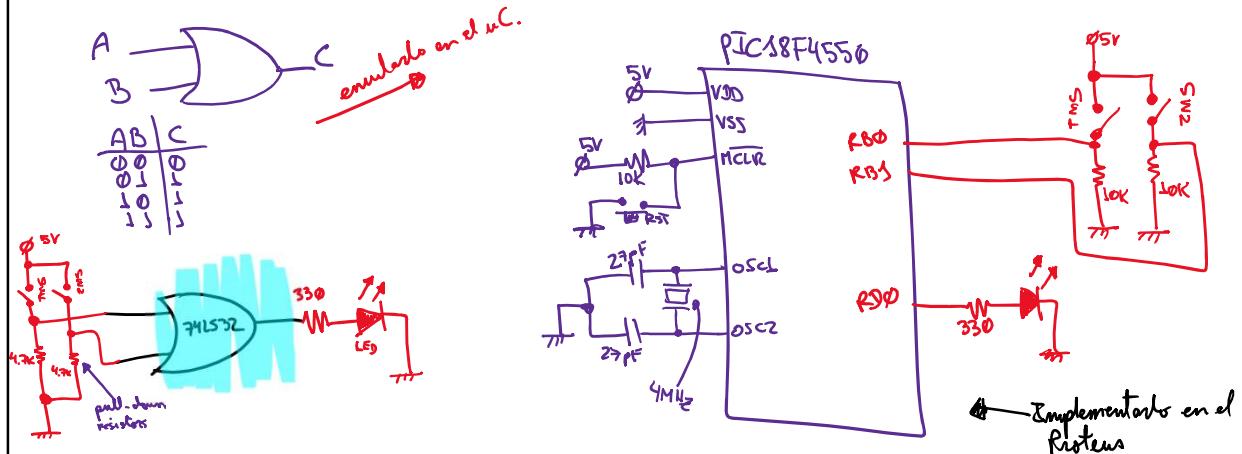
17

Relación entre diagrama de flujo y código en MPASM:



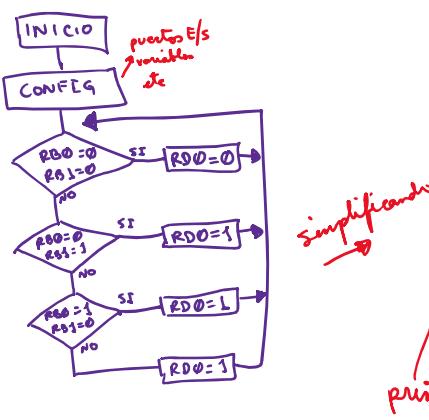
18

Ejemplo: Desarrollar un OR con el PIC18F4550



19

Diagrama de flujo de OR en el uC



```

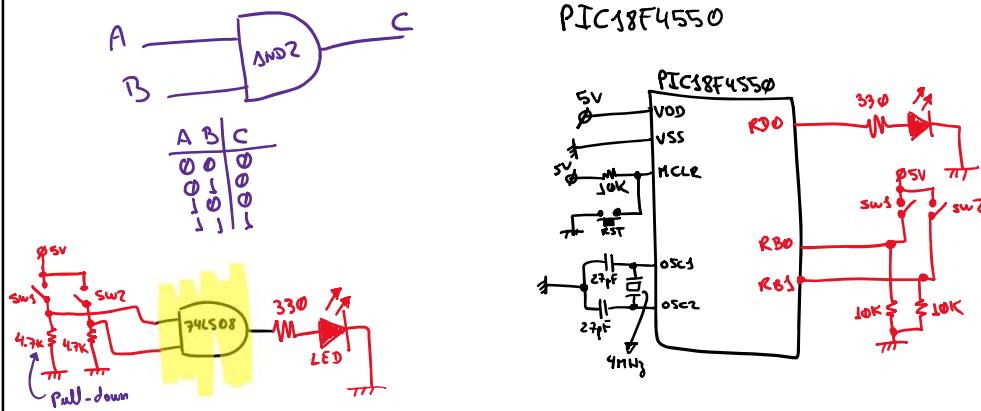
13 org 0x0000
14 goto configuro
15
16 org 0x0020
17 configuro:
18 { bcf TRISD, 0
19 bsf TRISB, 0
20 bsf TRISB, 1
21
22 principal:
23 btfsc PORTB, 0
24 goto falso F
25 btfsc PORTB, 1
26 goto falso F
27 bcf LATD, 0 V
28 goto principal
29
30 bsf LATD, 0
31 goto principal
32 end
  
```

20

Ejemplo de desarrollo de una aplicación con el microcontrolador PIC18F4550

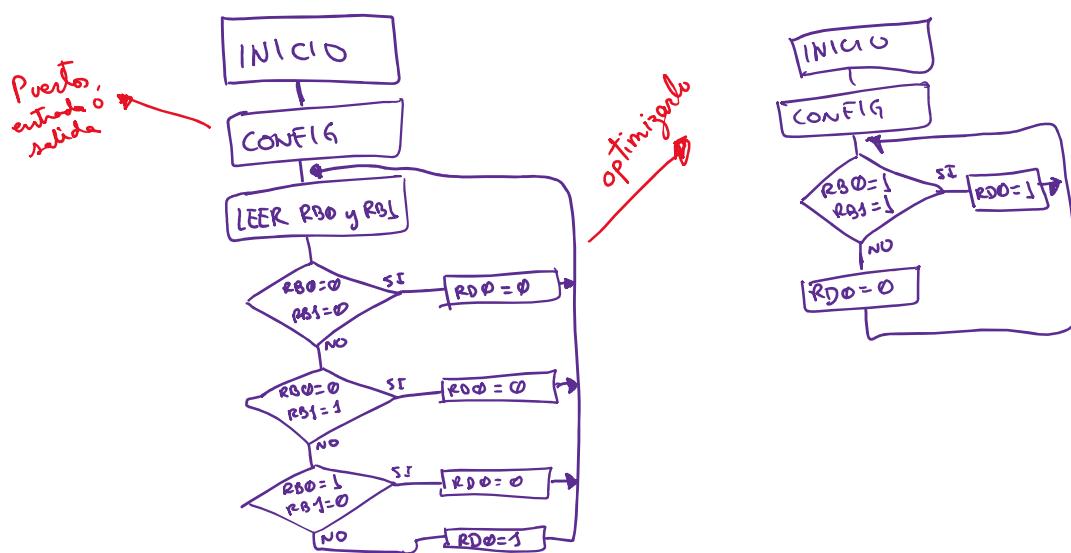
- Emular el funcionamiento de una AND2 en el microcontrolador PIC18F4550

- Construir el circuito usando el PIC18F4550



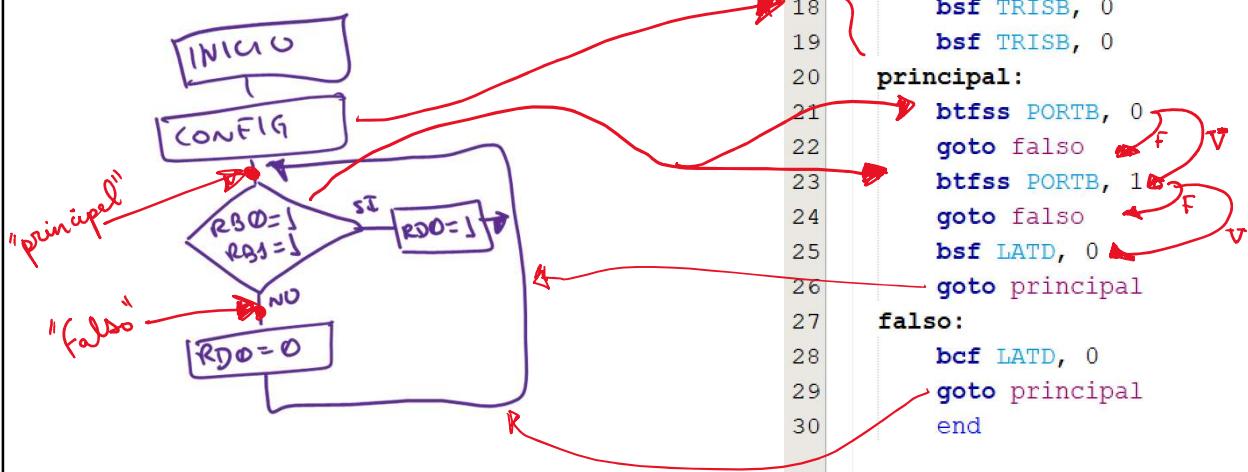
21

Modelo en diagrama de flujo:



22

Correspondencia entre diagrama de flujo y código en assembler



23

Cuestionario:

1. Averiguar el modo de operación de las instrucciones BTFSS, BTFSC, INCFSZ, DECFSZ, CPFSEQ, CPFSLT, CFPSGT.
2. ¿Qué instrucciones están de mas (redundantes) en los códigos de los ejemplos y por qué?

24

Fin de la sesión!