# Microcontroladores
# Laboratorio Semana 1

Semestre: 2022-1

Profesor: Kalun José Lau Gan

1

## Agenda

- Requerimientos de software:
  - El MPLAB X IDE v6.00
  - El XC8 v2.36
  - El Proteus VSM v8.xx en adelante
- Requerimientos de hardware:
  - Lista de materiales
  - Computadora
  - Instrumentos de laboratorio y herramientas
- Requerimientos de documentos:
  - Hoja técnica del microcontrolador PIC18F4550 rev.E
  - https://ww1.microchip.com/downloads/en/devicedoc/39632e.pdf
- Nuestro primer ejemplo

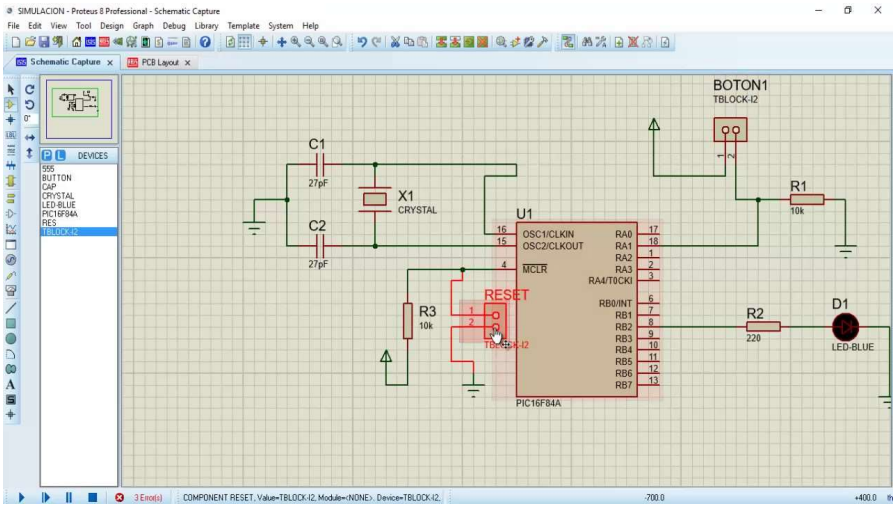
2

# Verificación del funcionamiento de los softwares:

- ¿Instalaste la versión v5.50 u otros mas antiguos del MPLAB X IDE?
  - **Debes** de instalar la última versión (v6.00) para trabajar en este curso
  - https://www.microchip.com/en-us/development-tools-tools-and-software/mplab-x-ide
  - Revisar si instalaste el compilador XC8 (v2.36)
  - https://www.microchip.com/en-us/development-tools-tools-and-software/mplab-xc-compilers
- ¿Se instaló correctamente el MPLAB X IDE? (Ver si abre correctamente el programa)
- El Proteus v8 en adelante. ¿Funciona correctamente? ¿No se cierra en plena simulación?
- Verificar si el Proteus instalado tiene la librería de simulación para el microcontrolador PIC18F4550

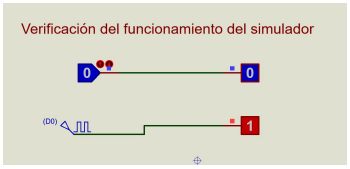


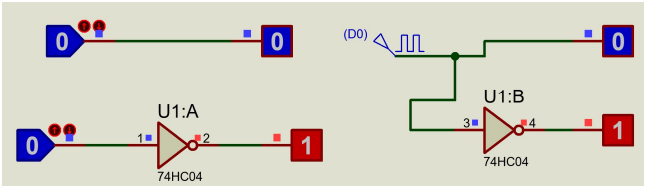
# El Proteus VSM




- Simulador de circuitos

# Verificación: Simulación en Proteus



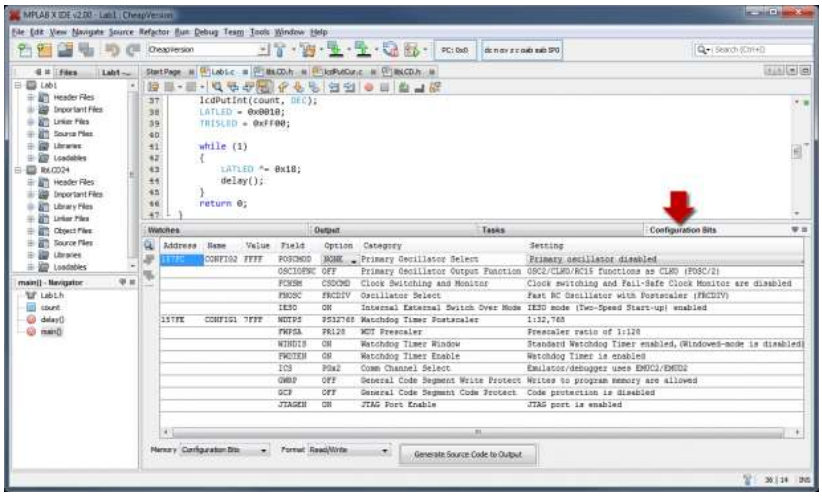




Logic state (entradas)
Logic probes (salidas)
Clock signal (señal de reloj)

Simulación



# 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

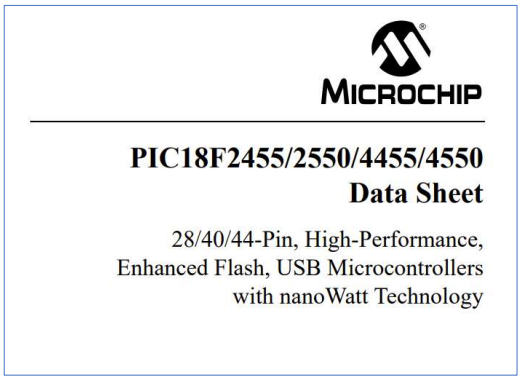
# MPASM vs. PICASM (XC8 Assembler)

- MPASM fué el lenguaje de programación hasta la v5.35, actualmente en obsolecencia
- XC8 PIC Assembler es el nuevo formato de lenguaje y soportado por la nueva versión 6.00
- Las instrucciones de los microcontroladores no han variado, solo la sintaxis de programación.
- MPLAB -> MPLAB X (32bits) -> MPLAB X (64bits)

7

# 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.
- Link de descarga: https://ww1.microchip.com/downloads/en/devicedoc/39632e.pdf


MICROCHIP

PIC18F2455/2550/4455/4550
Data Sheet

28/40/44-Pin, High-Performance, Enhanced Flash, USB Microcontrollers with nanoWatt Technology

8

# 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 en el simulador y/o implementado)
3. Elaboramos el algoritmo (Flowchart, Nassi-Schneiderman, etc)
4. Redactamos el código en un lenguaje de programación (Assembler, BASIC, C, Python, etc)
5. Compilar y realizar la pruebas (simulación, emulación, programación)

9

# 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 ó PICASM los comentarios van antecedidos por un punto y coma (;)

10

## Consideraciones importantes al usar simuladores

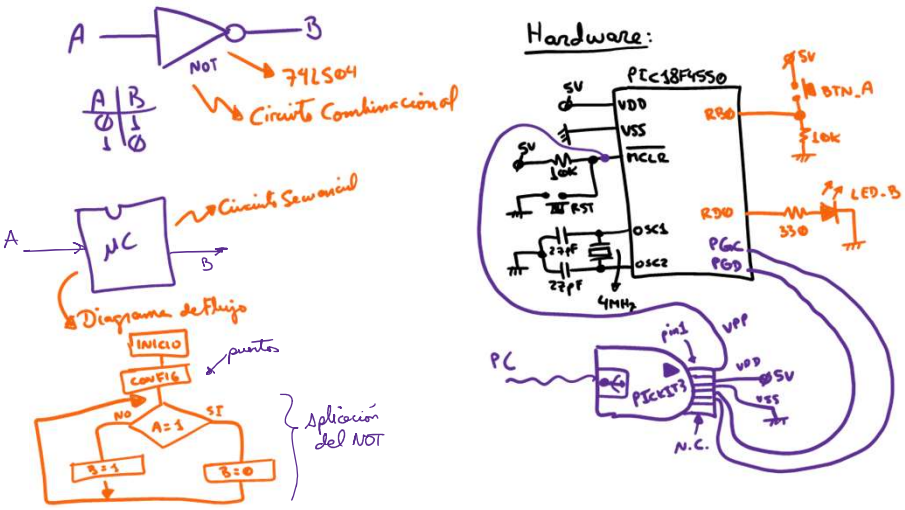
- El uso de simuladores ha permitido acelerar los procesos de validación de circuitos eléctricos y electrónicos, **pero** no es un determinante a la hora de validar en forma física.
- En la mayoría de casos en ingeniería electrónica el producto final es algo físico por lo que no solamente podemos fiarnos de una simulación y dar por sentado que la propuesta funcione correctamente.
- En Proteus hay elementos que no se muestran en el momento de hacer simulaciones.



## Sobre el MPLAB X IDE

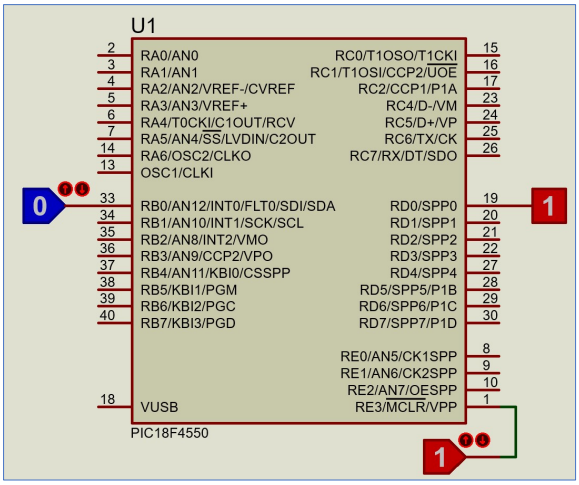
1. Crear un proyecto (seleccionar Standalone Project)
2. Seleccionar el dispositivo microcontrolador (PIC18F4550)
3. Seleccionar la herramienta “pic-as” (XC8 PIC Assembler)
4. Crear el archivo header (*.inc) e incluirle los bits de configuración (Window / Target Memory Views / Configuration Bits)
5. Crear el archivo fuente (*.s) e incluir el archivo header
6. Para compilar: 

# Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F4550






- Circuito en Proteus



- Código en XC8 Assembler

cabecera.inc

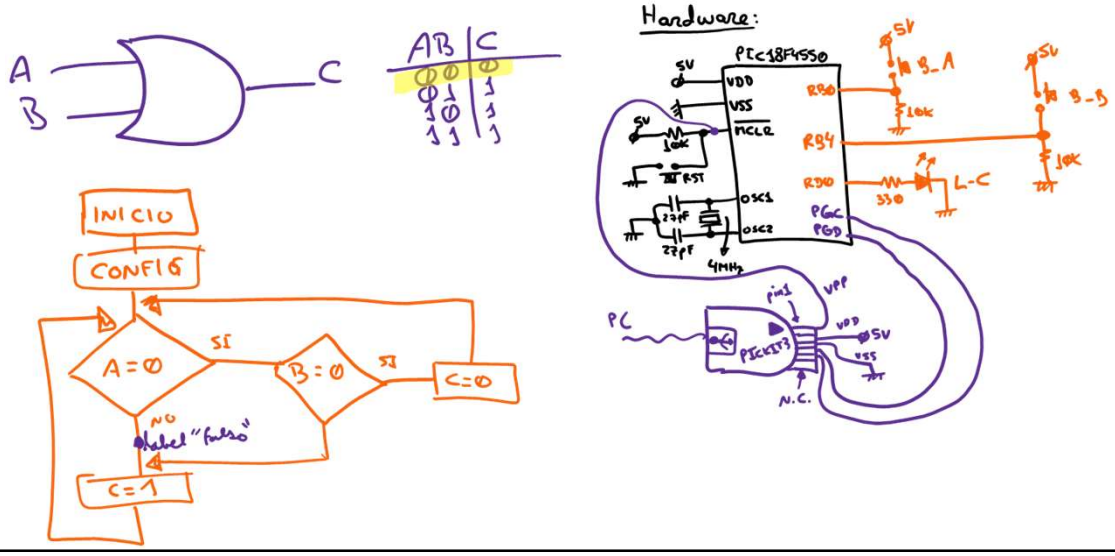
```
; PIC18F4550 Configuration Bit Settings

; Assembly source line config statements

; CONFIG1L
  CONFIG  PLLDIV = 1            ; PLL Pre
  CONFIG  CPUDIV = OSC1_PLL2    ; System
  CONFIG  USBDIV = 1            ; USB Clo

; CONFIG1H
  CONFIG  FOSC = XT_XT          ; Oscilla
  CONFIG  FCMEN = OFF           ; Fail-Sa
  CONFIG  IESO = OFF            ; Interna

; CONFIG2L
  CONFIG  PWRT = ON             ; Power-u
  CONFIG  BOR = OFF             ; Brown-o
  CONFIG  BORV = 3              ; Brown-o
  CONFIG  VREGEN = OFF          ; USB Vol

; CONFIG2H
  CONFIG  WDT = OFF             ; Watchdo
  CONFIG  WDTPS = 32768         ; Watchdo

; CONFIG3H
  CONFIG  CCP2MX = ON           ; CCP2 MU
  CONFIG  PBADEN = OFF          ; PORTB A
  CONFIG  LPT1OSC = OFF         ; Low-Pow
  CONFIG  MCLRE = ON            ; MCLR Pi

; CONFIG4L
  CONFIG  STVREN = ON           ; Stack F
  CONFIG  LVP = OFF             ; Single-
  CONFIG  ICPRT = OFF           ; Dedicat
  CONFIG  XINST = OFF           ; Extende
```

maincode.s

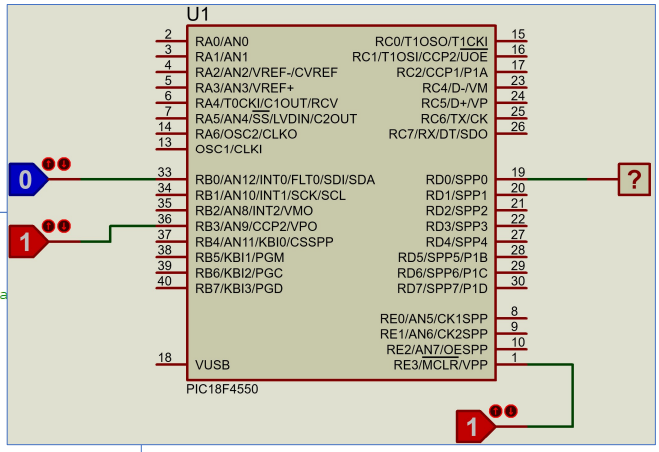
```
    PROCESSOR 18F4550                   ;Modelo del microcontrolador
    #include "cabecera.inc"             ;Llamada a archivo cabecera

    PSECT not_gate, class=CODE, reloc=2, abs    ;Creación de program section

not_gate:
    ORG 0000H                           ;Zona de vector de reset
    goto configuro

    ORG 0020H                           ;Zona de programa de usuario
configuro:
    ;Configuraciones iniciales adicionales
    bcf TRISD, 0                        ;RD0 es una salida
    bsf TRISB, 0                        ;RB0 es una entrada

loop:
    btfss PORTB, 0                      ;Pregunto si RB0 es uno
    goto falso                          ;Falso, salta a etiqueta falso
    bcf LATD, 0                         ;Verdad, manda a 0 el RD0
    goto loop                           ;Retorna al inicio (etiqueta loop)
falso:
    bsf LATD, 0                         ;Manda a 1 el RD0
    goto loop                           ;Retorna al inicio (etiqueta loop)
    end not_gate                        ;Cierre del program section
```



# Desarrollar una compuerta OR con el microcontrolador PIC18F4550



- Circuito en Proteus




- Código en XC8 Assembler

```
1       PROCESSOR 18F4550
2       #include "cabecera.inc"
3
4       PSECT principal, class=CODE, reloc=2, abs   ;Decla
5       org 0x0000          ;vector de reset
6       goto configuro
7
8       org 0x0020          ;zona de programa de usuario
9   configuro:
10              bsf TRISB, 0, 0     ;RB0 como entrada
11              bsf TRISB, 4, 0     ;RB4 como entrada
12              bcf TRISD, 0, 0     ;RD0 como salida
13  inicio:
14              btfsc PORTB, 0, 0   ;Pregunto si RB0 es cero
15              goto falso          ;Falso, salta a label falso
16              btfsc PORTB, 4, 0   ;Verdadero. Pregunto si RB4 es cero
17              goto falso          ;Falso, salta a label falso
18              bcf LATD, 0, 0      ;Verdadero. RD0 a 0
19              goto inicio         ;Retorna a inicio
20  falso:      bsf LATD, 0, 0              ;RD0 a 1
21              goto inicio         ;Retorna a inicio
22              end principal       ;Fin del programa
```



# Cuestionario:

1. ¿Cómo funcionan las instrucciones BSF, BCF, BTFSS y GOTO?
2. ¿Cuál es la diferencia entre BTFSS y BTFSC?
3. Averiguar el modo de operación de las instrucciones BTFSS, BTFSC, INCFSZ, DECFSZ, CPFSEQ, CPFSLT, CFPSGT.
4. ¿Qué instrucciones están de mas (redundantes) en los códigos de los ejemplos y por qué?

Fin de la sesión!