Sistemas Digitales: Introducción a los microcontroladores Parte 2

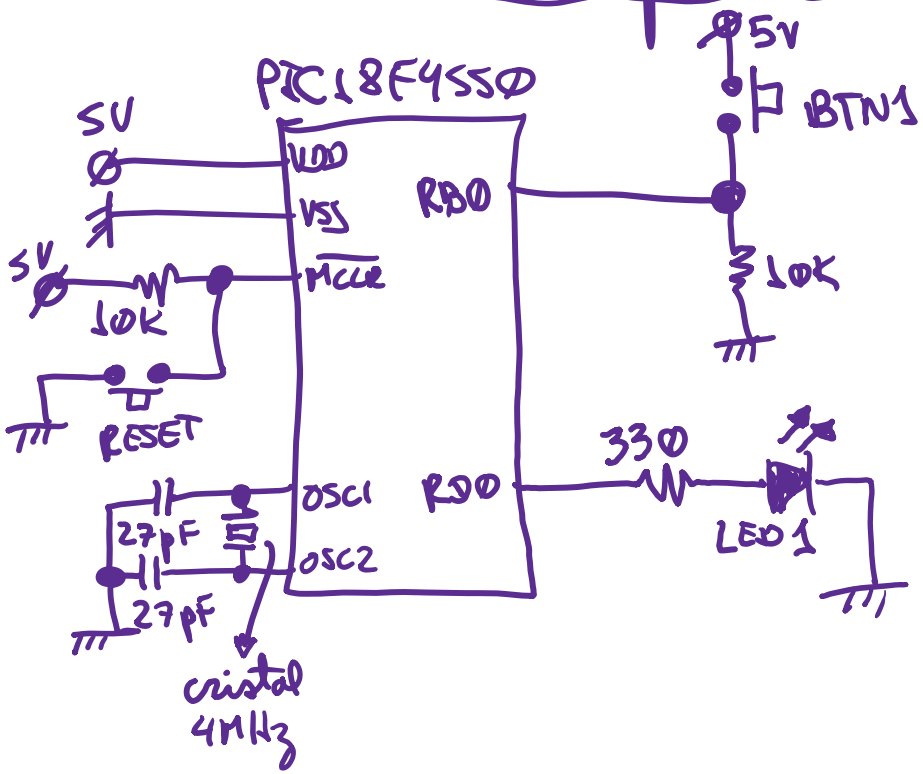
Ingeniería Electrónica

UPC 2018

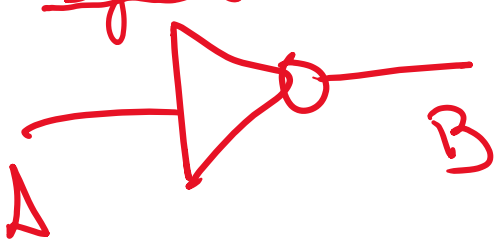
Por Kalun Lau

Ejemplo: Negador lógico

Circuito a implementar:



symbol:

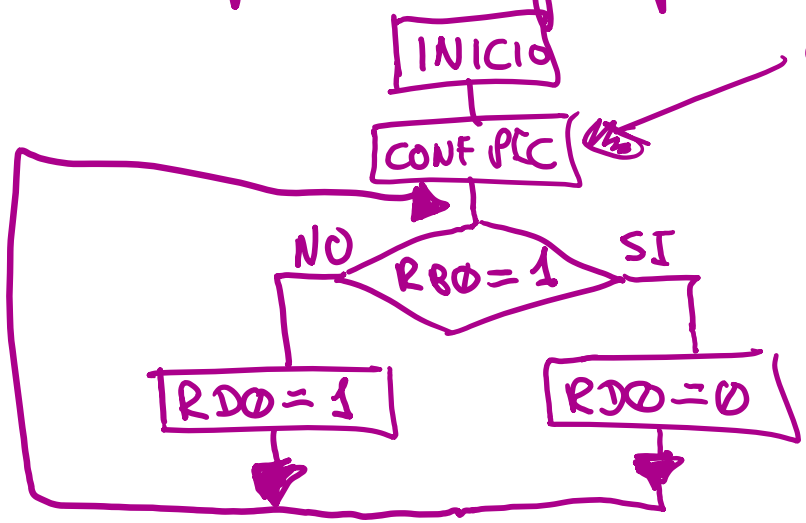


truth table:

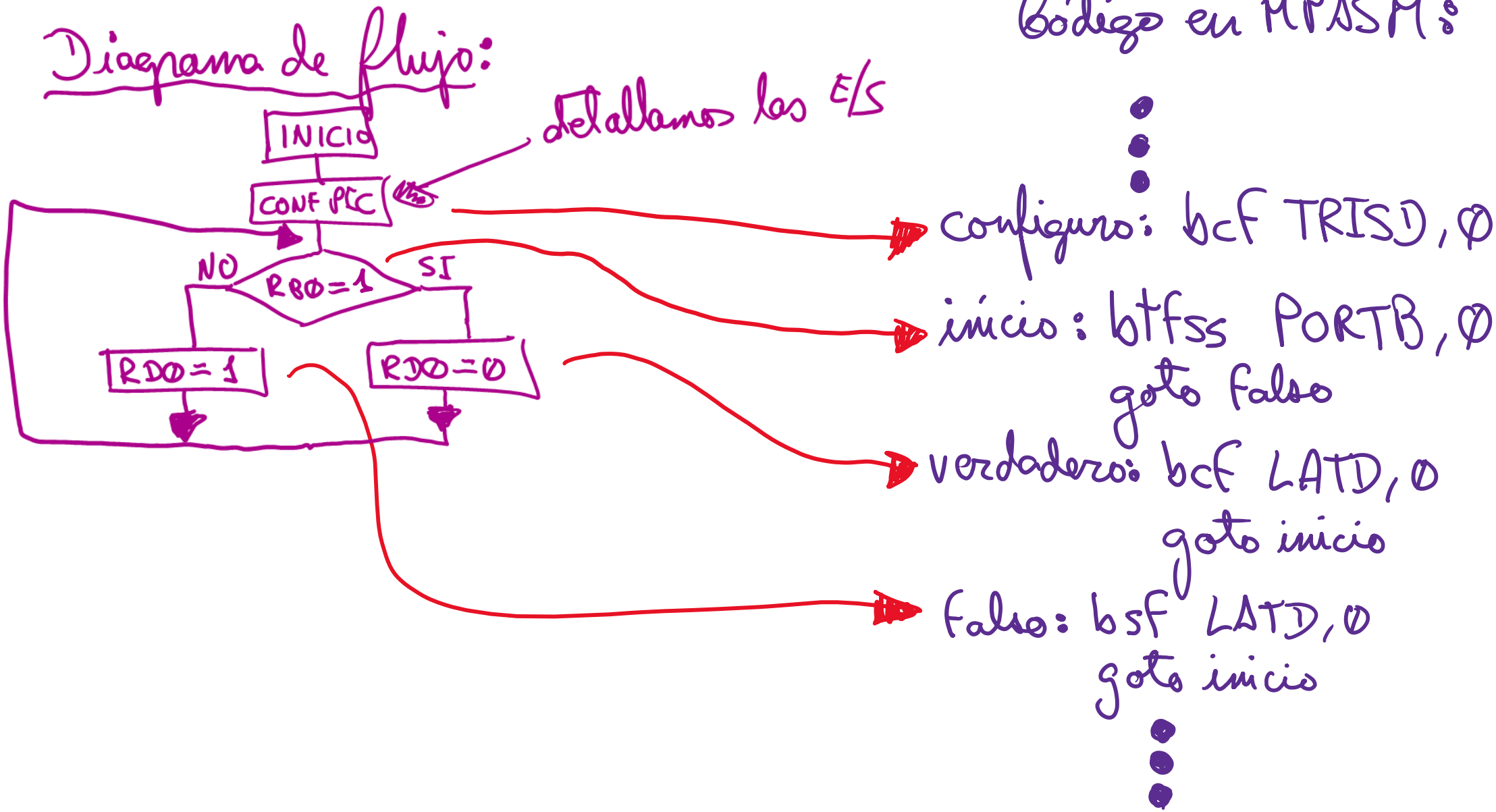
A	B
0	1
1	0

Diagrama de flujo:

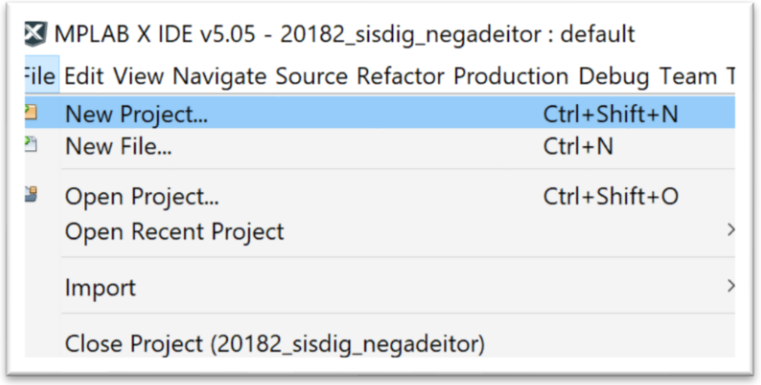
detallamos las E/S



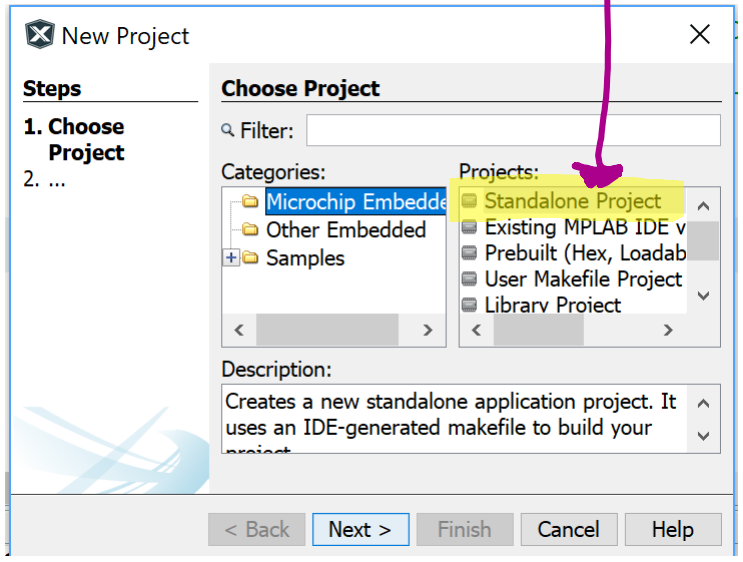
Ejemplo: Negador lógico

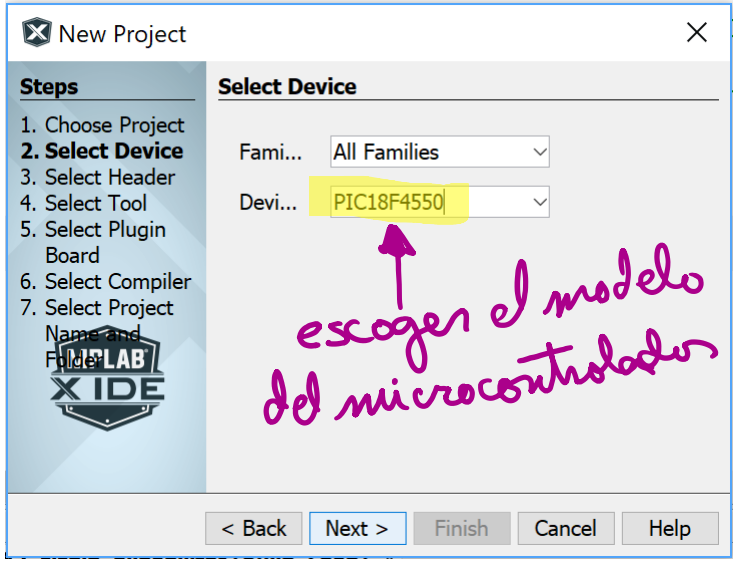


Creación del proyecto en el MPLAB X

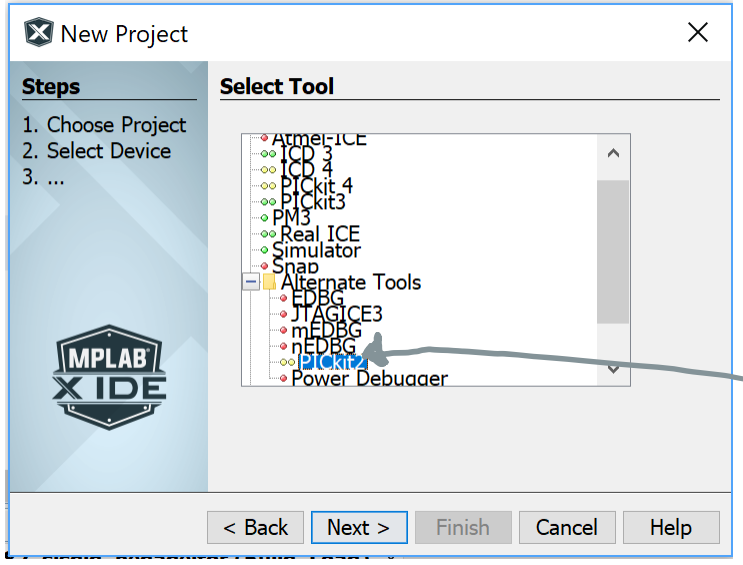


seleccionar

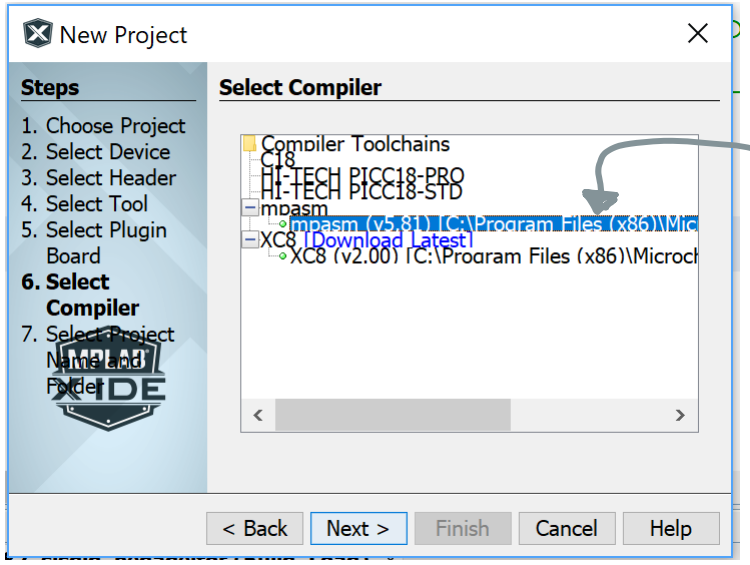




escoger el modelo del microcontrolador

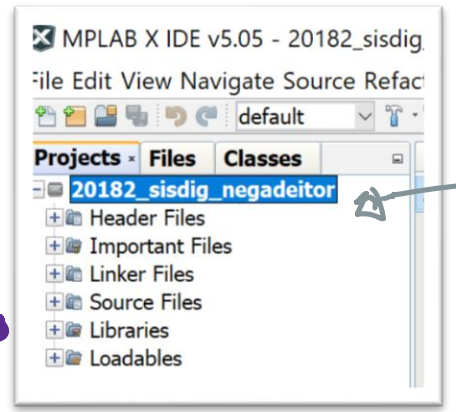


seleccionar PICKIT2

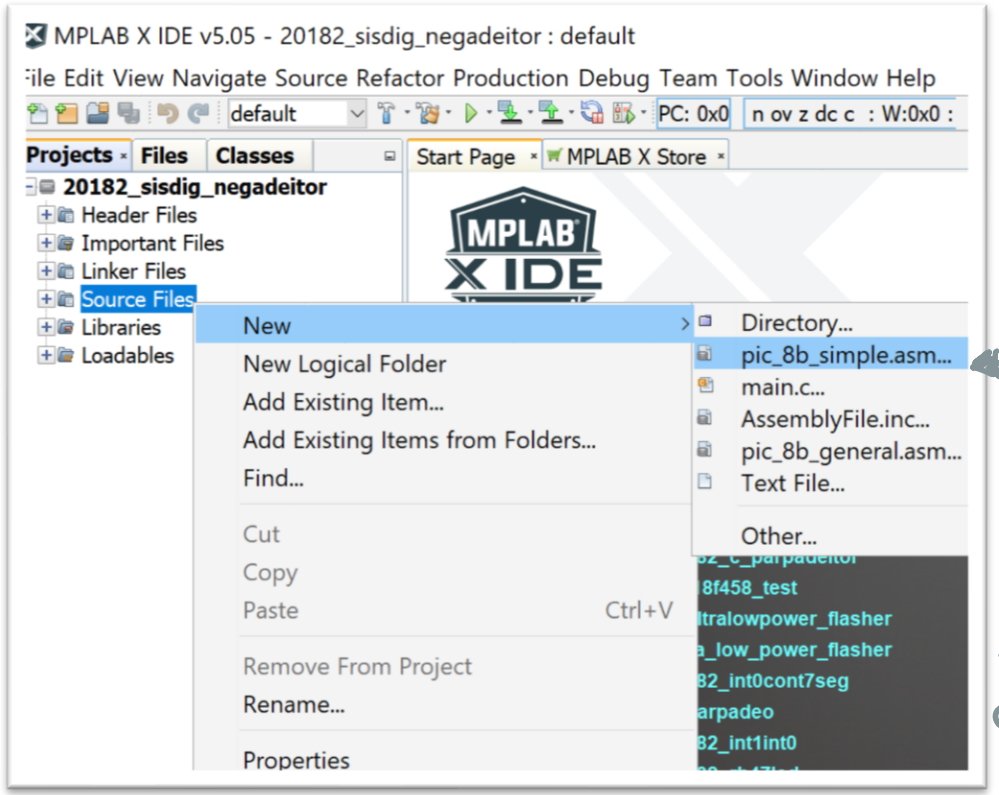


seleccionar MPASM

Creación del proyecto en el MPLAB X

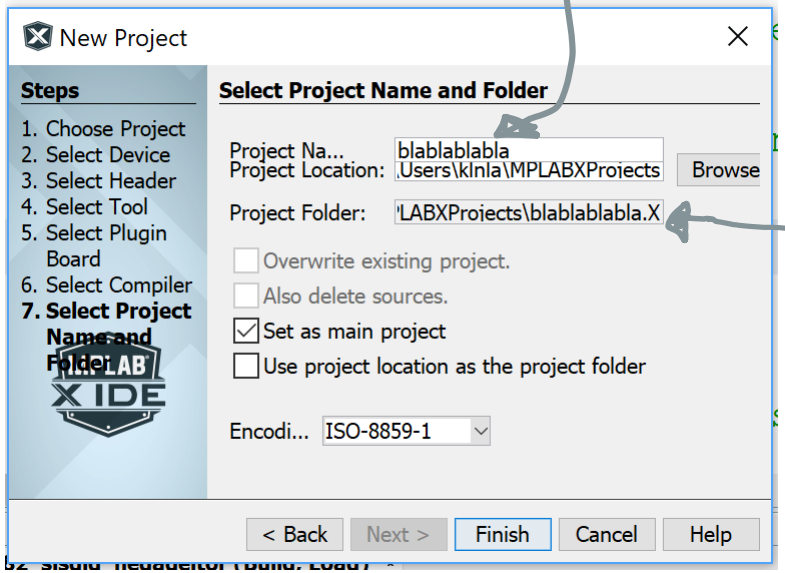


proyecto creado



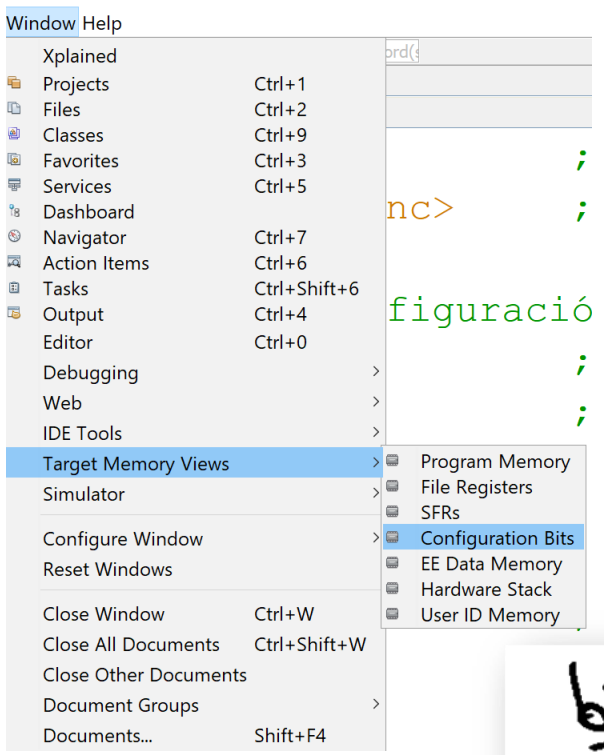
creación de un nuevo archivo fuente

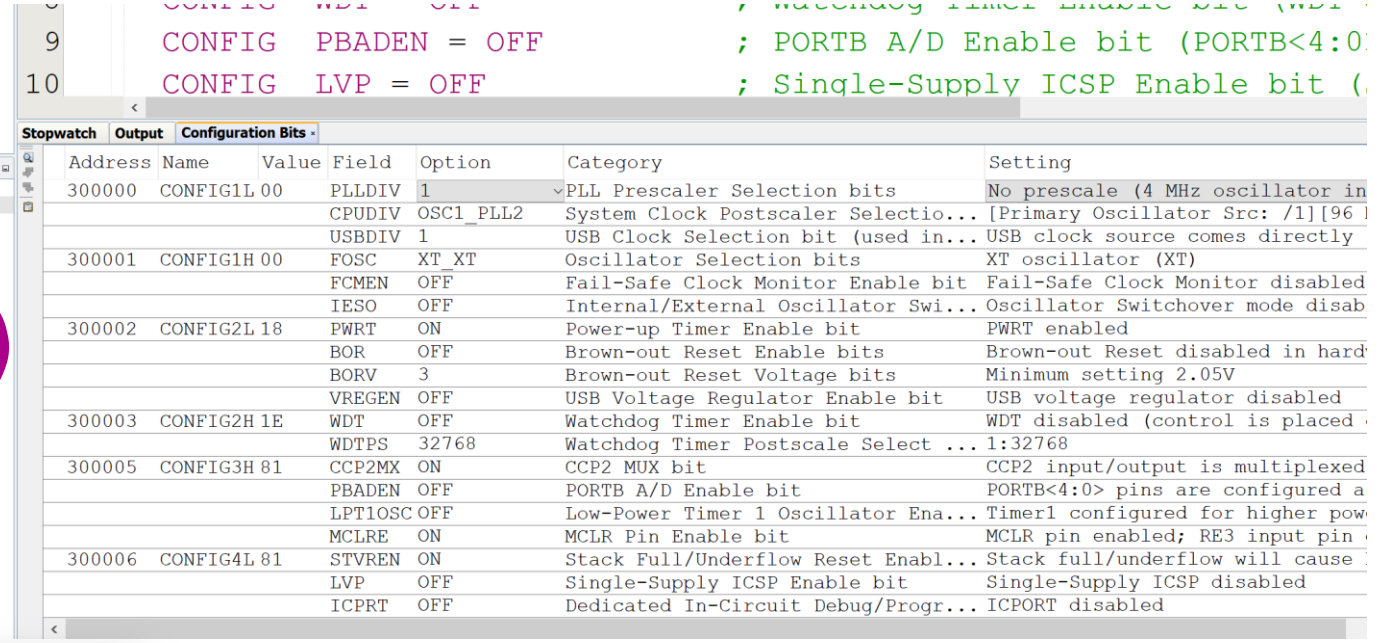
nombre del proyecto

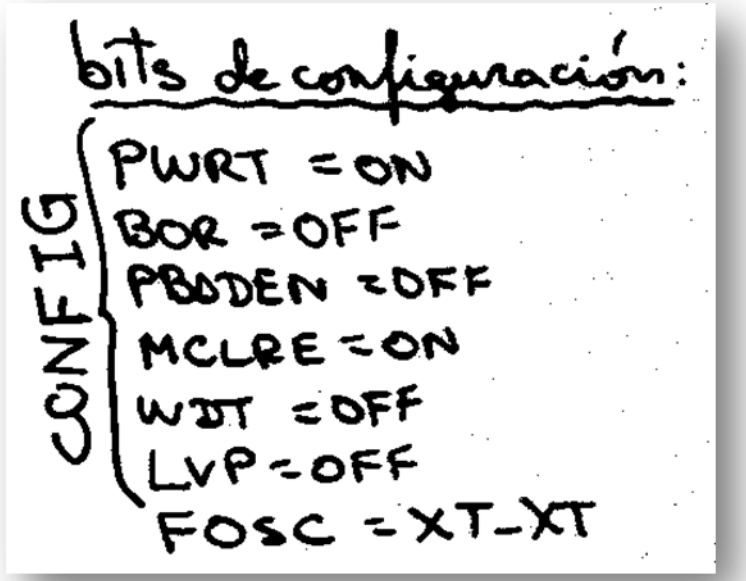


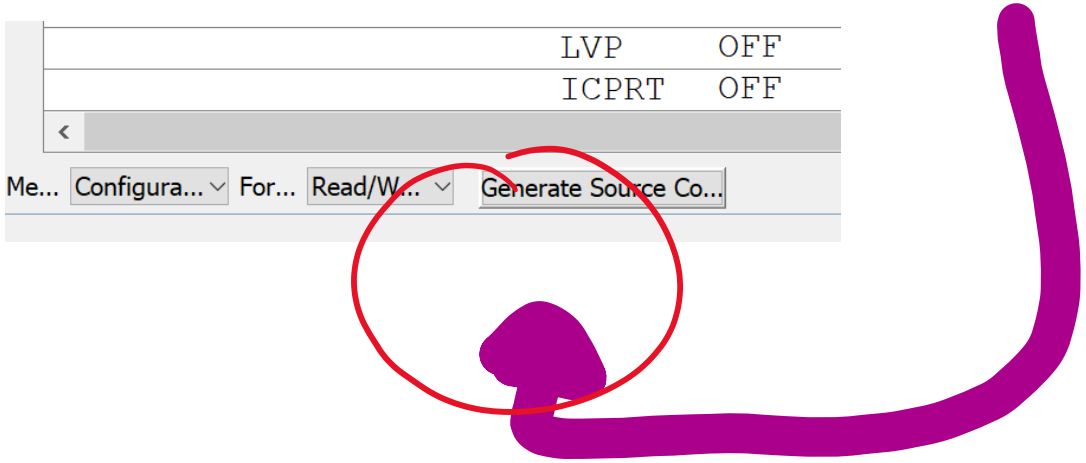
verificar ruta donde se va a almacenar el proyecto

Obtención de los bits de configuración

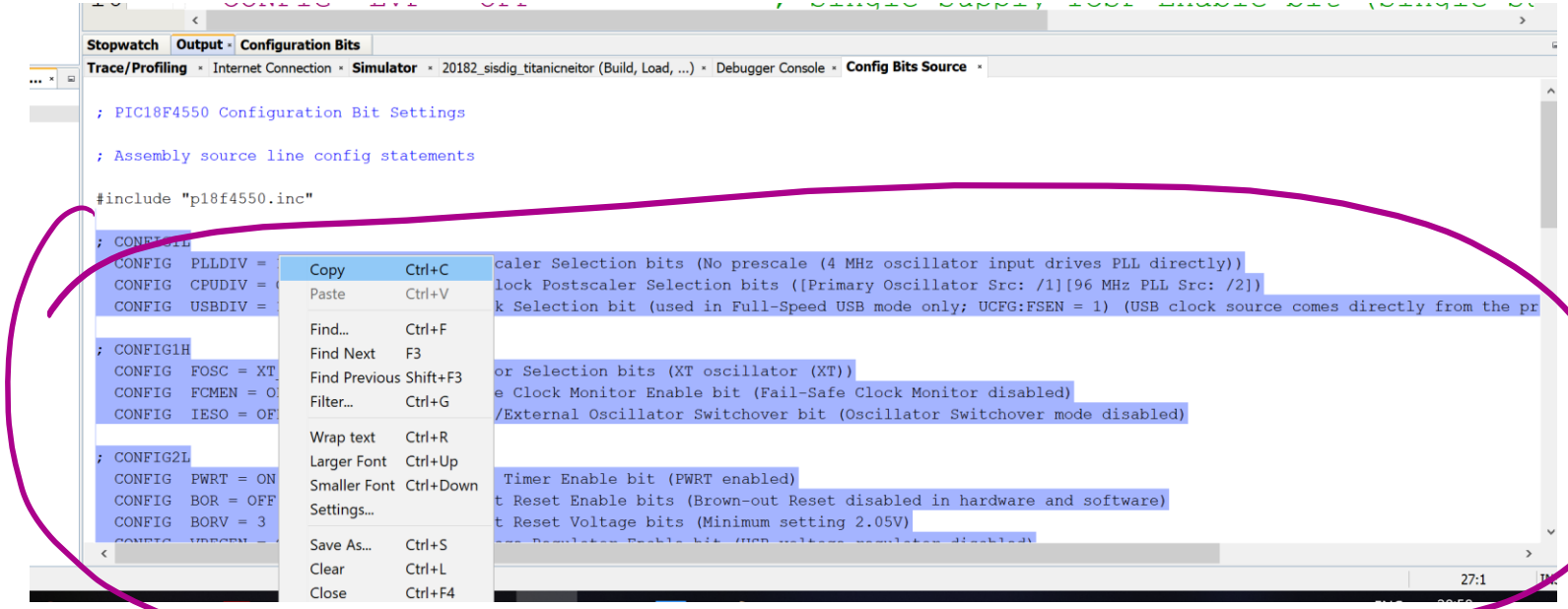


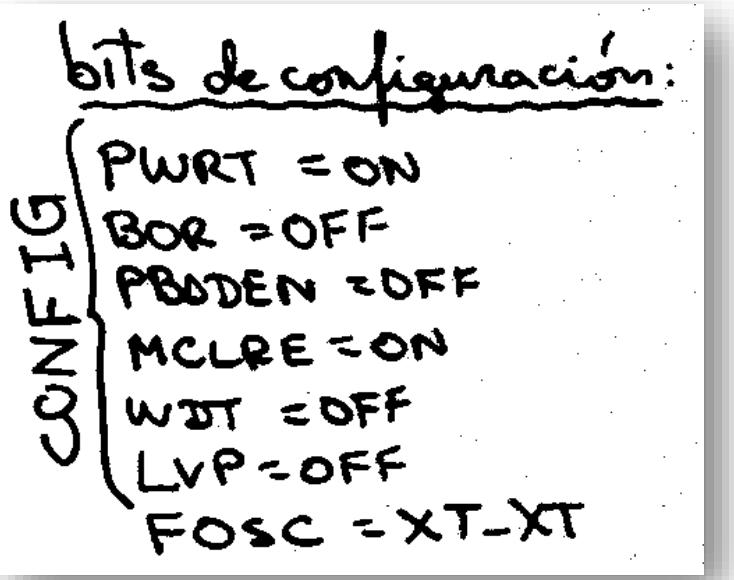






Obtención de los bits de configuración

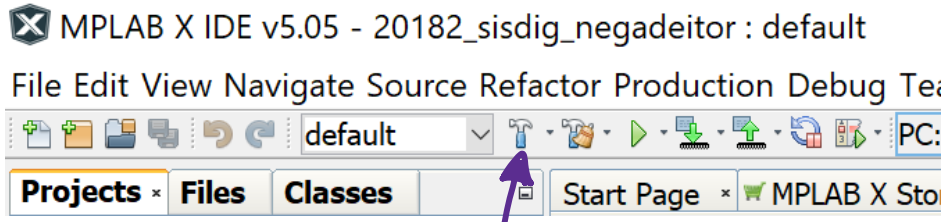




copias y pegas a tu código, luego borras los bits que no has modificado

Plantilla y código del programa

```
1       list p=18f4550              ;Modelo del microcontrolador
2       #include <p18f4550.inc>     ;librería de nombres
3
4   ;Zona de los bits de configuración del microcontroleitor
5     CONFIG  FOSC = XT_XT          ; Oscillator Selection bits (XT oscillator (XT))
6     CONFIG  PWRT = ON             ; Power-up Timer Enable bit (PWRT enabled)
7     CONFIG  BOR = OFF             ; Brown-out Reset Enable bits (Brown-out Reset disabled in hardware and software)
8     CONFIG  WDT = OFF             ; Watchdog Timer Enable bit (WDT disabled (control is placed on the SWDTEN bit))
9     CONFIG  PBADEN = OFF          ; PORTB A/D Enable bit (PORTB<4:0> pins are configured as digital I/O on Reset)
10    CONFIG  LVP = OFF             ; Single-Supply ICSP Enable bit (Single-Supply ICSP disabled)
11
12      org 0x0000                  ;Vector de reset
13      goto configura
14
15      org 0x0020                  ;Zona de programa de usuario
16  configura:
17      bcf TRISD, 0                ;Puerto D0 como salida
18      bsf TRISB, 0                ;Puerto B0 como entrada
19  inicio:
20      btfss PORTB, 0
21      goto falson
22  verdaderon:
23      bcf LATD, 0
24      goto inicio
25  falson:
26      bsf LATD, 0
27      goto inicio
28
29      end
```



botón para compilar el proyecto

MPASM → *.asm

bin → *.hex

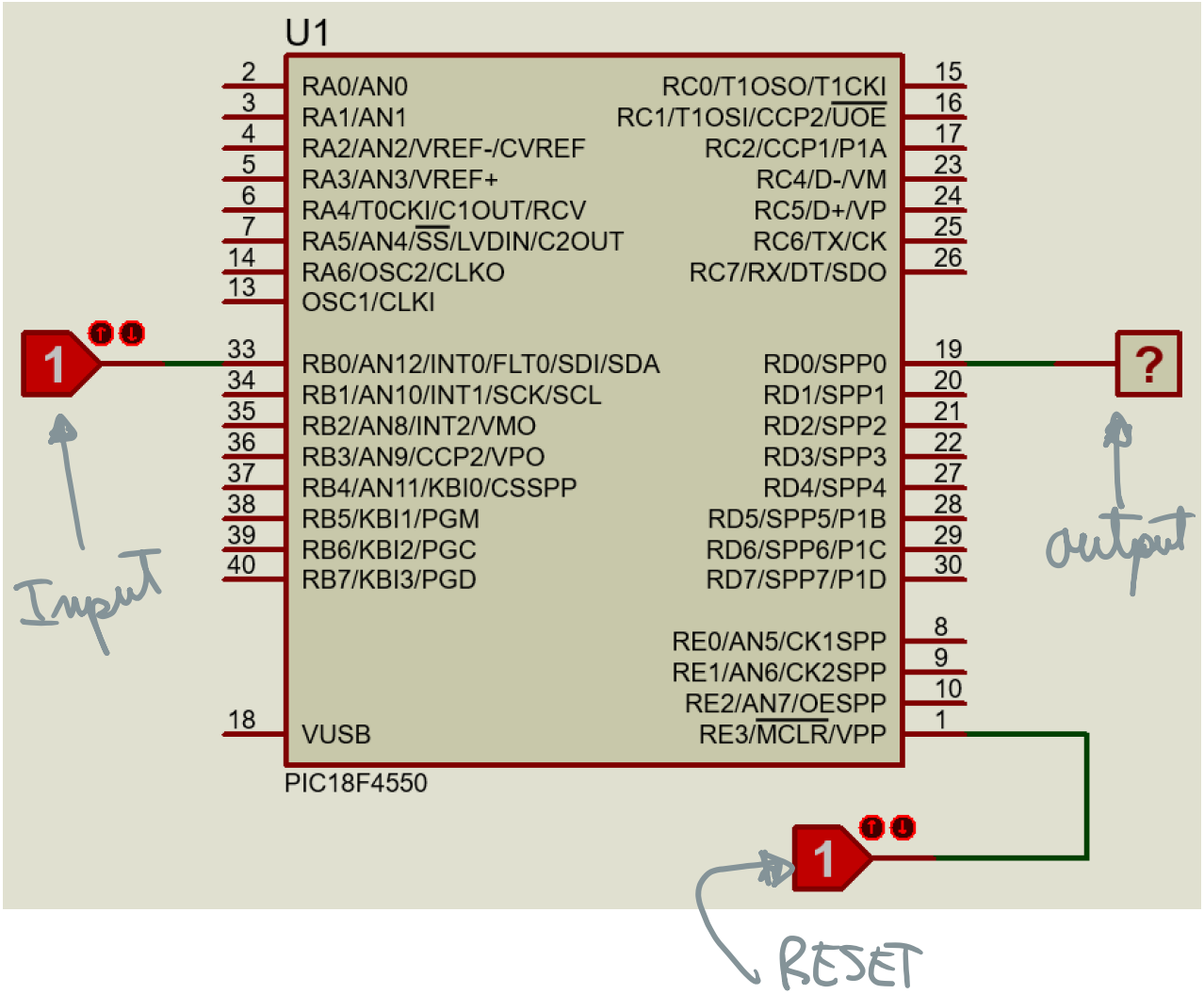
lo que se usa para grabar al PIC18F4550

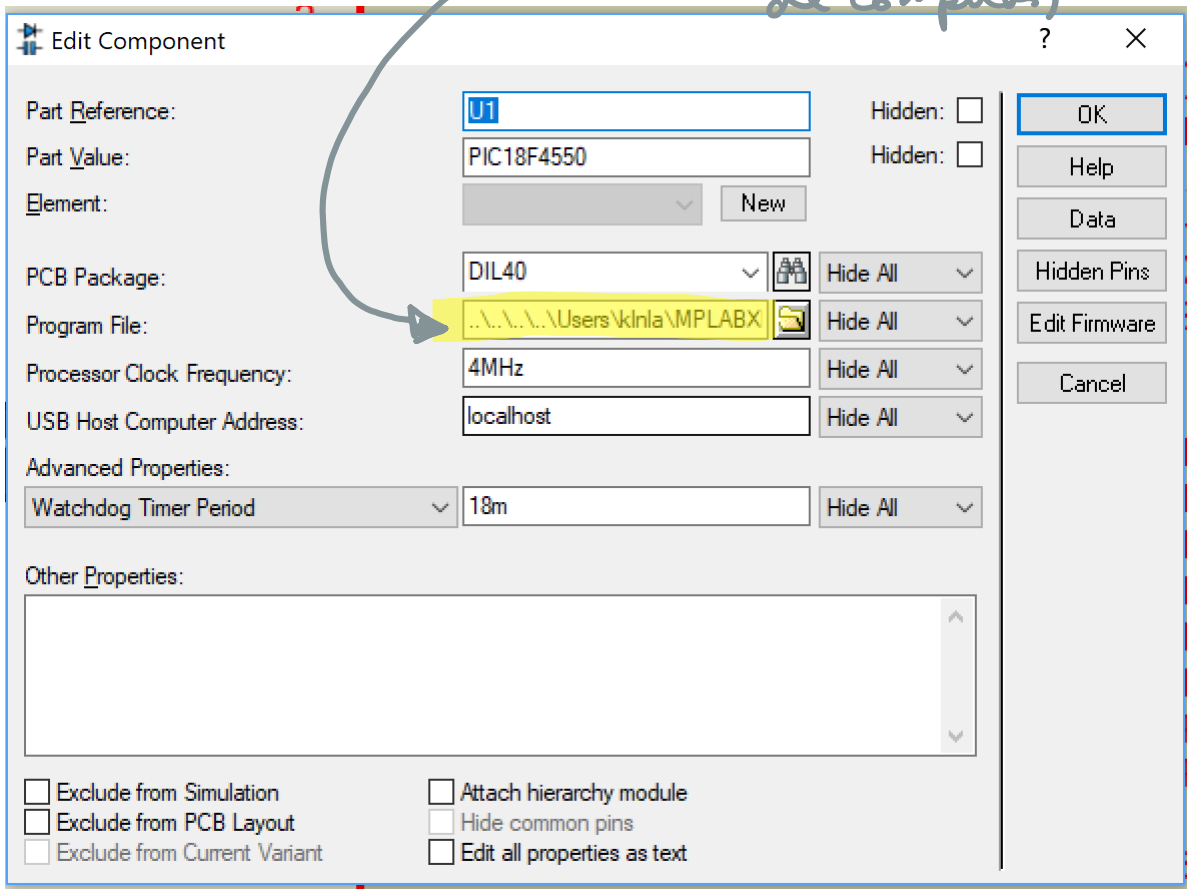
Ventana de salida (output)

```
IPLINK 5.09, LINKER
levice Database Version 1.44
:opyright (c) 1998-2011 Microchip Technology Inc.
:rrors    : 0

IP2HEX 5.09, COFF to HEX File Converter
:opyright (c) 1998-2011 Microchip Technology Inc.
:rrors    : 0

iake[2]: Leaving directory 'C:/Users/klnla/MPLABXProjects/20182_sisdig_negadeitor.X'
iake[1]: Leaving directory 'C:/Users/klnla/MPLABXProjects/20182_sisdig_negadeitor.X'

UILD SUCCESSFUL (total time: 1s)
oading code from C:/Users/klnla/MPLABXProjects/20182_sisdig_negadeitor.X/dist/default/production/20182_sisdig_negadeitor.X.production.
oading completed
```

si "build sucessful" → se creará el archivo *.hex

Simulación en Proteus VSM

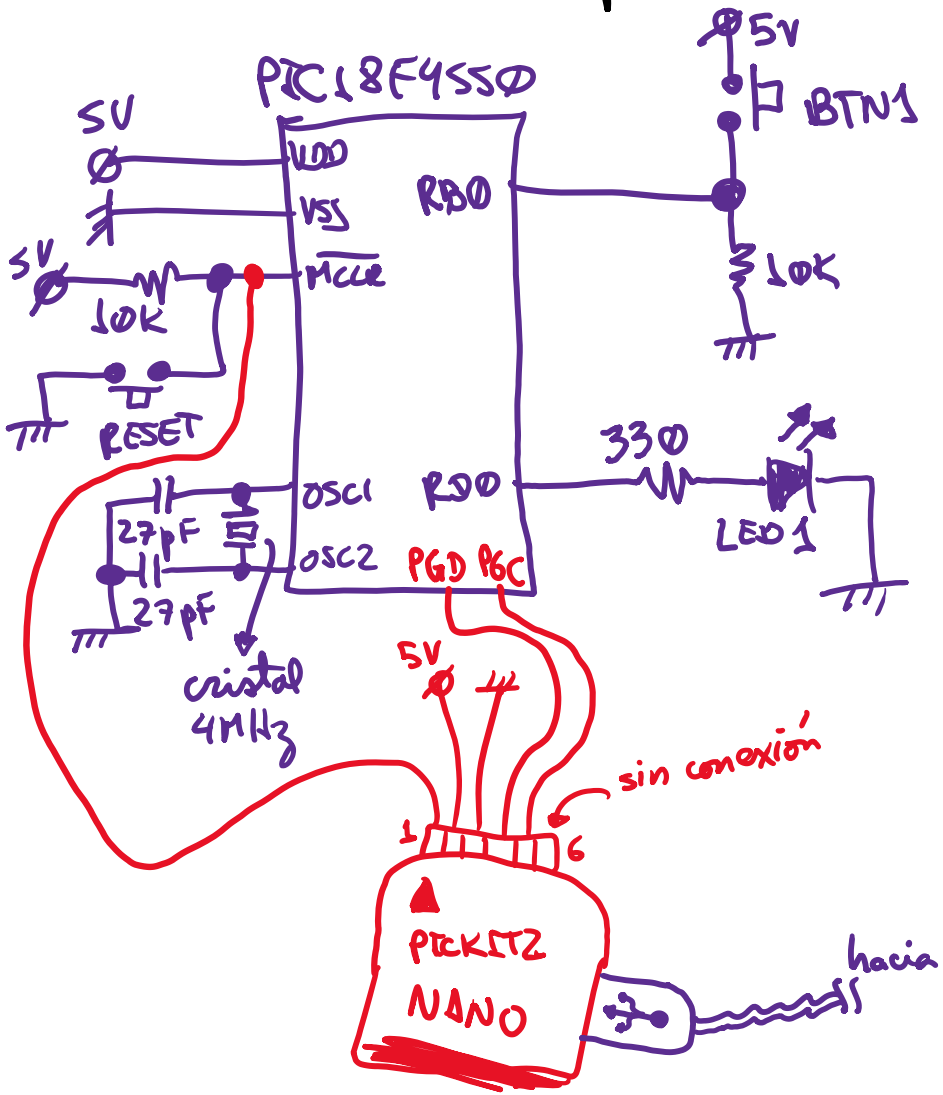


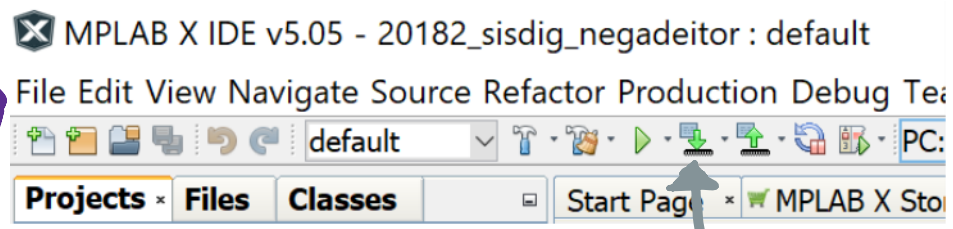


ruta donde se ha creado el archivo *.hex (luego de compilar)

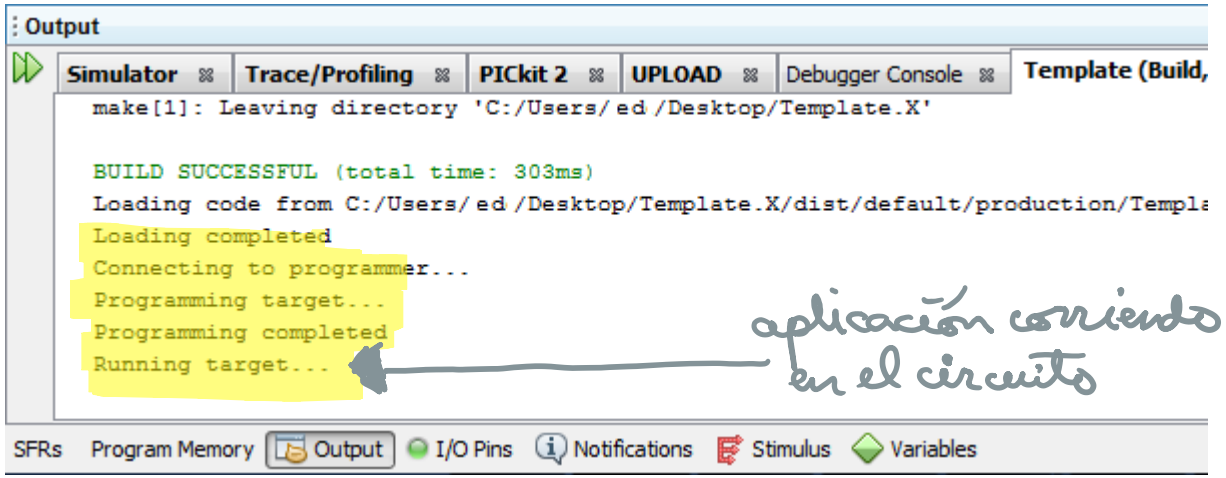
ventana de propiedades del microcontrolador dentro del diseño en Proteus (doble clic)

Conexión del PICKIT2 con el circuito de la aplicación:





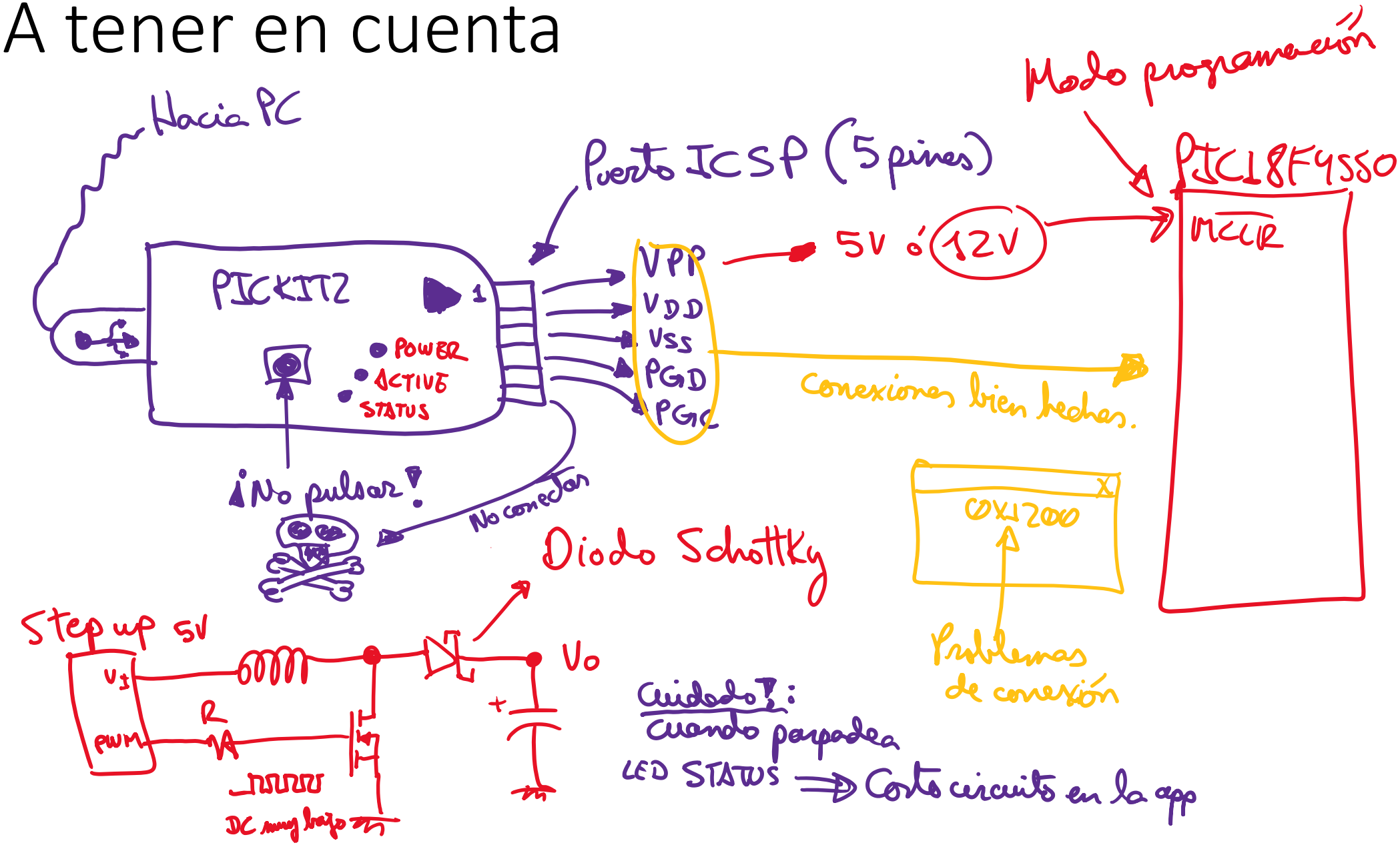
botón para grabar el *.hex en el microcontrolador usando el PICKIT2



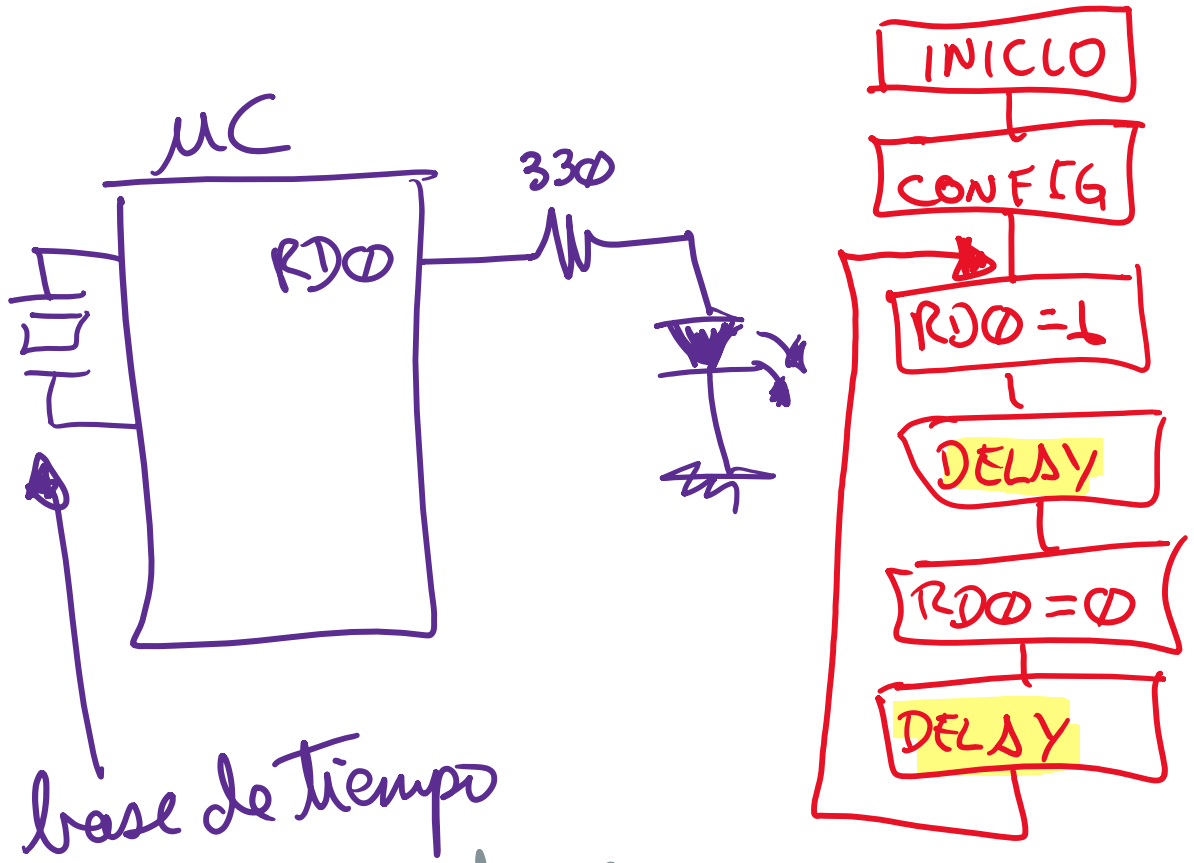
aplicación corriendo en el circuito

(verificar que el PICKIT2 se encuentre reconocido por el S.O. Windows)

A tener en cuenta



Titanicneitor: Titilar un LED



¿Ciclo de instrucción?

↳ Tiempo se demora en ejecutar una instrucción

Fosc/4

4MHz/4 = 1MHz

$T = \frac{1}{F} = 1\mu s$

⇒ "nop" demora 1µs cuando cristal = 4MHz

Titanicneitor: Titilar un LED

Requerimos generar un DELAY...

nop
nop
...

Requeriríamos 500000 nops para un retardo de 500ms...

No alcanza en la memoria de programa!

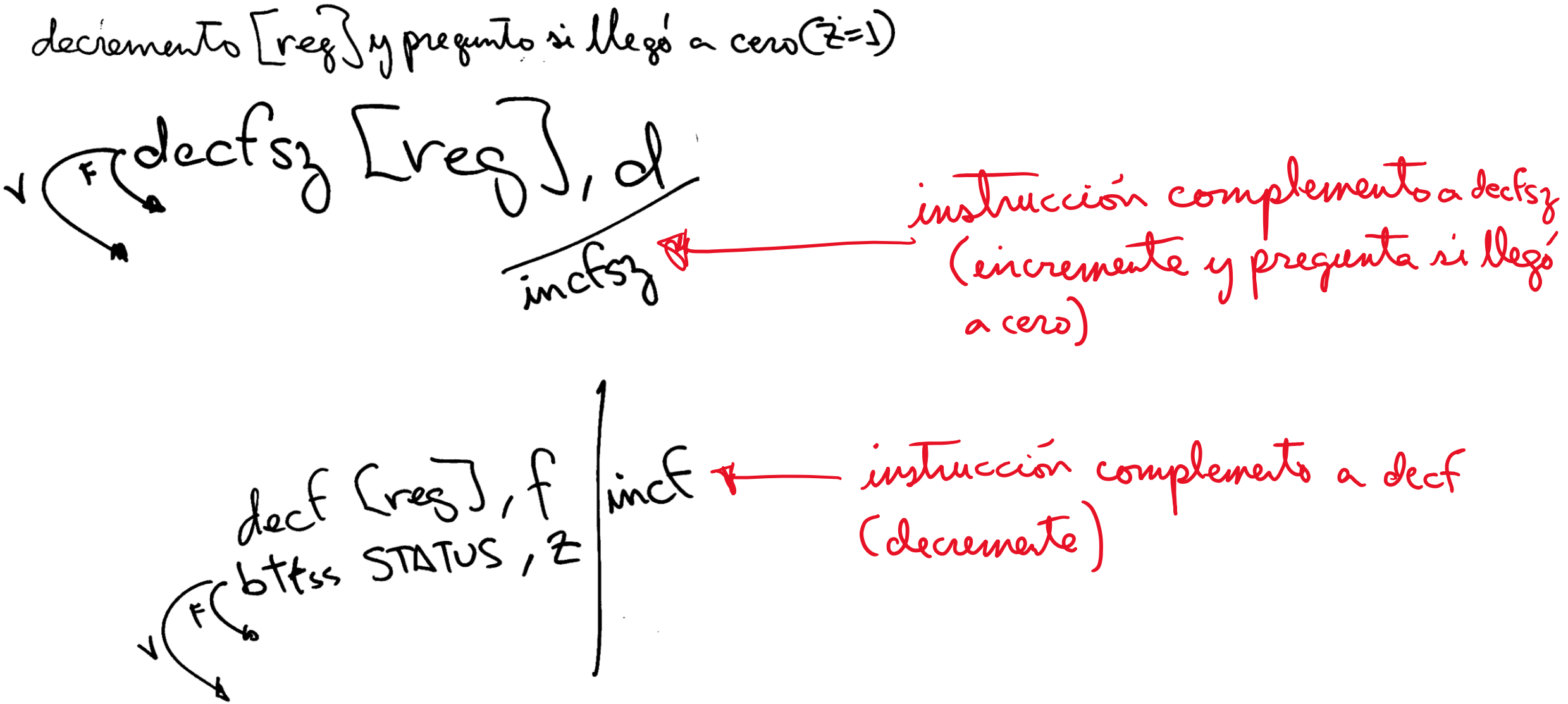
Restricción: el uC PIC18F4550 no puedes colocar 500000, solo tiene registro de 8 bits (max 255 cte)

En alto nivel:

```
For(int i=0; i<250; i++){
    For(int j=0; j<250; j++){
        nop
    }
}
```

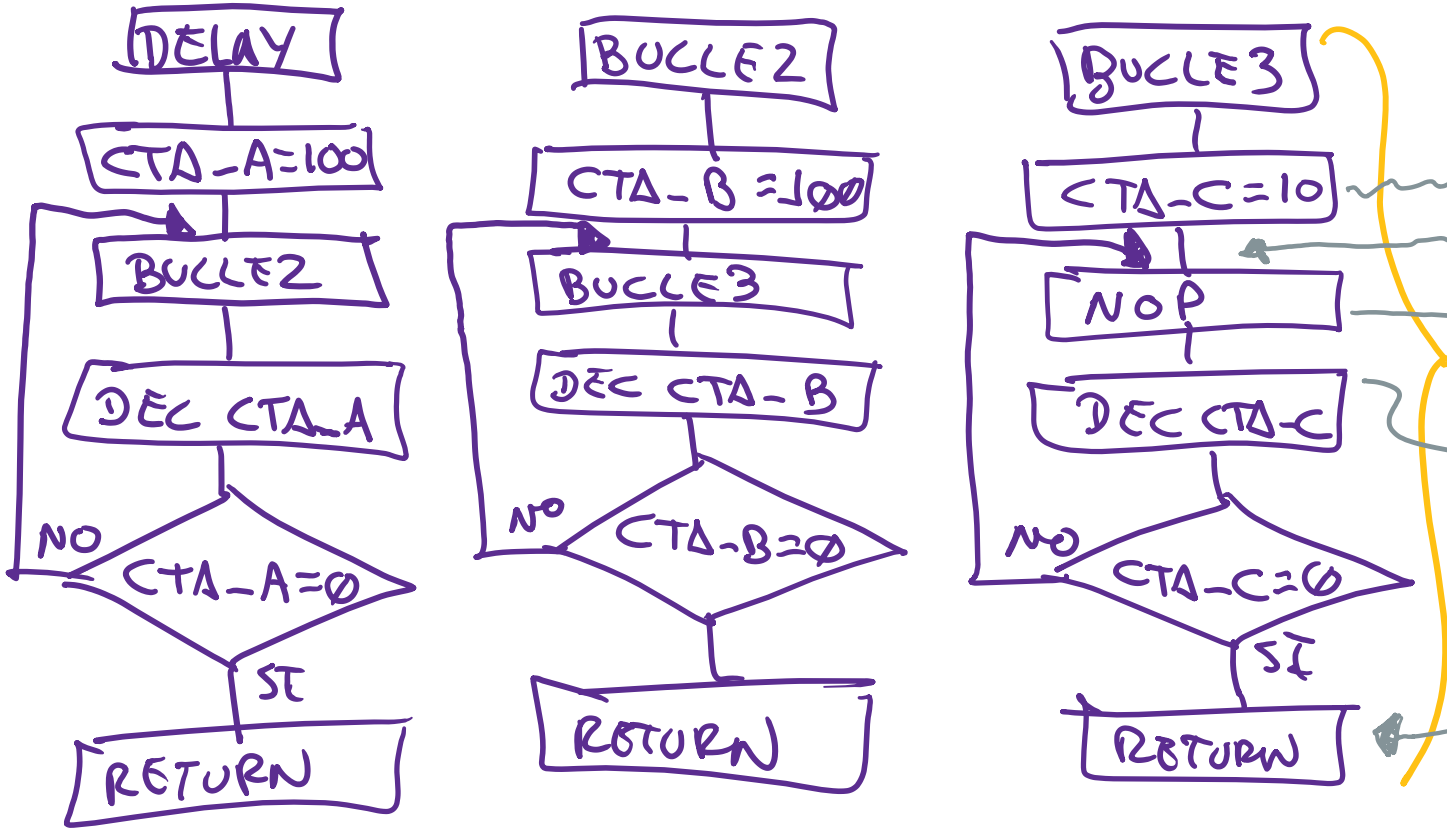
nop → 625000 veces.

Instrucciones de decremento e incremento de un registro



Titanicneitor: Titilar un LED

⇒ Necesitan 3 anillos de repetición para alcanzar DELAY ~ 500ms.



Ejemplo de codificación en base al diagrama de flujo

```
bucle3:
        movlw .10
        movwf cta_c
otro3:
        nop
        decfsz cta_c, f
        goto otro3
        return
```

Codificación en Assembler del Titanicneitor

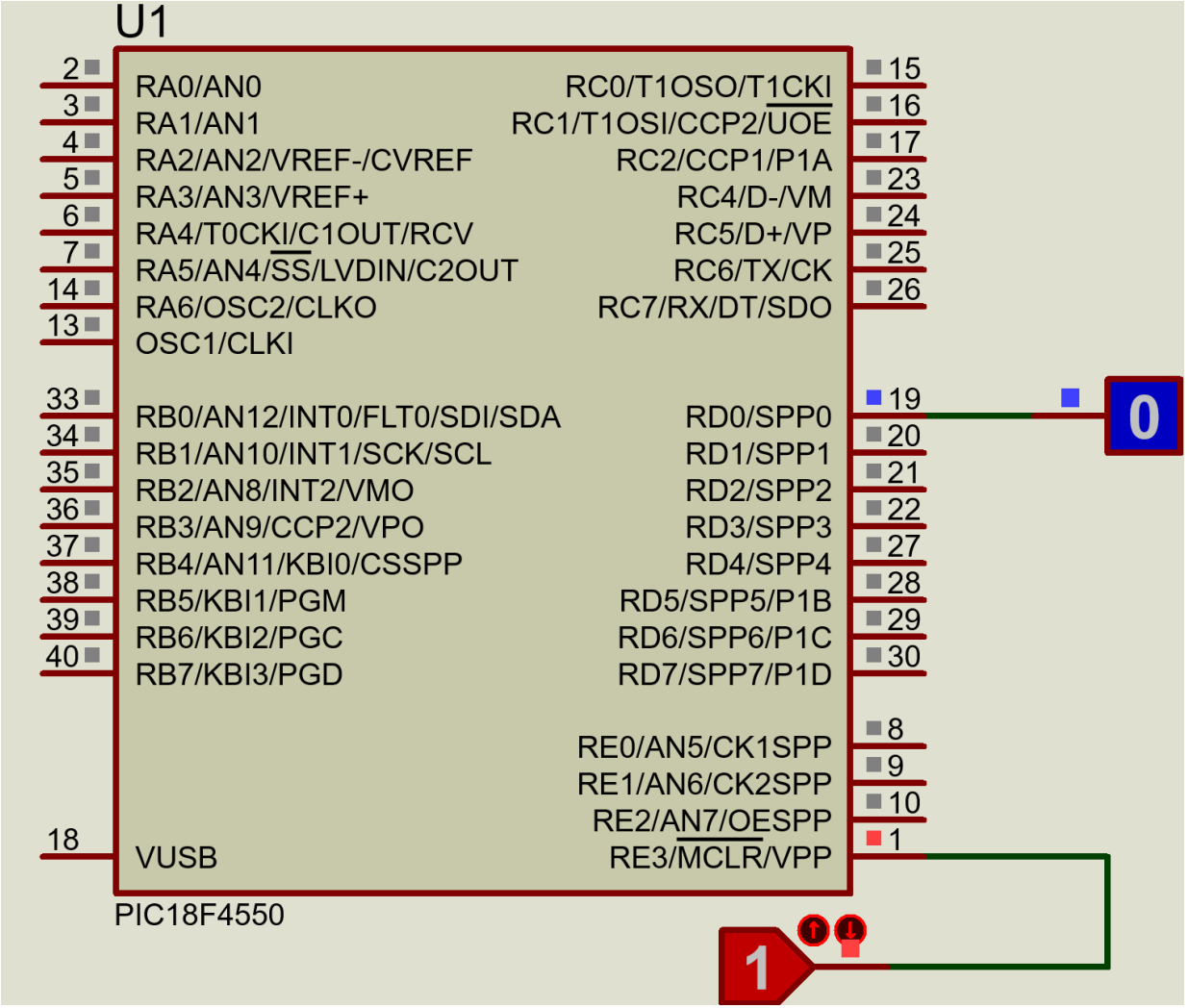
```
1       list p=18f4550              ;Modelo del microcontrolador
2       #include <p18f4550.inc>     ;librería de nombres
3
4   ;Zona de los bits de configuración del microcontroleitor
5     CONFIG  FOSC = XT_XT          ; Oscillator Selection bits (XT oscillator (XT))
6     CONFIG  PWRT = ON             ; Power-up Timer Enable bit (PWRT enabled)
7     CONFIG  BOR = OFF             ; Brown-out Reset Enable bits (Brown-out Reset d
8     CONFIG  WDT = OFF             ; Watchdog Timer Enable bit (WDT disabled (contr
9     CONFIG  PBADEN = OFF          ; PORTB A/D Enable bit (PORTB<4:0> pins are conf
10    CONFIG  LVP = OFF             ; Single-Supply ICSP Enable bit (Single-Supply I
11
12    cblock 0x0020                 ;Zona de declaración de etiquetas a los
13      cta_a                       ;registros GPR (variables)
14      cta_b
15      cta_c
16      endc
17
18      org 0x0000                  ;Vector de reset
19      goto configura
20
21      org 0x0020                  ;Zona de programa de usuario
22  configura:
23      bcf TRISD, 0                ;Para hacer que el puerto D0 sea salida
24  inicio:
25      bsf LATD, 0
26      call delaymon
27      bcf LATD, 0
28      call delaymon
29      goto inicio
```

} rutina principal

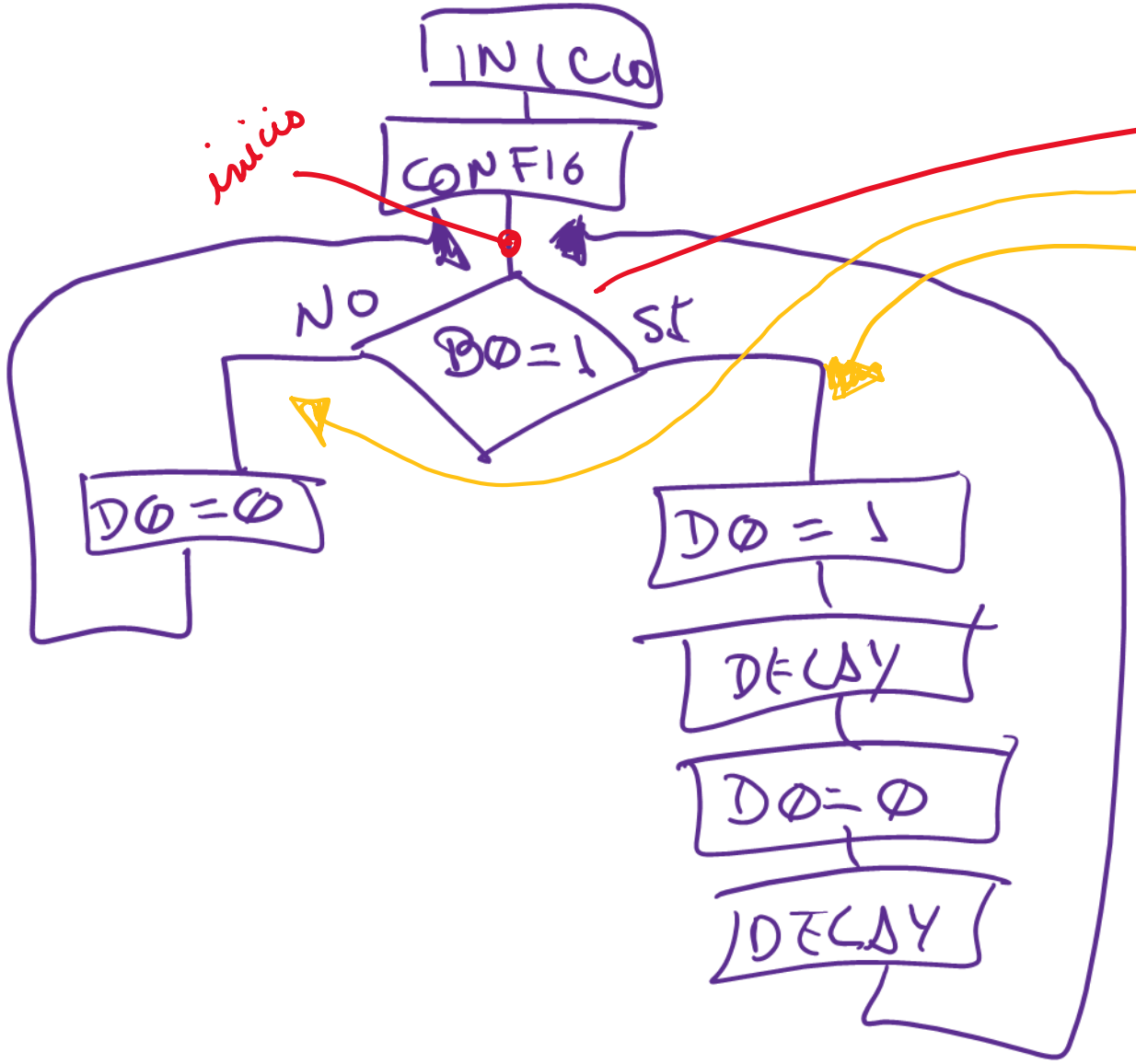
```
30
31  delaymon:
32      movlw .100
33      movwf cta_a
34  otro1:
35      call bucle2
36      decfsz cta_a, f
37      goto otro1
38      return
39
40  bucle2:
41      movlw .100
42      movwf cta_b
43  otro2:
44      call bucle3
45      decfsz cta_b, f
46      goto otro2
47      return
48
49  bucle3:
50      movlw .10
51      movwf cta_c
52  otro3:
53      nop
54      decfsz cta_c, f
55      goto otro3
56      return
57
58      end
```

Subrutina de retardo

Simulación en Proteus

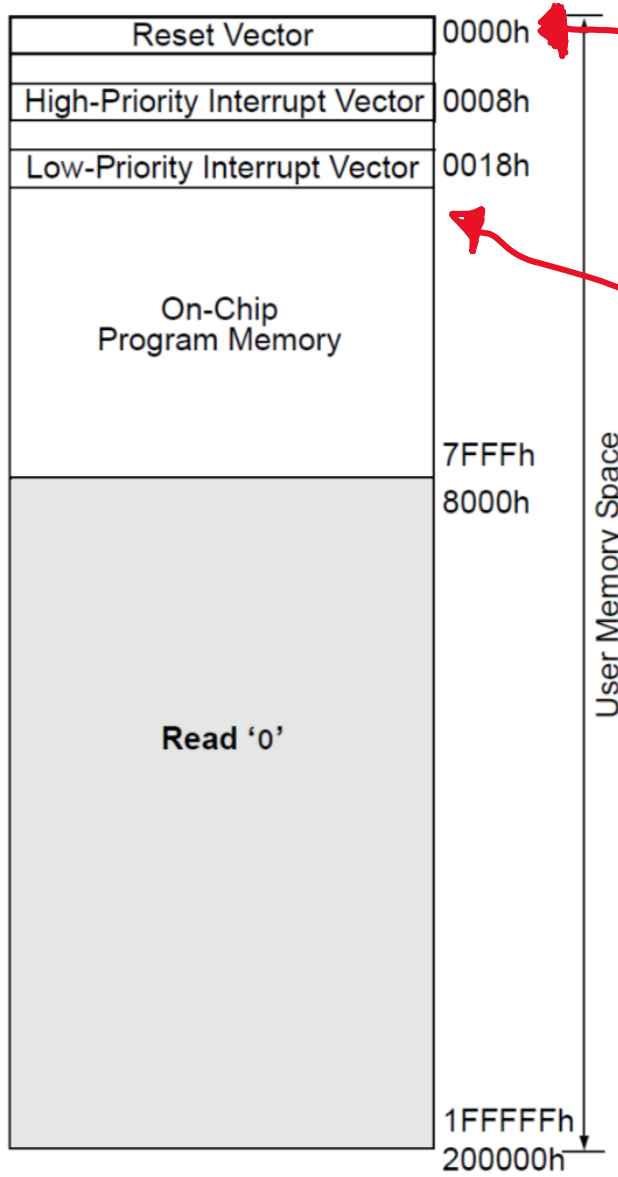


Modificando rutina inicial del titilador: Añadiendo entrada de control




```
inicio:
  btfss PORTB,0
  goto falson
  bsf LATD,0
  call retardon
  bcf LATD,0
  call retardon
  goto inicio
falson:
  bcf LATD,0
  goto inicio
```

Recordando la memoria de programa:



Reservado

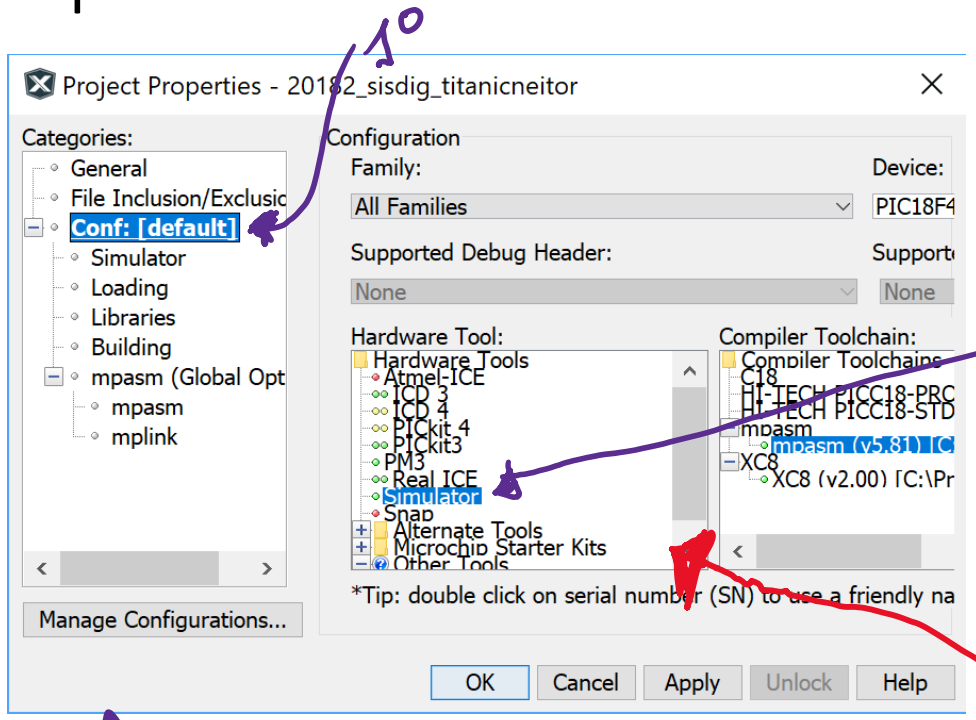
Aquí se dirige cuando hay un PoR, energizado del uC ó desborde de WDT (Watchdog Timer)

Power on Reset

A partir de aquí se escribe el programa de usuario

no implementado en el PIC18F45550

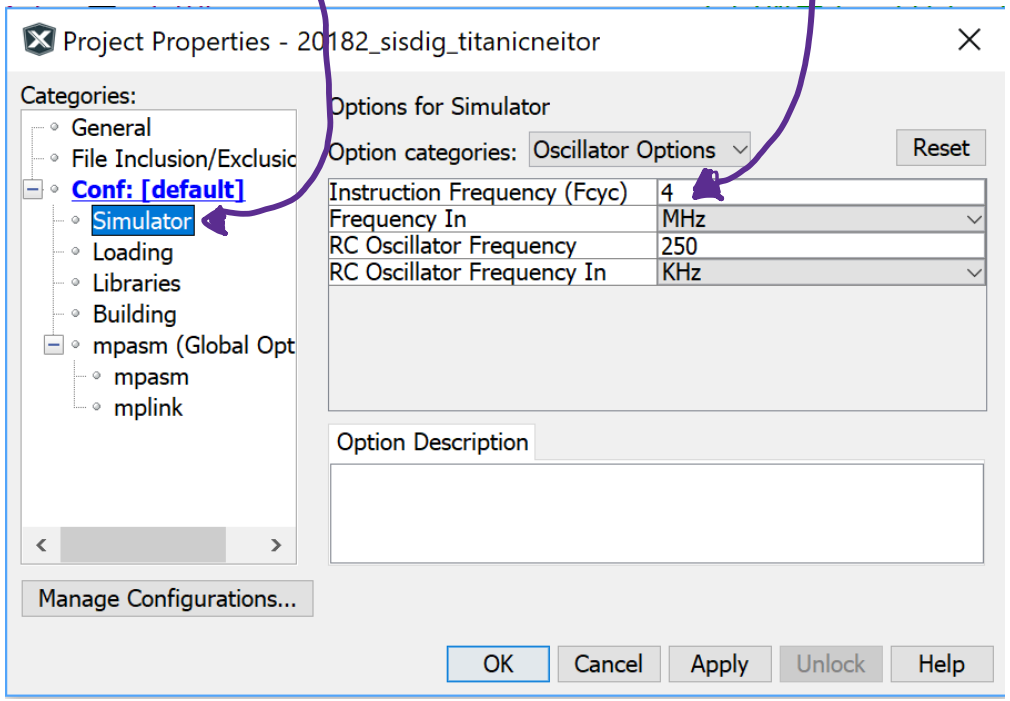
Usando el “stopwatch” del modo simulador del MPLAB X para obtener la duración de la subrutina de retardo



Seleccionas "Simulator"

2º

Frecuencia del oscilador principal



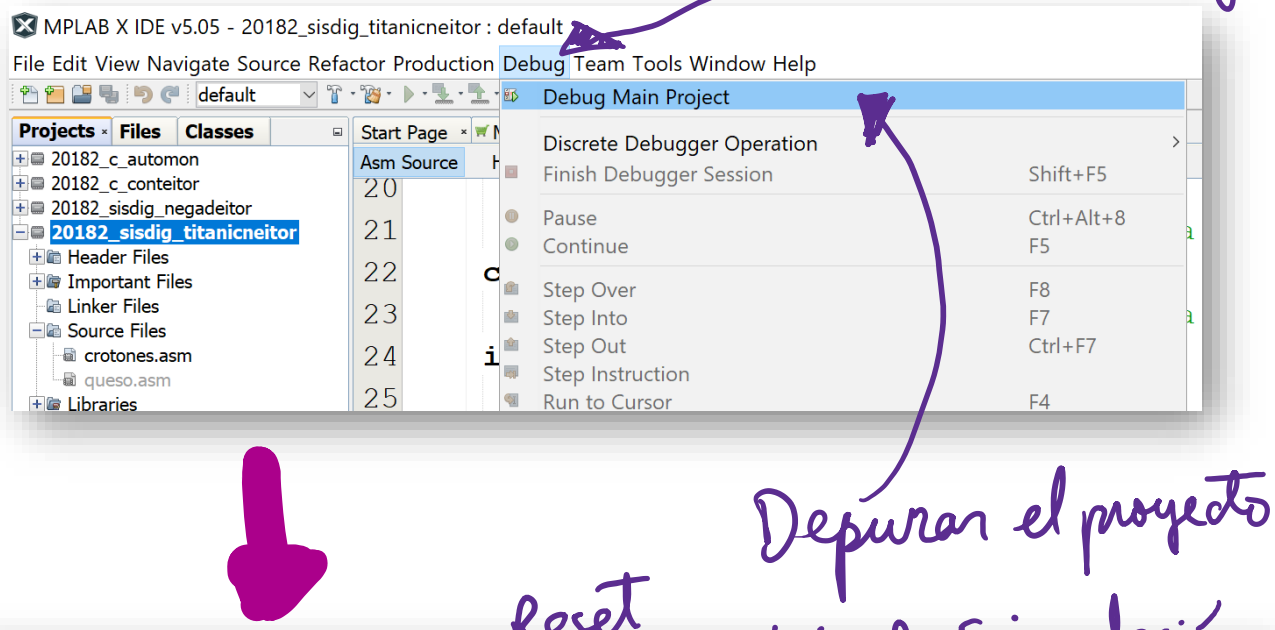
Ventana de propiedades del proyecto

Cambiar de PICKIT2 hacia Simulator !

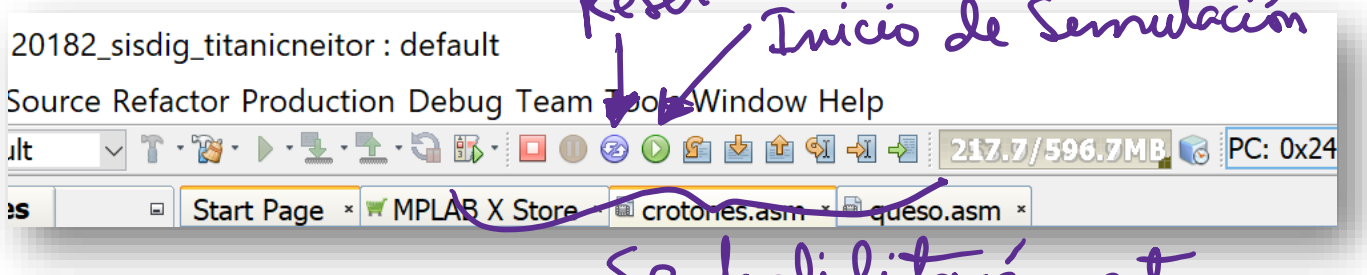
Usando el “stopwatch” del modo simulador del MPLAB X para obtener la duración de la subrutina de retardo

```
24      inicio:
25          bsf LATD, 0
26          call delaymon
27          bcf LATD, 0
28          call delaymon
29          goto inicio
30
31      delaymon:
32          movlw .100
```

Establecer los breakpoints (lugares donde se detendrá la simulación) haciendo clic en el número

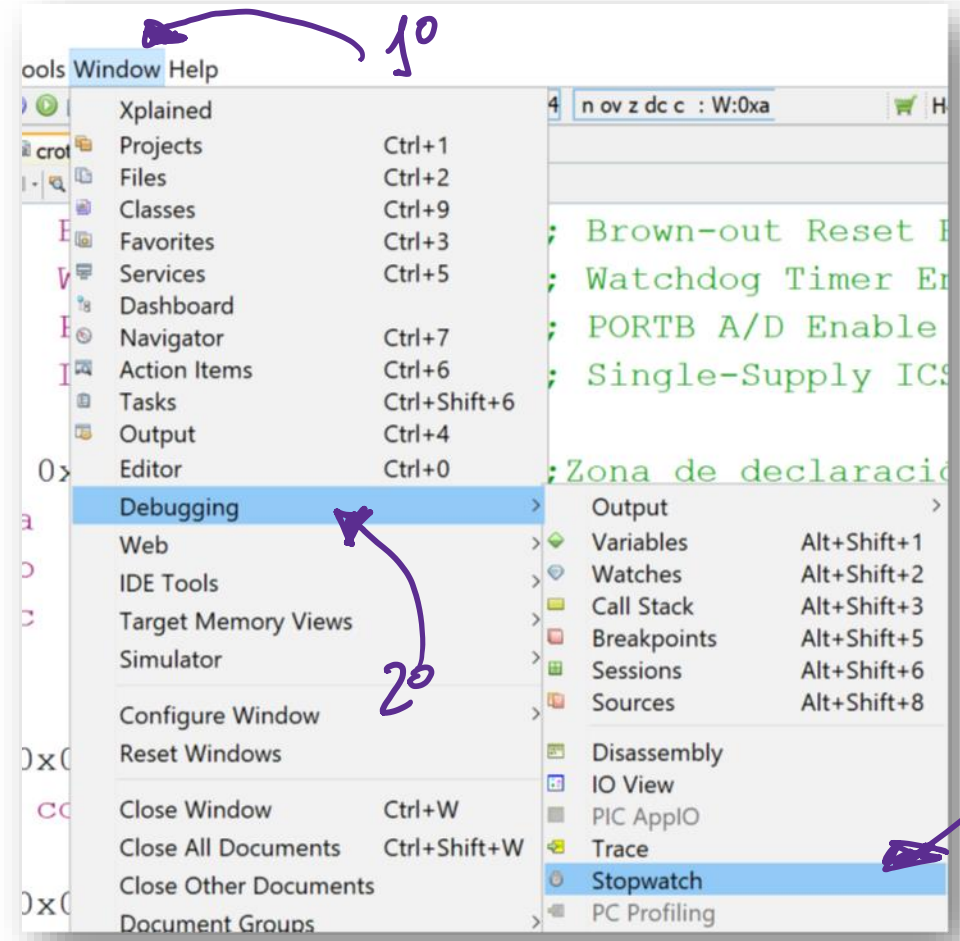


Depurar el proyecto



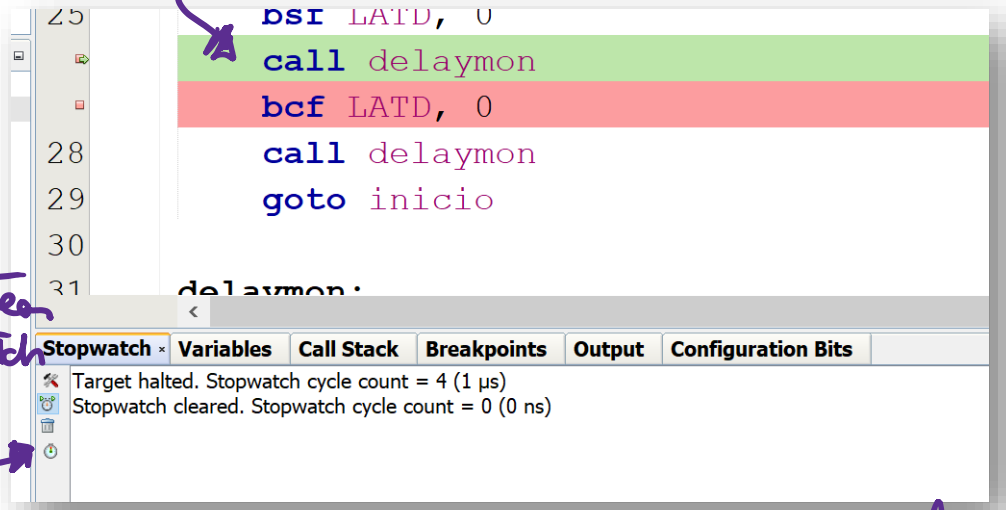
Se habilitarán estos botones

Usando el “stopwatch” del modo simulador del MPLAB X para obtener la duración de la subrutina de retardo



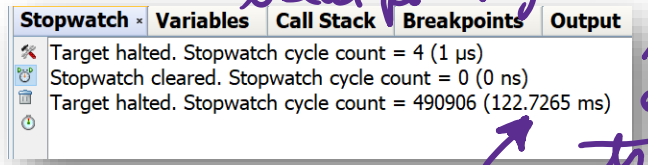
Para abrir el stopwatch

3º El puntero se detiene en el primer breakpoint



4º Reseteo el stopwatch

5º Volver a hacer clic en “Play”



6º Se detendrá en el segundo breakpoint y el stopwatch mostrará el tiempo transcurrido