

EL256 - Microcontroladores

Semana 1 Laboratorio

Semestre 2023-1

Profesor: Kalun José Lau Gan

1

Agenda

- Requerimientos de software:
 - El MPLAB X IDE v6.05
 - El XC8 v2.41
 - 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 PIC18F45K50
 - Hoja técnica del microcontrolador PIC18F57Q43
- Nuestros primeros ejemplos

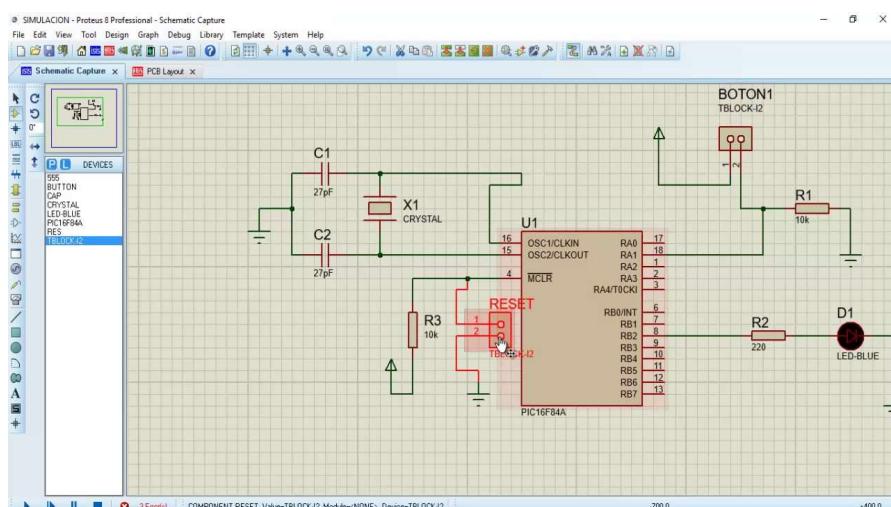
2

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 (v6.05) 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 (v2.41)
 - <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), no olvides de instalar el XC8
- 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 PIC18F45K50
- Tener en cuenta que el Proteus por el momento no tiene soporte para simular PIC18F57Q43

3

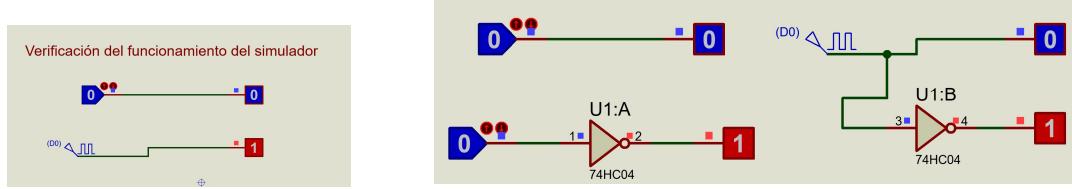
El Proteus VSM



- Simulador de circuitos

4

Verificación: Simulación en Proteus

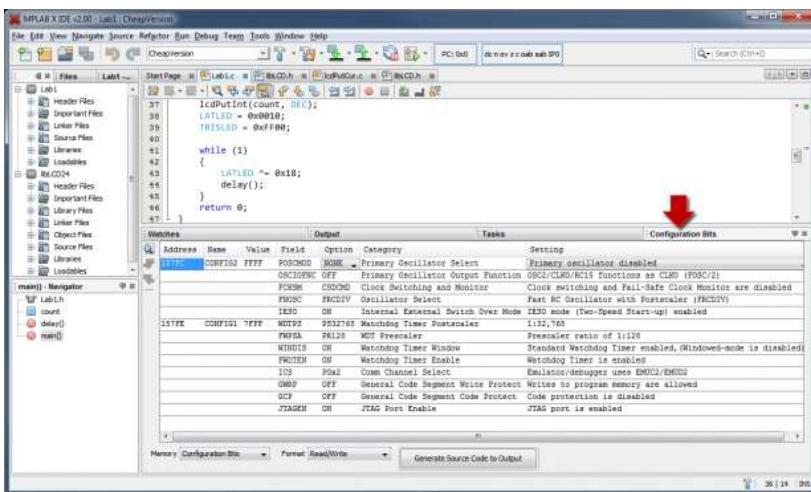


Logic state (entradas)
Logic probes (salidas)
Clock signal (señal de reloj)

} Simulación

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

MPASM vs. PICASM (XC8 Assembler)

- MPASM fué el lenguaje de programación hasta la v5.35, actualmente en **obsolescencia**.
- XC8 PIC Assembler es el nuevo formato de lenguaje y soportado por la nueva versión 6.05 del MPLAB X.
- Las instrucciones en los microcontroladores no han variado, solo la sintaxis de programación (el core del microcontrolador es el mismo).
- Evolución: 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 fidedigna.
- En nuestro caso tendremos siempre presente la hoja técnica del microcontrolador PIC18F45K50 o del PIC18F57Q43

MICROCHIP

PIC18(L)F2X/45K50

28/40/44-Pin, Low-Power, High-Performance Microcontrollers with XLP Technology

Universal Serial Bus Features:	<ul style="list-style-type: none"> USB V2.0 Compliant Cryogenicless Full Speed (12 Mb/s) and Low-Speed Operation (1.5 Mb/s) Supports Control, Interrupt, Isochronous and Bulk Transfers Supports up to 32 Endpoints (16 Bidirectional) 1 Kbyte Dual Access RAM for USB On-Chip USB Transceiver
Flexible Oscillator Structure:	<ul style="list-style-type: none"> 3x and 4xPLL Clock Multipliers Two External Clock modes, Up to 48 MHz (12 MHz) Internal 31 kHz Oscillator Secondary Oscillator using Timer1 @ 32 kHz Fail-Safe Clock Monitor Allows for safe shutdown if any clock stops
Digital-to-Analog Converter (DAC) module:	<ul style="list-style-type: none"> Fixed Voltage Reference (FVR) with 1.024V, 2.048V and 4.096V output levels 5-bit rail-to-rail resistive DAC with positive and negative reference selection
Charge Time Measurement Unit (CTMU):	<ul style="list-style-type: none"> Supports capacitive touch sensing for touch screen and button switches
Enhanced USART module:	<ul style="list-style-type: none"> Supports RS-485, RS-232 and LIN/J2602 Auto-wake-up on Start bit Auto-Baud Detect
Extreme Low-Power Management with XLP:	<ul style="list-style-type: none"> Sleep mode: 20 nA, typical Watchdog Timer: 10 nA, typical Timer1 Oscillator: 800 nA @ 32 kHz Peripheral Module Disable
Special Microcontroller Features:	<ul style="list-style-type: none"> Low-Power, High-Speed CMOS Flash Technology C Compiler Optimized Architecture for Re-entrant Code

Peripheral Highlights:

8

Revisión de documentos

- Hoja técnica del PIC18F45K50
 - <http://ww1.microchip.com/downloads/en/devicedoc/30000684B.pdf>
- Hoja técnica del PIC18F57Q43
 - <https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/PIC18F27-47-57Q43-Data-Sheet-40002147F.pdf>
- Hoja técnica del Curiosity Nano PIC18F57Q43
 - <http://ww1.microchip.com/downloads/en/DeviceDoc/PIC18F57Q43-Curiosity-Nano-HW-UserGuide-DS40002186B.pdf>

9

Procedimiento para desarrollar una aplicación con el microcontrolador PIC18F45K50

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)

10

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 XC8 PICASM los comentarios van antecedidos por un punto y coma (;)

11

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.

12

Sobre el MPLAB X IDE: Procedimiento de uso

1. Crear un proyecto (seleccionar Standalone Project)
2. Seleccionar el dispositivo microcontrolador (PIC18F45K50 ó PIC18F57Q43)
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: 

13

El microcontrolador PIC18F45K50

- Versión mejorada del PIC18F4550 y vigente en la actualidad.
- Costo menor frente al PIC18F4550 y posee mejores prestaciones tanto en velocidad (hasta 16MHz con el oscilador interno) como en consumo energético (eXtreme Low Power).
- Emplea la misma plataforma de programación con respecto a todos los microcontroladores de Microchip (ICSP).
- Mas detalles técnicos en su hoja técnica



14

Diagrama de pines del PIC18F45K50

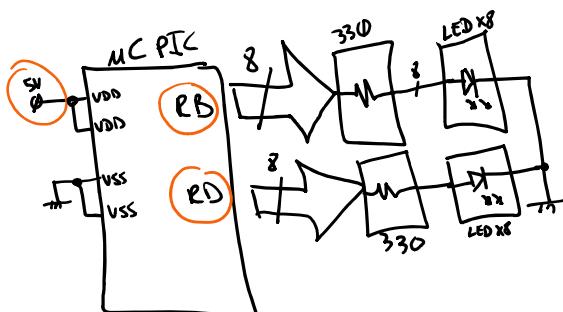
MCLR/VPP/RE3	1		40	RB7
RA0	2		39	RB6
RA1	3		38	RB5
RA2	4		37	RB4
RA3	5		36	RB3
RA4	6		35	RB2
RA5	7		34	RB1
RE0	8		33	RB0
RE1	9		32	Vdd
RE2	10		31	Vss
Vdd	11		30	RD7
Vss	12		29	RD6
RA7	13		28	RD5
RA6	14		27	RD4
RC0	15		26	RC7
RC1	16		25	RC6
RC2	17		24	D+
Vusb3v3	18		23	D-
RD0	19		22	RD3
RD1	20		21	RD2

PIC18(L)F45K50

- No hay RC3, RC4 ni RC5
- Dos pines para Vdd y Vss
- ~MCLR: Master RESET, activo en bajo
- RA, RB y RD completos (8bits cada uno)
- Solo hay RE(3:0)
- Algunos puertos de E/S son solo entradas o solo salidas (revisar hoja técnica)
- Por defecto en un Power On Reset los puertos de E/S son entradas analógicas

15

Hay dos pines de Vdd y dos pines de Vss, es necesario conectar todos?

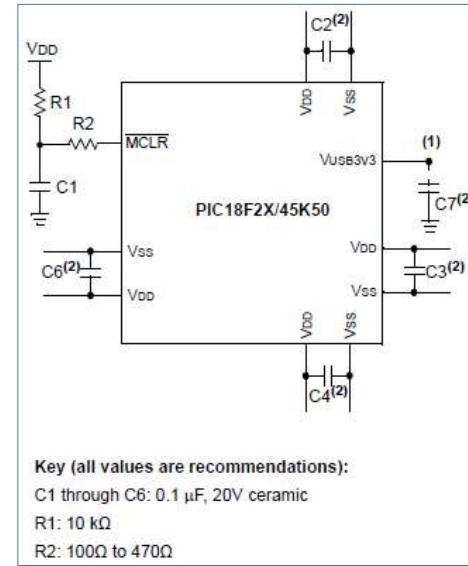


Se deben de conectar todos los pines de alimentación para que el microcontrolador pueda obtener mayor capacidad de corriente en caso lo requiera la aplicación.

16

Detalles técnicos iniciales

- Voltaje de alimentación del PIC18F45K50
 - Voltajes menores a -0.5V son perjudiciales
 - Voltaje de operación máximo 5.5V
 - Voltaje de operación mínimo 2.3V



17

Los bits de configuración (configuration bits)

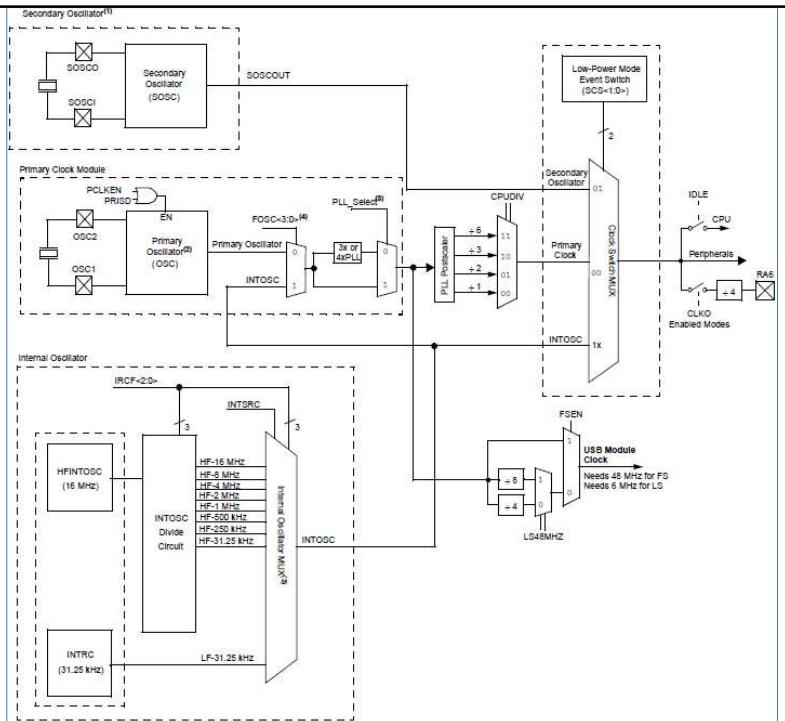
- Son las configuraciones iniciales del microcontrolador, fundamentales para un adecuado funcionamiento al momento de energizar.
- Dichos bits de configuración serán contemplados en el archivo cabecera o header (con extensión *.inc) dentro del proyecto creado en el MPLAB X
- Algunos de los parámetros a configurar:
 - Fuente de reloj
 - Habilitación de la entrada de reset (MCLR)
 - Brown-out reset
 - Power-up timer
 - Watchdog timer

18

Detalles técnicos iniciales

- Fuente de reloj

- Oscilador secundario sirve para colocar un cristal de 32.768KHz y hacer aplicaciones de reloj en tiempo real con el Timer1.
- El PLL de este microcontrolador sirve para incrementar la frecuencia de trabajo, hasta 48MHz. El PLL trabaja desde 8MHz en adelante.
- Usaremos el oscilador interno para todas nuestras aplicaciones.



19

Configuración inicial de fuente de reloj

Para obtener 4MHz al CPU y periféricos a partir del oscilador interno:

1. Habilitaremos el funcionamiento del oscilador interno (HFINTOSC) de 16MHz – Esto se hace en el bit de configuración FOSC=INTOSCCLKO
2. Configuraremos el divisor para obtener 4MHz (IRCF) en el registro OSCCON
3. Seleccionamos INTSRC para que pase los 4MHz que salen del divisor (registro OSCCON2)
4. Seleccionaremos como fuente de reloj a INTOSC (SCS 1x) en el registro OSCCON

20

Configuración inicial de fuente de reloj

Registro OSCCON:01010010 = 52H

- IRCF en 101 para 4MHz

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	R/W-0
IDLEN		IRCF<2:0>		OSTS ⁽¹⁾	HFIOS	SCS<1:0>	
bit 7							bit 0
Legend:							
R = Readable bit -n = Value at POR	W = Writable bit '1' = Bit is set	U = Unimplemented bit, read as '0' '0' = Bit is cleared	q = depends on condition x = Bit is unknown				
bit 7	IDLEN: Idle Enable bit 1 = Device enters Idle mode on SLEEP instruction 0 = Device enters Sleep mode on SLEEP instruction						
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	HFIOS: 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).						
Note 1: Reset state depends on state of the IESO Configuration bit. 2: Default output frequency of HFINTOSC on Reset.							

21

Configuración inicial de fuente de reloj

Registro OSCCON2:00000000 = 00H

- Desactivamos oscilador primario
- Nos aseguramos que INTSRC se encuentre en 0

REGISTER 3-2: OSCCON2: OSCILLATOR CONTROL REGISTER 2							
R-0/0	R-0/q	R/W-0	R/W-0/0	R/W-0/u	R/W-1/1	R-0/0	R-0/0
PLLRDY	SOSCRUN	INTSRC	PLLEN	SOSCGO ⁽¹⁾	PRISD	HFIOPR	LFIOPS
bit 7							bit 0
Legend:							
R = Readable bit '1' = Bit is set -n/n = Value at POR and BOR/Value at all other Resets	W = Writable bit '0' = Bit is cleared	U = Unimplemented bit, read as '0' 'x' = Bit is unknown	q = depends on condition				
bit 7	PLLRDY: PLL Run Status bit 1 = System clock comes from PLL 0 = System clock comes from an oscillator, other than PLL						
bit 6	SOSCRUN: SOSC Run Status bit 1 = System clock comes from secondary SOSC 0 = System clock comes from an oscillator, other than SOSC						
bit 5	INTSRC: HFINTOSC Divided by 512 Enable bit 1 = HFINTOSC used as the 31.25 kHz system clock reference – high accuracy 0 = INTRC used as the 31.25 kHz system clock reference – low power.						
bit 4	PLLEN: Software PLL Enable bit If FOSC<3:0> = 100x, 010x or 001x 1 = PLL enabled 0 = PLL disabled Else, No effect on PLL operation.						
bit 3	SOSCGO⁽¹⁾: Secondary Oscillator Start Control bit 1 = Secondary oscillator is enabled. 0 = Secondary oscillator is shut off if no other sources are requesting it.						
bit 2	PRISD: Primary Oscillator Drive Circuit Shutdown bit 1 = Oscillator drive circuit on 0 = Oscillator drive circuit off (zero power)						
bit 1	HFIOPR: HFINTOSC Status bit 1 = HFINTOSC is running 0 = HFINTOSC is not running						
bit 0	LFIOPS: INTRC Frequency Stable bit 1 = INTRC is stable 0 = INTRC is not stable						
Note 1: The SOSCGO bit is only reset on a POR Reset.							

22

Configuración inicial de fuente de reloj

Registro OSCTUNE:00000000 = 00H

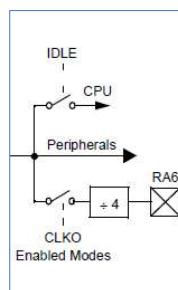
- Sirve para la calibración fina de la frecuencia de salida del HFINTOSC, generalmente no se tiene que modificar.
- Se tiene también el selector de configuración del PLL que no lo modificamos si es que no usamos el PLL

REGISTER 3-3: OSCTUNE: OSCILLATOR TUNING REGISTER							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPLLMULT	TUN<6:0>						
bit 7	bit 0						
Legend:							
R = Readable bit -n = Value at POR	W = Writable bit '1' = Bit is set	U = Unimplemented bit, read as '0' '0' = Bit is cleared	x = Bit is unknown				
bit 7 SPLLMULT: Software PLL Multiplier Select bit If PLL Enabled, SPLLMULT changes are ignored. Else, Selects which PLL multiplier will be used: 1 = 3xPLL is selected 0 = 4xPLL is selected							
bit 6-0 TUN<6:0>: Frequency Tuning bits – affects HFINTOSC ⁽¹⁾ 0111111 = Maximum frequency 0111110 = ... 0000001 = 0000000 = Center frequency. Oscillator module is running at the factory calibrated frequency. 1111111 = ... 1000000 = Minimum frequency							

23

Configuración del módulo de oscilador

- Se tiene la opción de CLKO para poder obtener la señal de reloj dividido entre 4 por el puerto RA6 para efectos de medir la frecuencia de manera externa con un instrumento ya sea un contómetro de frecuencia o un oscilloscopio, y verificar si se está obteniendo la frecuencia deseada.



24

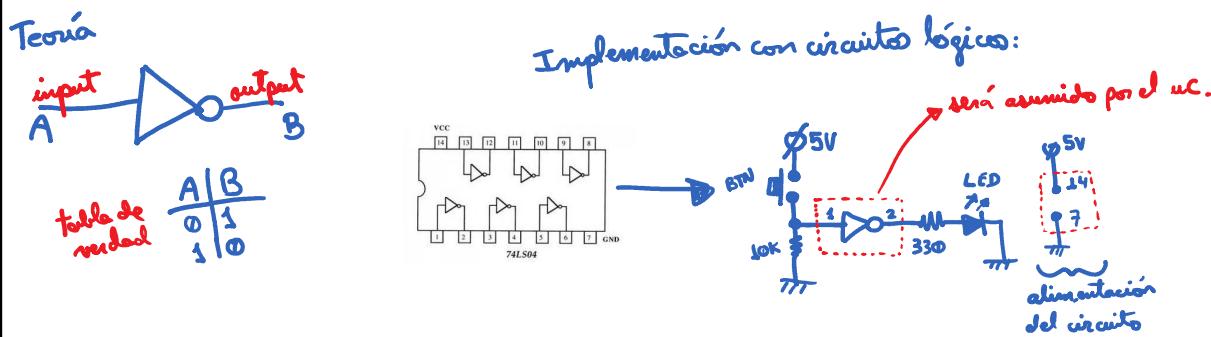
Manejo de puertos de E/S en el PIC18F45K50

- Revisar 11.0 en la hoja técnica
- Se tienen los siguientes registros para manipular los puertos:
 - TRISx Para configurar el sentido del Puerto (entrada ó salida), cero para salida y uno para entrada
 - PORTx Para leer el puerto
 - LATx Para escribir el puerto
 - ANSELx Para configurar el puerto en analógico o digital, uno para analógico y cero para digital.
 - SLRCON Para configurar la velocidad de respuesta en el puerto configurado como salida.

25

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

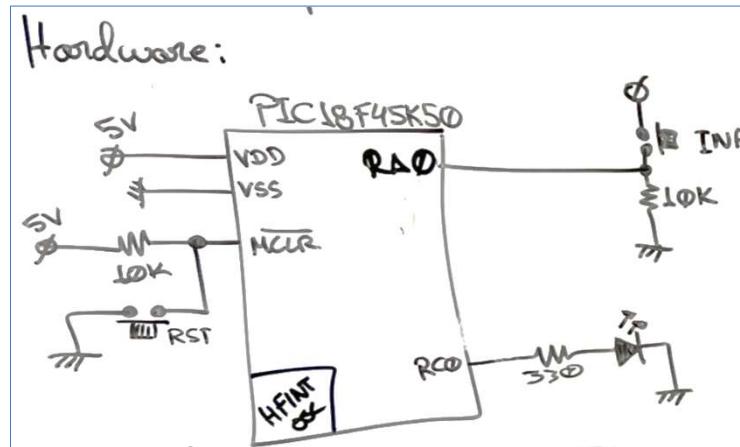
- Análisis



26

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

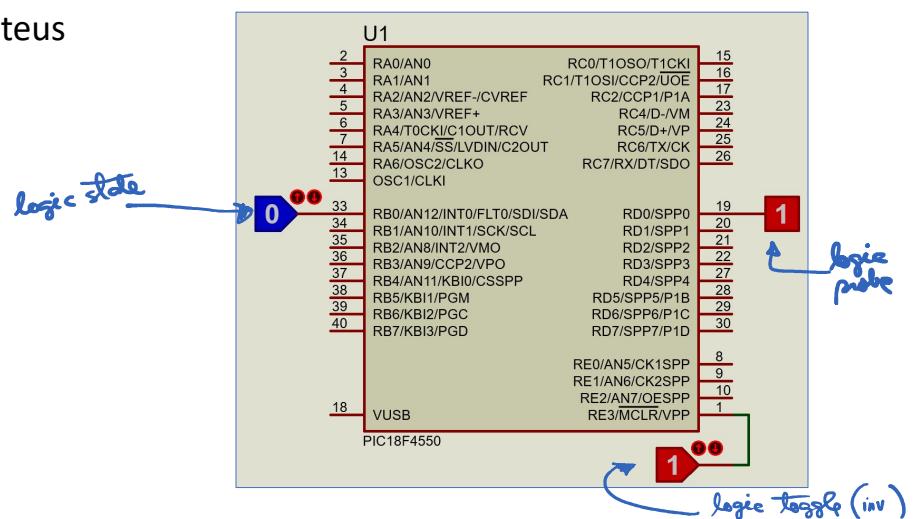
- Diagrama esquemático del circuito de la aplicación



27

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

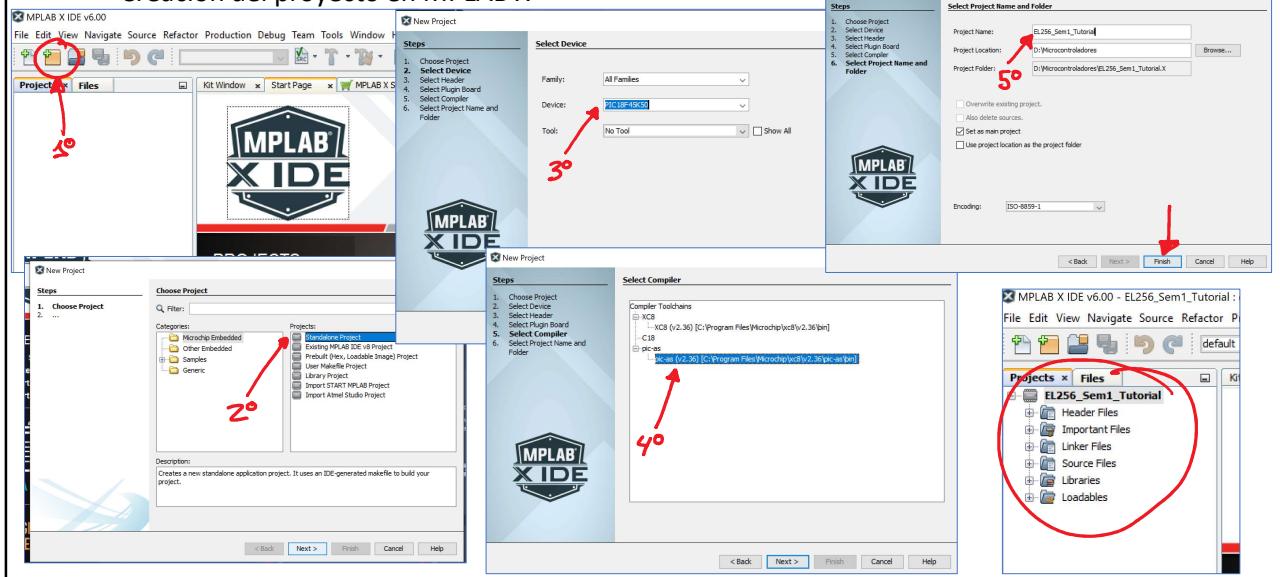
- Circuito en Proteus



28

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

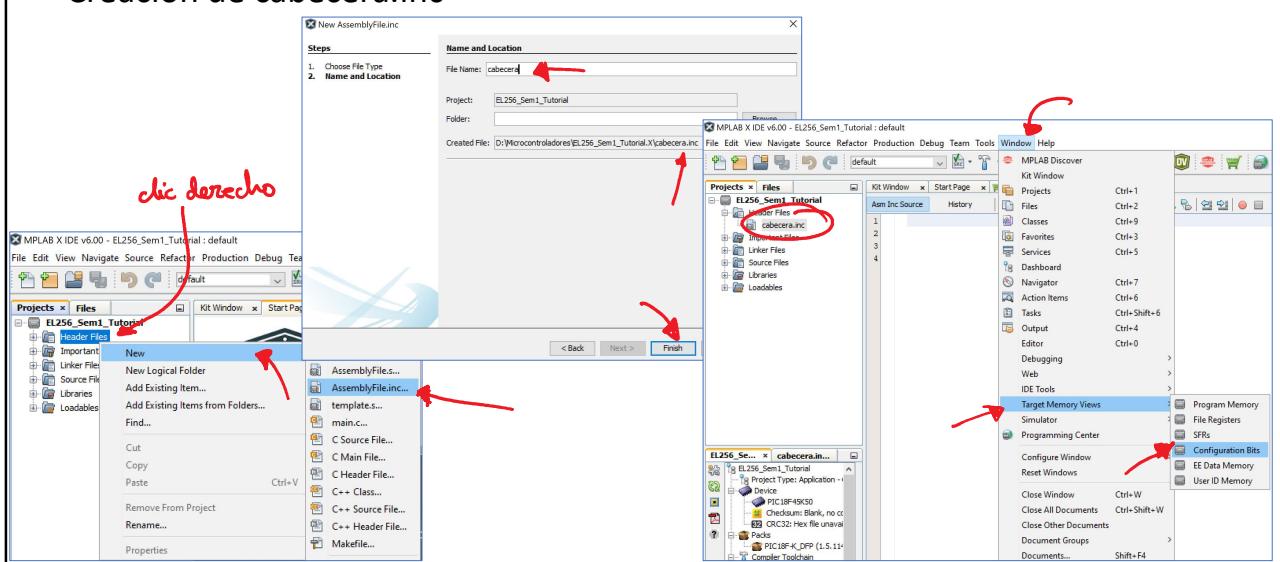
- Creación del proyecto en MPLAB X



29

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

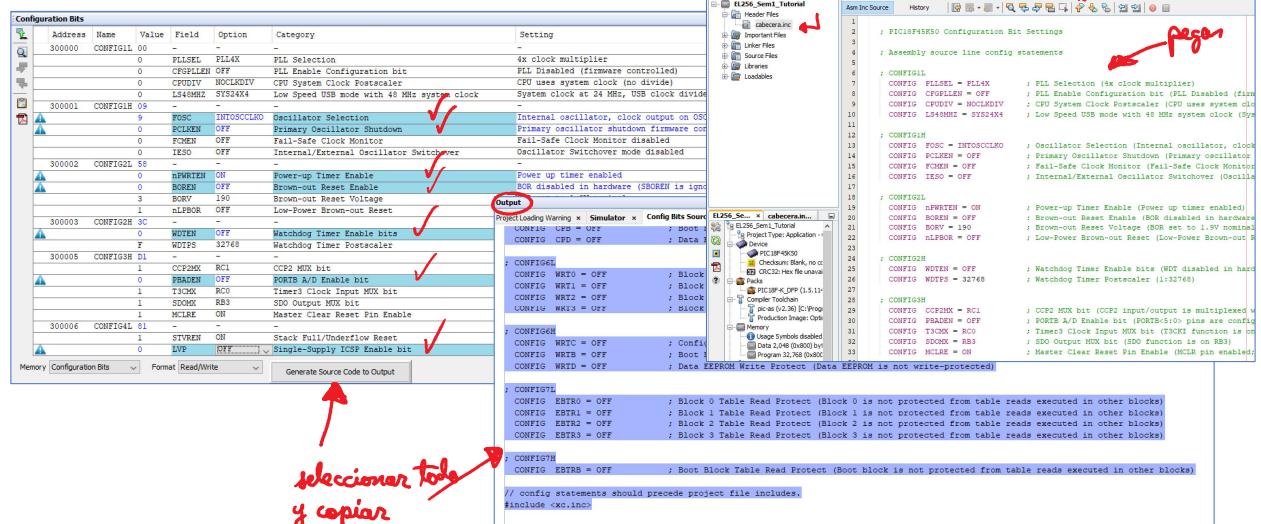
- Creación de cabecera.inc



30

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

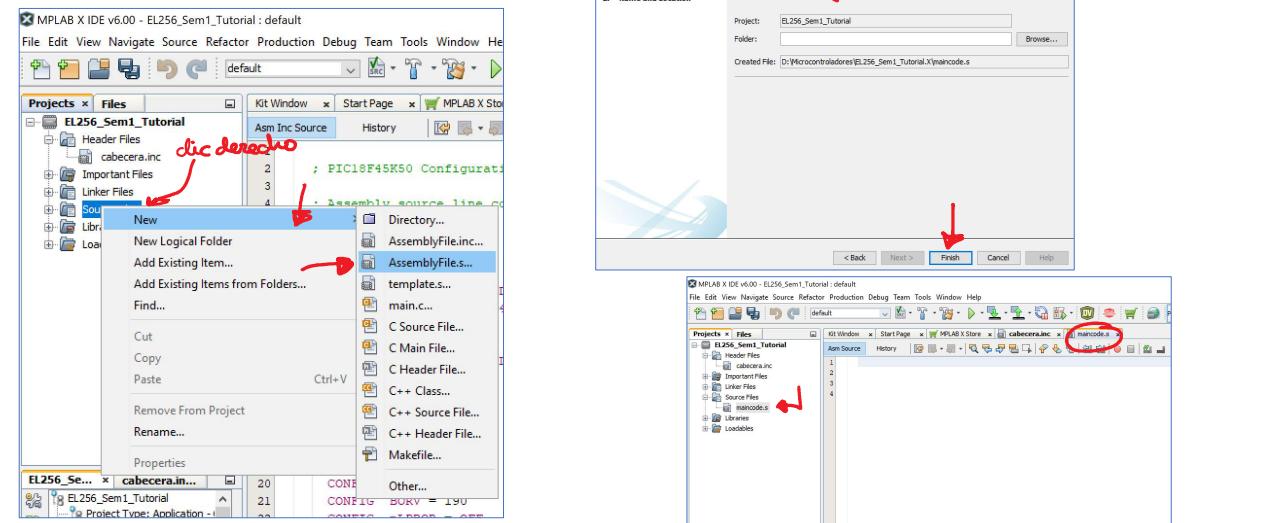
- Creación de cabecera.inc



31

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

- Creación de maincode.s



32

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

- Plantilla de maincode.s para todos los ejemplos

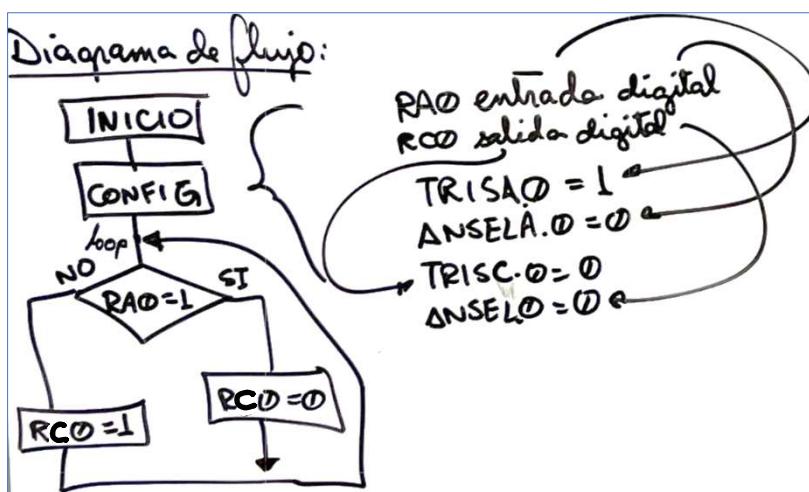
```

1 ;Este es un comentario
2 ;Ejemplo hecho por Kalun ps
3      PROCESSOR 18F45K50           ;modelo de micro
4      #include "cabecera.inc"     ;llamada a cabecera
5
6      PSECT upcino, class=CODE, reloc=2, abs  ;program section
7      upcino:                      ;etiqueta upcino
8          ORG 000000H              ;vector de reset
9          goto configuro         ;salto a etiqueta configuro
10
11         ORG 000020H              ;zona de prog de usuario
12         configuro:             ;etiqueta configuro
13             ;aqui van los registros de conf iniciales
14             movlb 0FH              ;bank15 al access bank
15             movlw 52H
16             movwf OSCCON           ;INTOSC a 4MHz
17
18         loop:                   ;etiqueta loop
19             ;programa de usuario
20
21             goto loop             ;salto a etiqueta loop
22
23         end upcino              ;fin de program section
24

```

33

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz



34

Ejemplo: Desarrollar un negador lógico con el microcontrolador PIC18F45K50 con FOSC=4MHz

- Código en XC8 Assembler

- Código en XC8 Assembler

cabecera.inc

maincode.s

```

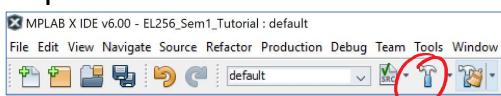
1 ; CONFIG
2 CONFIG PLLSEL = PLL4X ; PLL Selection (4x clock multipli
3 CONFIG CFGPLLLEN = OFF ; PLL Enable Configuration bit (PI
4 CONFIG CPUFUDIV = NOCLKDIV ; CPU System Clock Postscaler (CPU
5 CONFIG LS48MHZ = SYS24X4 ; Low Speed USB mode with 48 MHz
6
7 ; CONFIGAF
8 CONFIG FOSC = INTOSCCLK0 ; Oscillator Selection (Internal 0
9 CONFIG PCLEEN = OFF ; Primary Oscillator Shutdown (P
10 CONFIG FCMEN = OFF ; Fail-Safe Clock Monitor (Fail-S
11 CONFIG IESO = OFF ; Internal/External Oscillator Sele
12
13 ; CONFIGF
14 CONFIG RPWREN = ON ; Power-up Timer Enable (Power up
15 CONFIG BOREN = OFF ; Brown-out Reset Enable (BOR dis
16 CONFIG BORU = 190 ; Brown-out Reset Voltage (BOR set
17 CONFIG RPLBOR = OFF ; Low-Power Brown-out Reset (Low-P
18
19 ; CONFIGR
20 CONFIG WDTEEN = OFF ; Watchdog Timer Enable bits (WDT
21 CONFIG WDTPS = 32768 ; Watchdog Timer Postscaler (1:32)
22
23 ; CONFIGC
24 CONFIG CCP2MX = RCL ; CCP2 MUX bit (CCP2 input/output
25 CONFIG PFRDEN = OFF ; PORTB A/D Enable bit (PORTB<5:0)
26 CONFIG T2CKX = RCO ; Timer2 Clock Input MUX bit (T2CH
27 CONFIG SDOMX = RBD ; SDO Output MUX bit (SDO function
28 CONFIG MCLEE = ON ; Master Clear Reset Pin Enable (M
29
30 ; CONFIGI
31 CONFIG STUREN = ON ; Stack Full/Underflow Reset (Stack
32 CONFIG LUOP = OFF ; Single-Supply ICSP Enable bit (I
33 CONFIG ICPRT = OFF ; Dedicated In-Circuit Debug/Progr
34 CONFIG XINST = OFF ; Extended Instruction Set Enable
35
36 ; CONFIGB
37 CONFIG LUR = 0 ; Low-Use Register (LUR) configuration
38 CONFIG LUT = 0 ; Low-Use Timer (LUT) configuration
39 CONFIG LIPRT = OFF ; Dedicated In-Circuit Debug/Progr
40

```

35

En el proceso de la compilación de un proyecto en MPLAB X:

- Para compilar se presiona el botón martillo



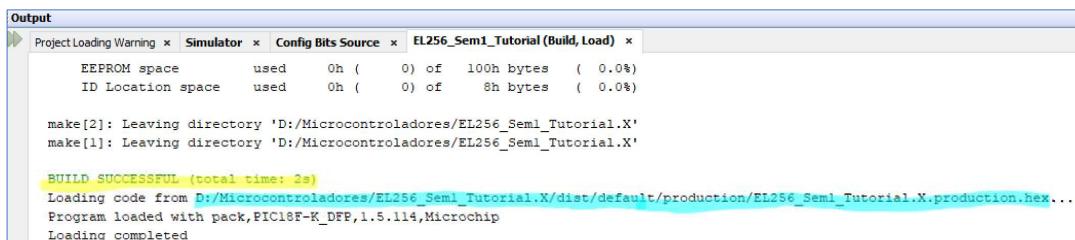
- Si hubo algun problema en la compilación, revisar los mensajes en la Ventana de Output, en dicha ventana se reportarán las líneas en donde hubo error, muy posiblemente de sintaxis



36

En el proceso de la compilación de un proyecto en MPLAB X:

- Si la compilación fué satisfactoria, la ruta del archivo compilado (*.hex) estará después del mensaje Build Successful. Dicho archivo es el necesario para que lo uses en el Proteus para la simulación y también para el momento en que vayas a grabar el microcontrolador



The screenshot shows the MPLAB X Output window with the tab 'EL256_Semi_Tutorial (Build, Load)' selected. The window displays the following text:

```

Output
Project Loading Warning x Simulator x Config Bits Source x EL256_Semi_Tutorial (Build, Load) x
EEPROM space used 0h ( 0 ) of 100h bytes ( 0.0%)
ID Location space used 0h ( 0 ) of 8h bytes ( 0.0%)

make[2]: Leaving directory 'D:/Microcontroladores/EL256_Semi_Tutorial.X'
make[1]: Leaving directory 'D:/Microcontroladores/EL256_Semi_Tutorial.X'

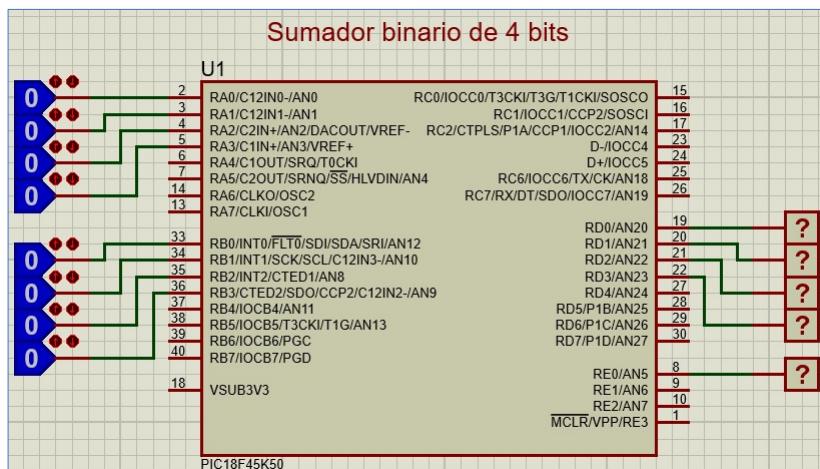
BUILD SUCCESSFUL (total time: 2s)
Loading code from D:/Microcontroladores/EL256_Semi_Tutorial.X/dist/default/production/EL256_Semi_Tutorial.X.production.hex...
Program loaded with pack, PIC18F-K_DFP, 1.5.114, Microchip
Loading completed

```

- El haber compilado satisfactoriamente no significa que al momento de las pruebas todo va a ir bien, si es que no funciona puede significar lo siguiente:
 - Error en la parte algorítmica
 - Error en la parte de la implementación del circuito (hardware)

37

Ejemplo: Sumador binario de 4 bits



38

Ejemplo: Sumador binario de 4 bits

Pasos:

1. Declarar los puertos de entrada/salida y digitales
2. Leer el contenido de A y mandarlo a Wreg
3. Sumar contenido de B con Wreg, resultado alojarlo en Wreg
4. Mover el contenido de Wreg hacia puerto de salida
5. ¿Y el acarrero? (pendiente)

39

Observaciones del datasheet

- Tener en cuenta lo siguiente:

Note: On a Power-on Reset, RA5 and RA<3:0> are configured as analog inputs and read as '0'. RA4 is configured as a digital input.

Note: On a Power-on Reset, RB<5:0> are configured as analog inputs by default and read as '0'; RB<7:6> are configured as digital inputs.

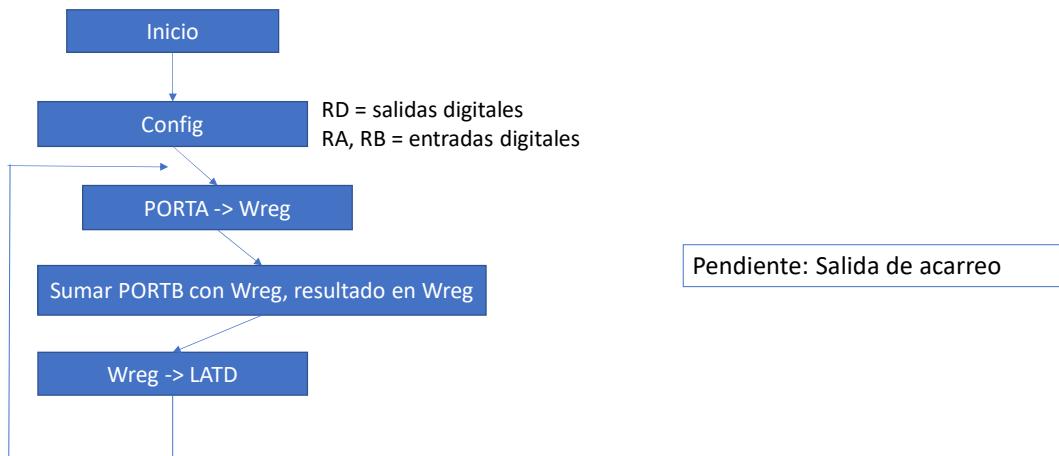
When the PBADEN Configuration bit is set to '0', RB<5:0> will alternatively be configured as digital inputs on POR.

Note: On a Power-on Reset, these pins are configured as analog inputs.

Note: On a Power-on Reset, RE<2:0> are configured as analog inputs.

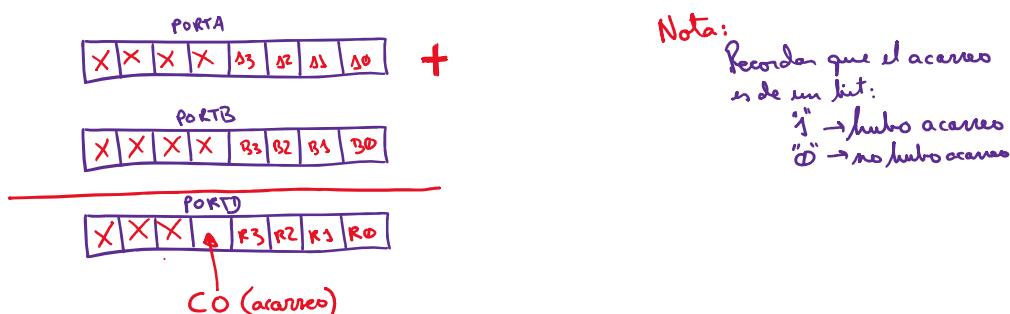
40

Diagrama de flujo



41

Análisis para la salida de acarreo del sumador



- El microcontrolador ejecuta una operación de suma en 8 bits
- Se tomará el quinto bit del resultado para obtener la salida para el acarreo

42

Código en XC8 PIC Assembler

```

1  PROCESSOR 18F45K50
2  #include "cabecera.inc"
3
4  temporal1 EQU 000H
5  temporal2 EQU 001H
6
7  PSECT principal, class=CODE, reloc=2, abs
8  principal:
9    ORG 000000H      ;Vector de RESET
10   goto configuro
11
12  ORG 000020H      ;Zona de programa de usuario
13 configuro:
14  ;Configuraciones de la aplicación
15  movlb 0FH
16  movlw 52H
17  movwf OSCCON      ;HFINTOSC a 4MHz / Internal oscillator block
18  clrf OSCCON2
19  movlw 0FH
20  movwf TRISD      ;RD3:RD0 como salidas
21  movwf ANSELA      ;RA2:RA0 como puertos digitales de entrada
22  movwf ANSELB      ;RB3:RB0 como puertos digitales de entrada
23  movwf ANSELD      ;RD3:RD0 como puertos digitales de salida
24  bcf TRISE,0       ;RE0 como salida
25  bcf ANSELE, 0     ;RE0 como digital
26
27  inicio:
28    movff PORTA, temporal1      ;Mover PORTA hacia temporal1
29    movff PORTB, temporal2      ;Mover PORTB hacia temporal2
30    movlw 0FH                  ;Valor de enmascaramiento
31    andwf temporal1, f         ;Enmascaramiento a temporal1
32    andwf temporal2, f         ;Enmascaramiento a temporal2
33    movf temporal1, W          ;Mueve contenido de temporal1 hacia Wreg
34    addwf temporal2, W          ;Sumo contenido de temporal2 con Wreg y resultado en Wreg
35    movwf LATD                ;Muevo contenido de Wreg (resultado de suma) a RD
36    btfss LATD, 4              ;Pregunto si bit4 de LATD es uno
37    goto noes
38    bsf LATE, 0                ;Si hubo acarreo
39    bra inicio                 ;Repito proceso
40
41  noes:
42    bcf LATE, 0                ;No hubo acarreo
43    bra inicio
44
45  end principal

```

43

Simulación



44

Fin de la sesión!