Sistemas Digitales: Introducción a los microcontroladores Parte 1

Ingeniería Electrónica

UPC 2018

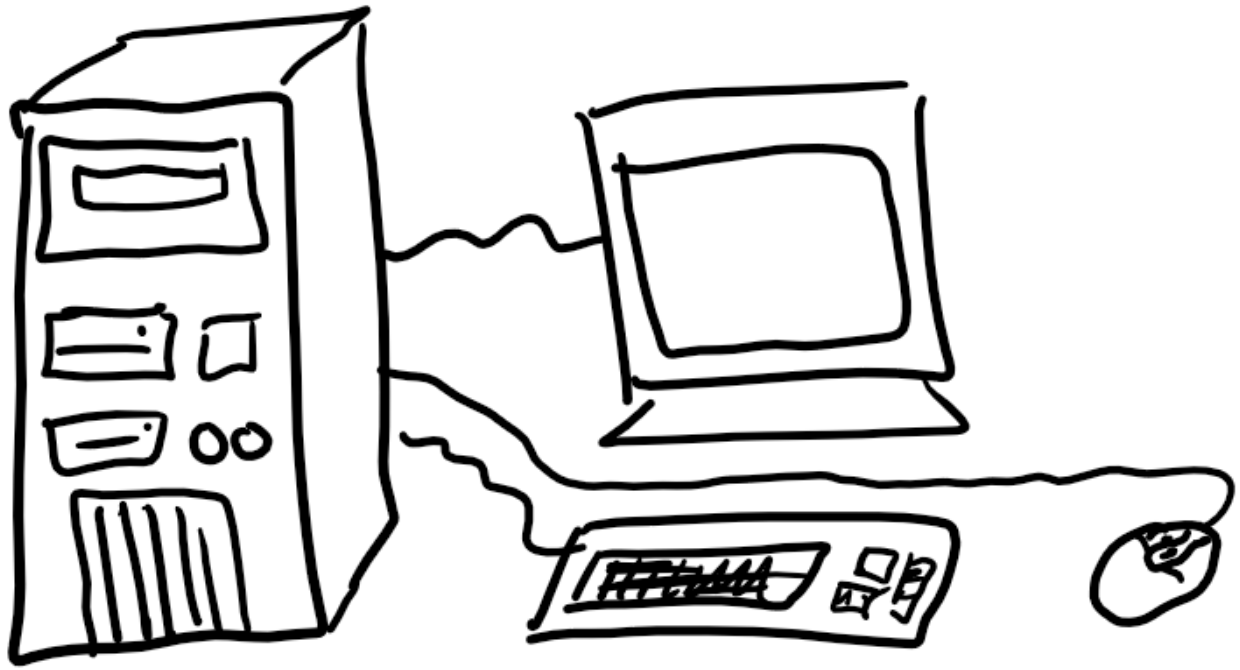
Por Kalun Lau

Microcontroladores

↳ Dispositivo programable "todo en uno" para desarrollar aplicaciones en electrónica.

↳ Es "la solución en un chip" → muy específicas

Ej:



Tiene un montón de periféricos: - Mouse - Soundcard
- Keyboard - Videocard
- HDD - DVD
- Ethernet - USB

Fabricantes: (Microchip, Renesas, NXP, ST, etc)

Microchip (http://www.microchip.com)

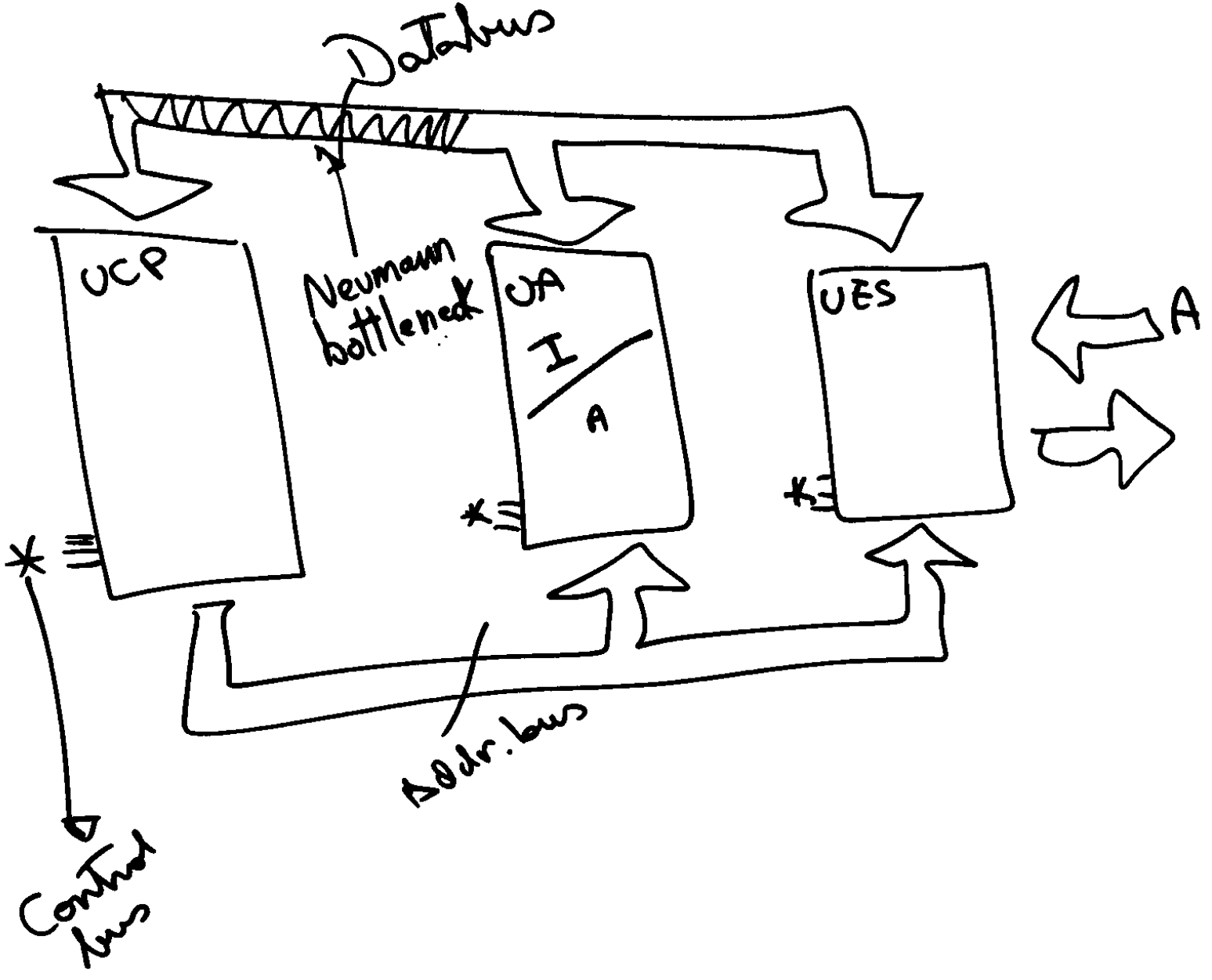
Nº1 en ventas microcontroladores de 8 bits

- PIC
- AVR (Atmel)
  - Arduino → Mega 328P

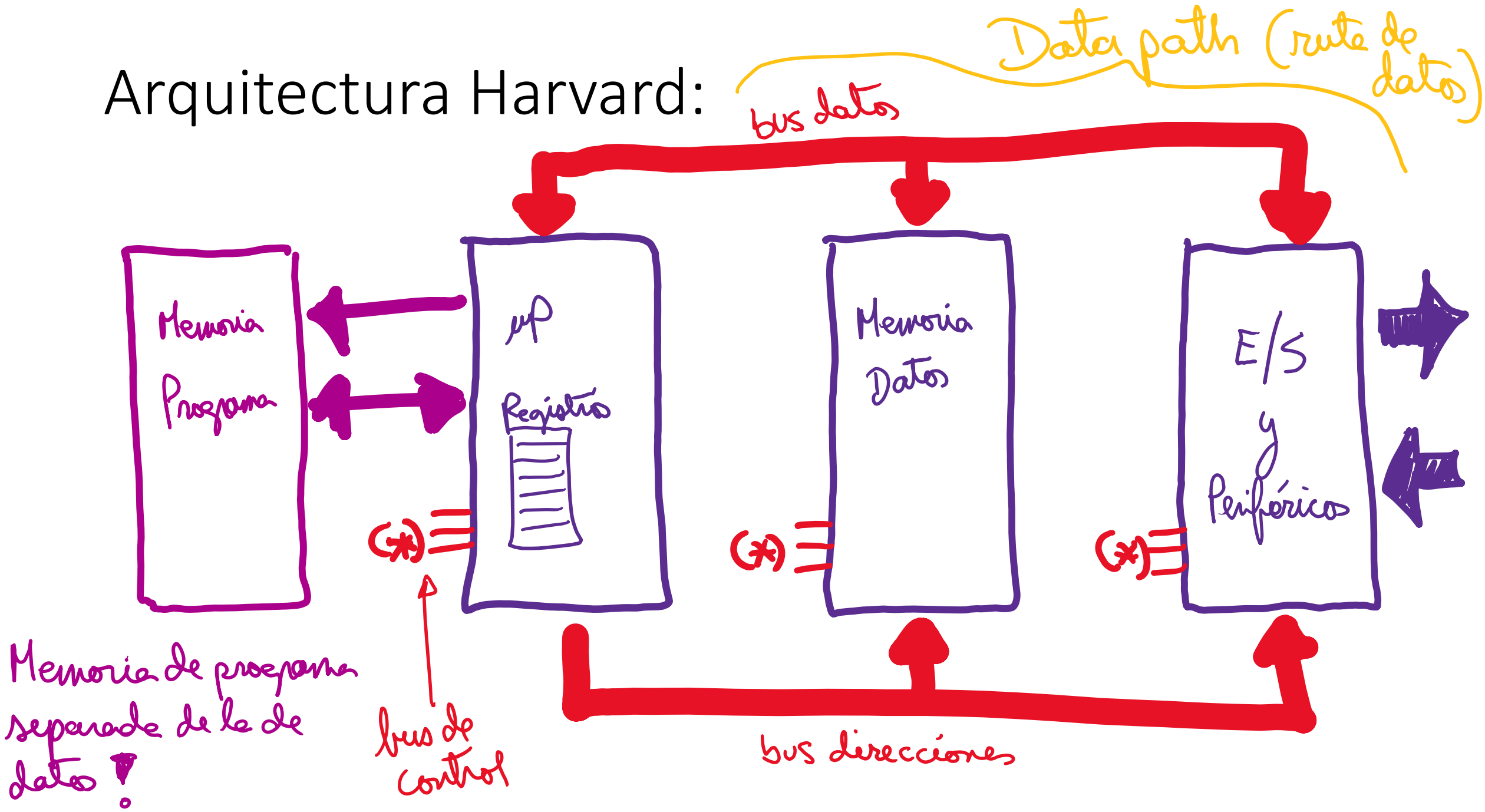
PIC16F84A

- se desarrollaron muchas aplicaciones (AN)
- bajo costo
- muchos desarrolladores lo emplearon para aplicaciones de control de procesos industriales y comerciales

Arquitectura Von Neumann

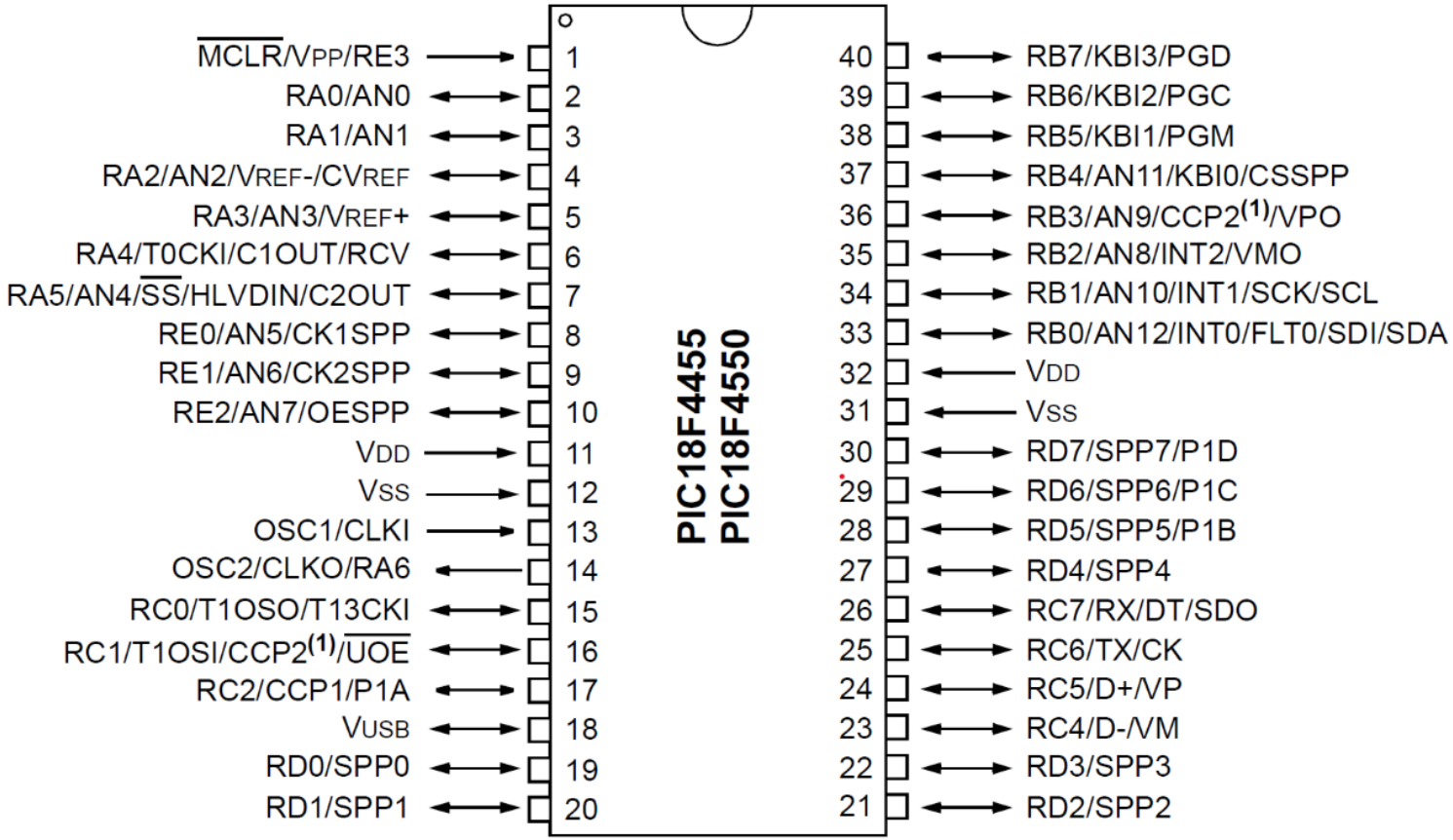


Arquitectura Harvard:



Nosotros usaremos el PIC18F4550!

40-Pin PDIP → cuarenta pines!



→ Tecnología CMOS

VDD = 5V
VSS = 0V

VDD mínimo:
3.0V (F)
2.0V (LF)

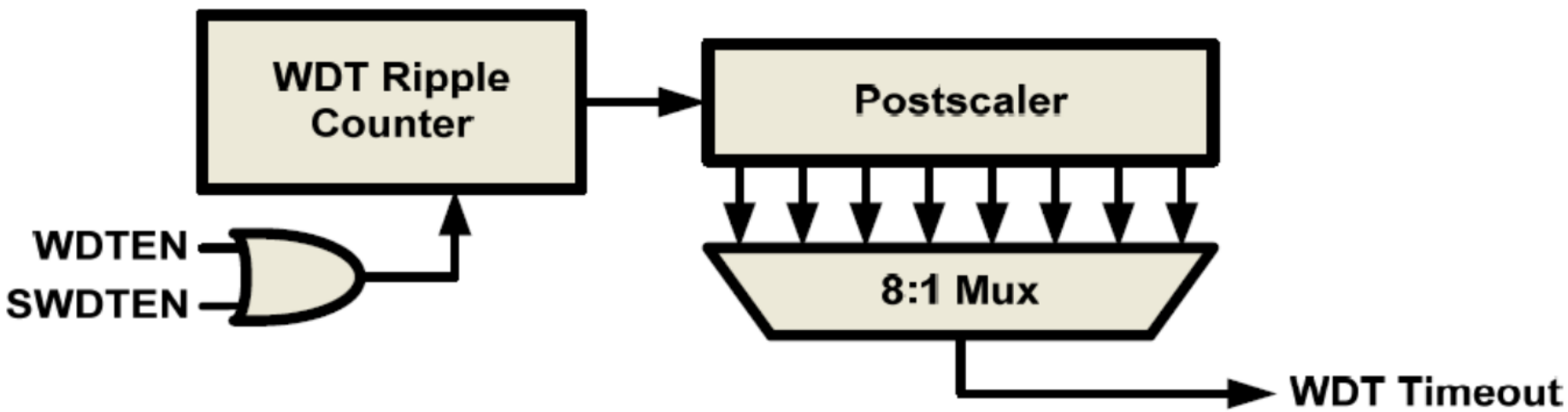
VDD máximo:
5.5V (F y LF)

Características Especiales PIC18F

- Amplio Voltaje de operación: 2.0V a 5.5V
- Memoria de Programa Flash Mejorada con 100,000 ciclos de borrado/escritura
- Memoria de Datos EEPROM con 1,000,000 de ciclos de borrado/escritura
- Retención de Datos en Memoria EEPROM Flash/Data : 100 años típico

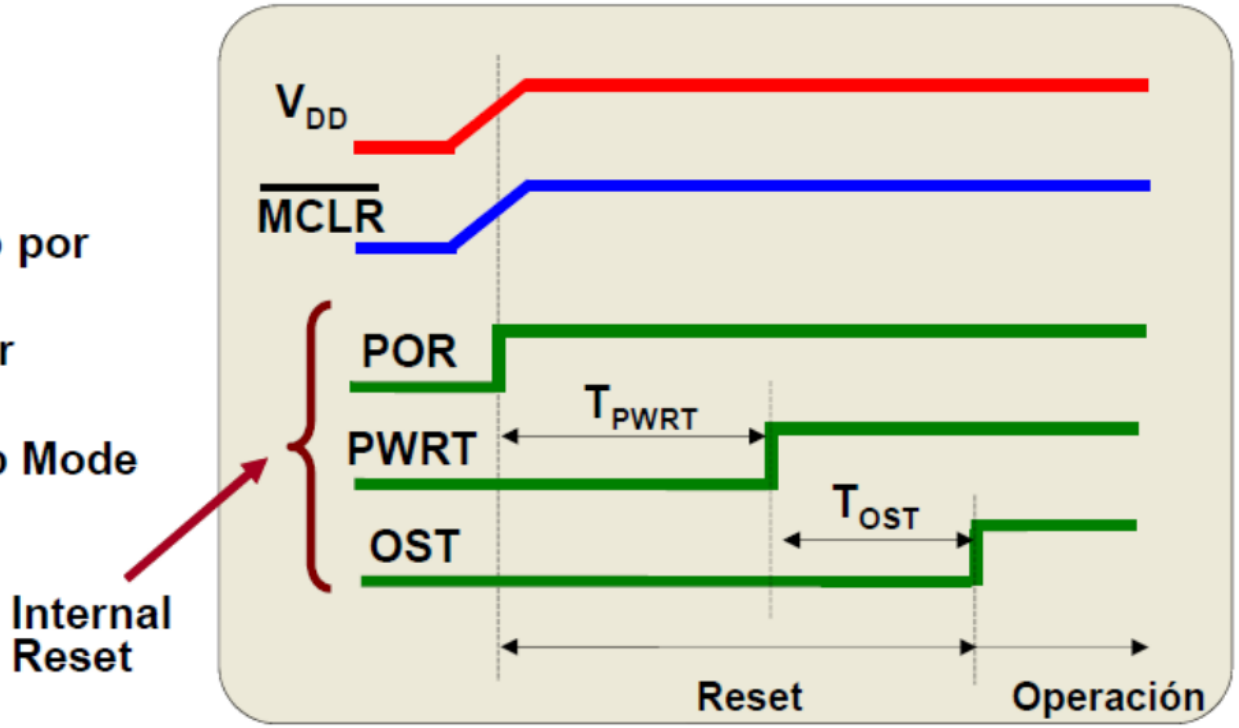
Watchdog Timer

- Ayuda al software a recuperarse de un mal funcionamiento
- Usa un oscilador libre RC en el chip
- WDT es borrado por la instrucción CLRWDT
- WDT Habilitable (WDTEN) no puede ser borrado por soft
- el overflow (desborde) del WDT reetea al chip
- Período del timeout programable : 18ms a 3.0s típico
- Opera en modo SLEEP; sobre el time out, despierta la CPU.



Generador de Reset Interno

- **POR: Power On Reset**
  - con MCLR ataado a V_{DD}, es generado un pulso de RESET cuando el flanco de subida a V_{DD} es detectado
- **PWRT: Power Up Timer**
  - 72 ms (nominal)
  - Desacoplado del BOR
- **OST: Oscillator Startup Timer**
  - Mantiene en RESET al dispositivo por 1024 cyclos de maquina (TCYs)
  - Le permite al cristal o al resonador estabilizarse
  - Bypaseado en: Two Speed Startup Mode
    - INTOSC usa un clock para el procesador no estable



BOR –Brown Out Reset

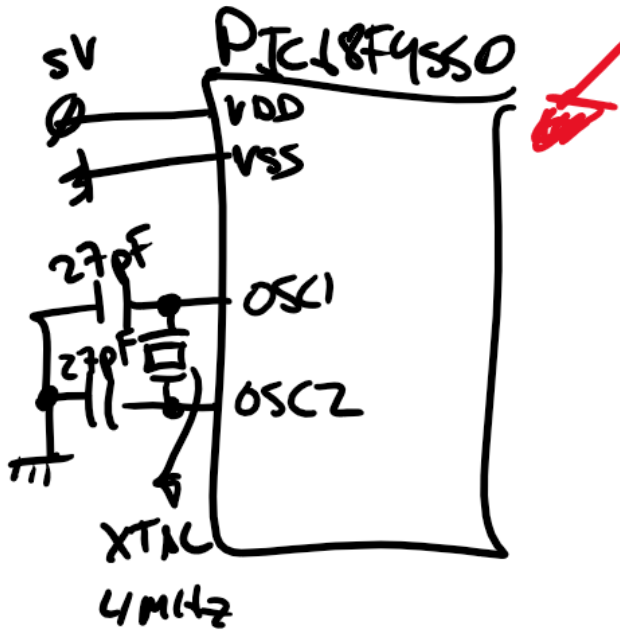
- Cuando el voltaje cae por debajo de un umbral particular, el dispositivo se pone en RESET
- Impide el funcionamiento irregular o inesperado
- Elimina la necesidad de un circuito externo BOR

PBOR – Programmable Brown Out Reset

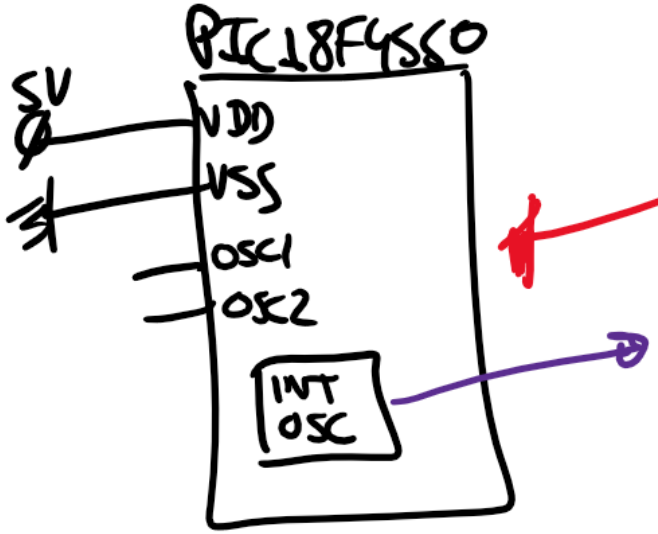
- Opción de configuración (fijado en tiempo de programa)
  - No puede ser activado/desactivado por software
- Cuatro puntos de disparo BVDD seleccionables
  - 2.5V – Mini V_{DD} para OTP MCUs PICmicro®
  - 2.7V
  - 4.2V
  - 4.5V
- Para otros umbrales use un supervisor externo (MCP1xx, MCP8xx/TCM8xx, or TC12xx)

Fuentes de reloj:

- Cristal externo:
  4MHz, 8MHz, 10MHz
  20MHz hasta 48MHz



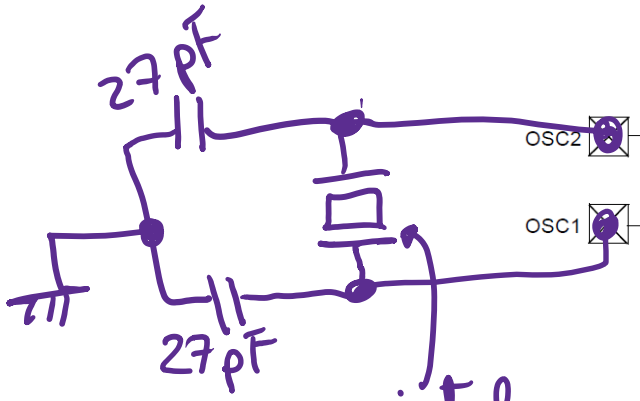
- Oscilador interno: 8MHz configurable



oscilador RC

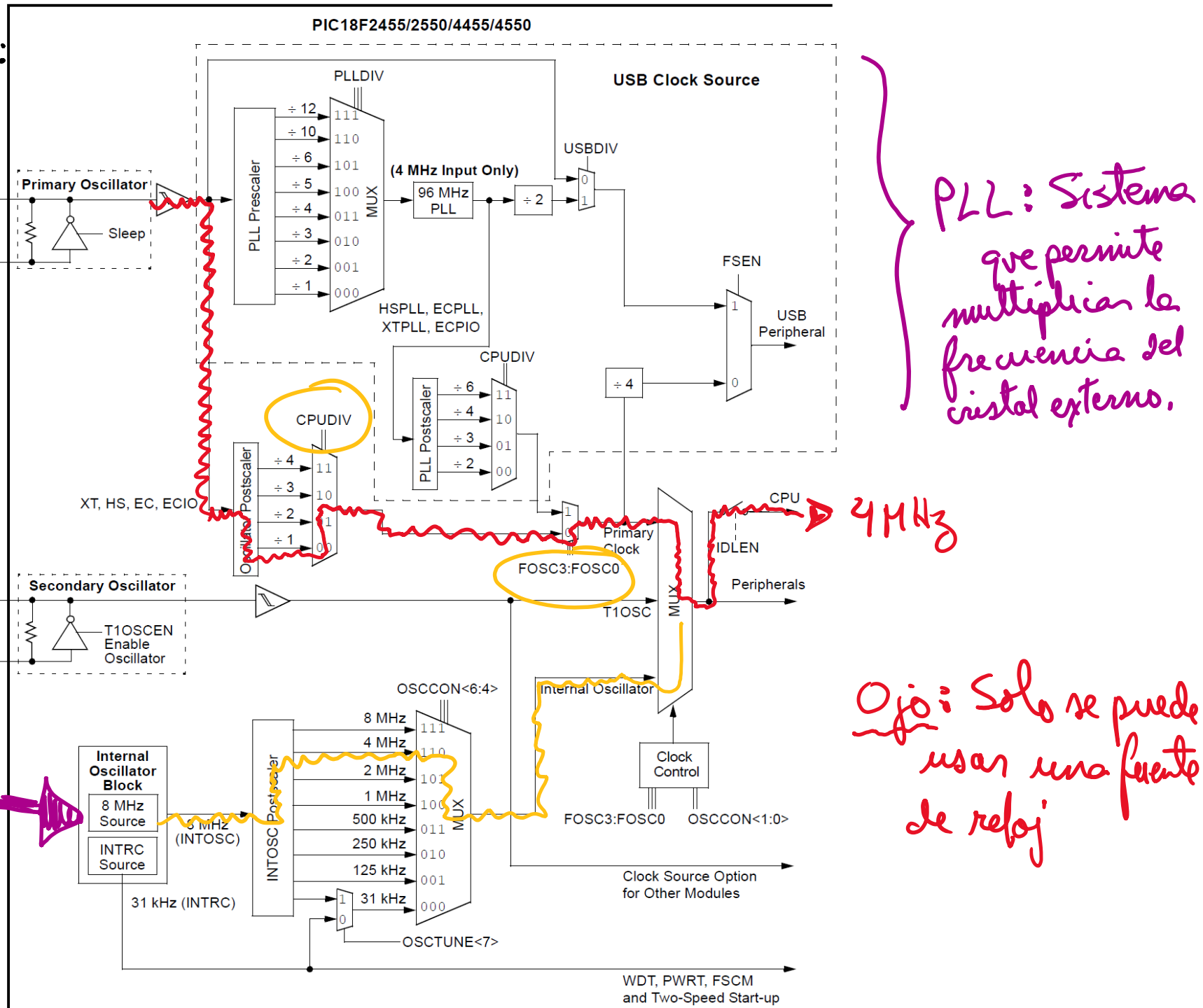
varía su precisión con la temperatura

Configuración inicial a detalle:



Cristal 4MHz

FOSC: XT_XT
(bit de configuración)



PLL: Sistema que permite multiplicar la frecuencia del cristal externo.

4MHz

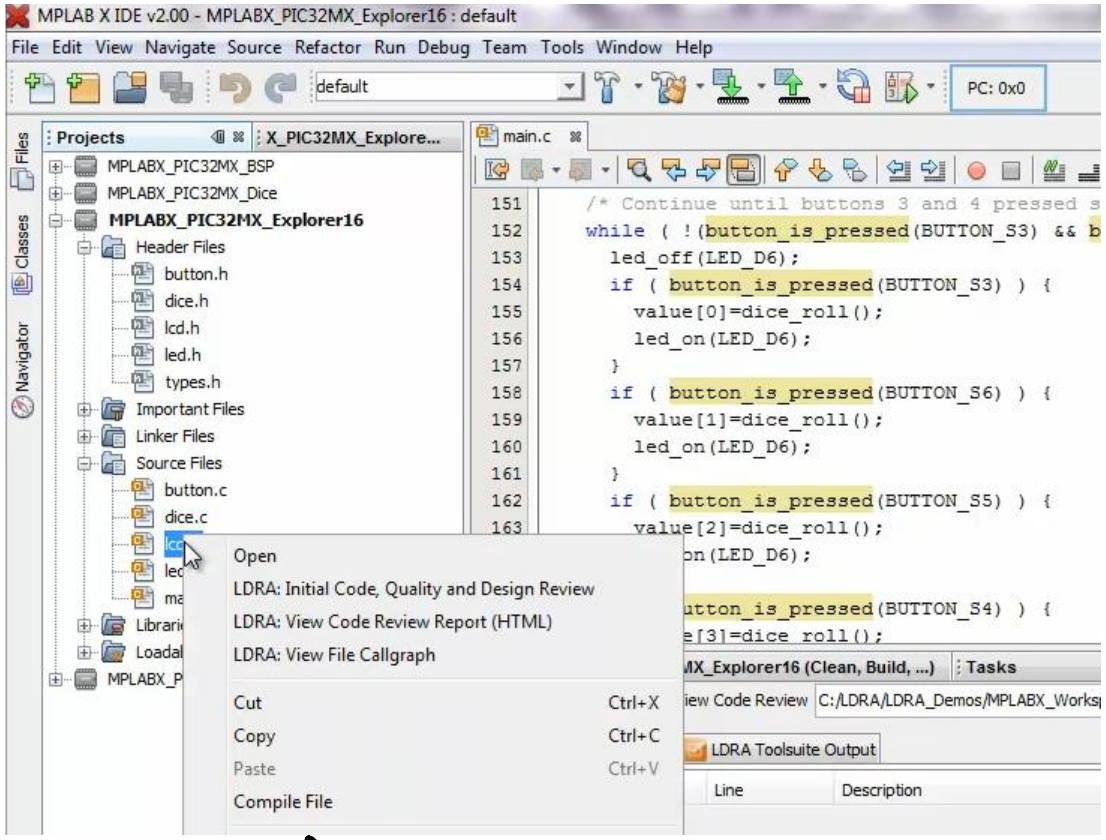
Opción para usar el oscilador interno:
FOSC: INTOSCIO_EC
(bit de configuración)

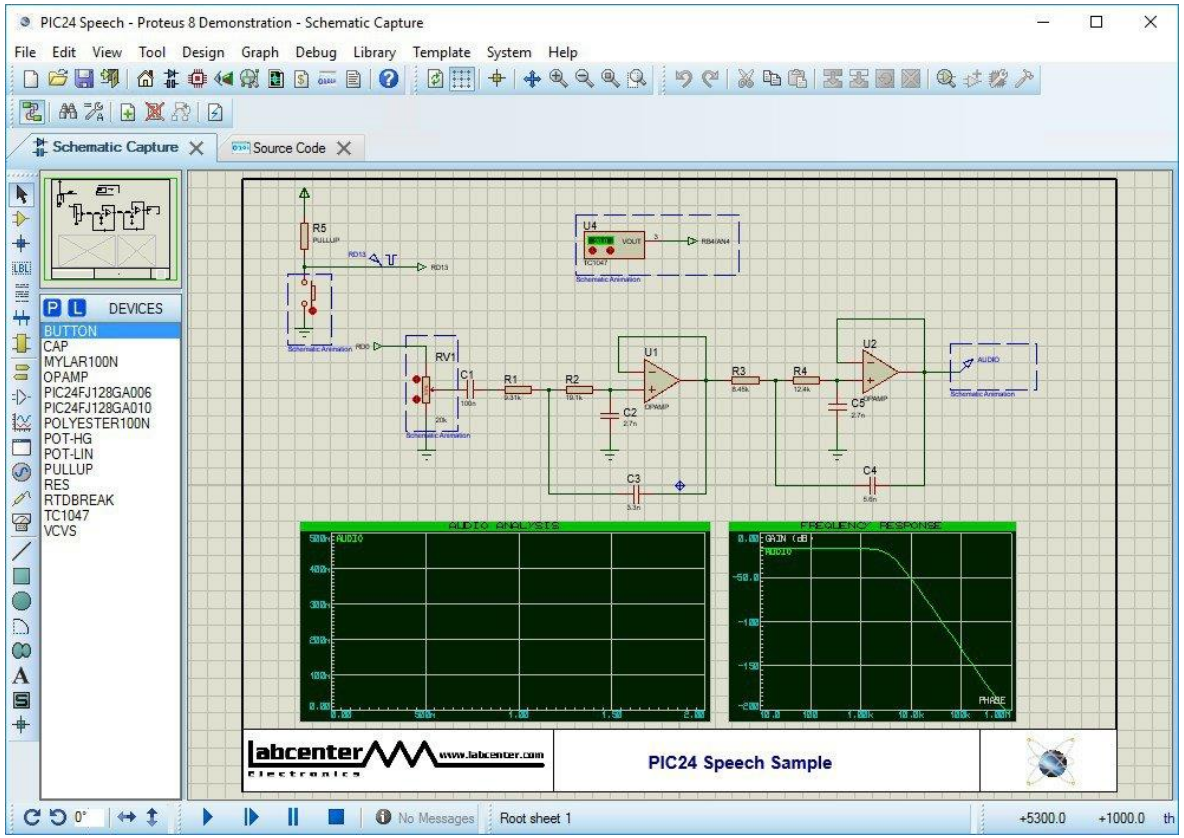
Ojo: Solo se puede usar una fuente de reloj

¿Cómo se programa?

- Lenguaje ensamblador (MPASM) ✓
- PBasic compiler (melabs.com, USD 249.99)
- C → XC8 (microchip.com)
  - → CCSC Compiler
  - → mikroC compiler (mikroe.com)

Herramientas de desarrollo para el microcontrolador PIC18F4550





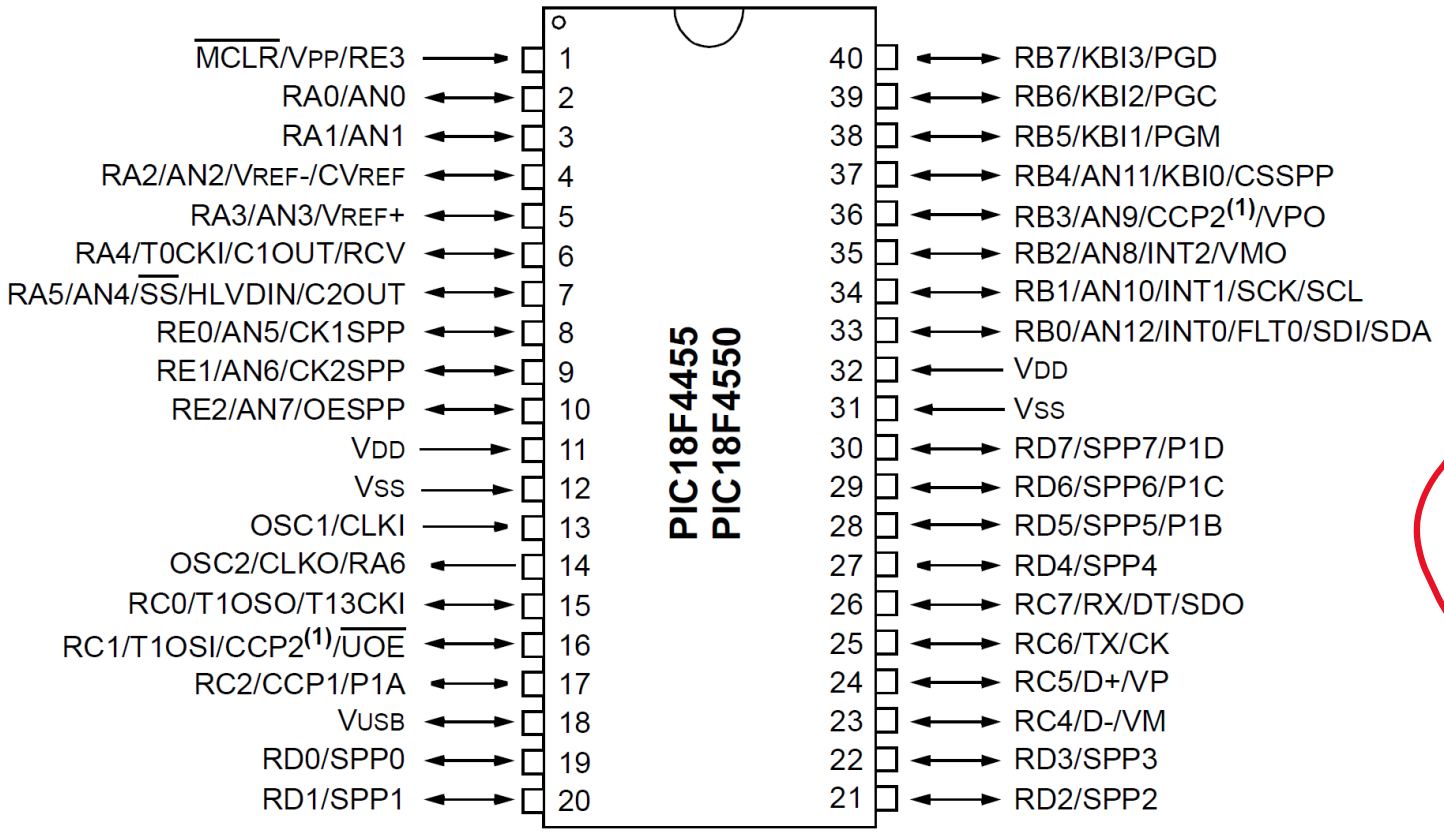
Microchip: MPLAB X (última versión: 5.05)
(www.microchip.com/mplab-x-ide/)

Se desarrollará el código

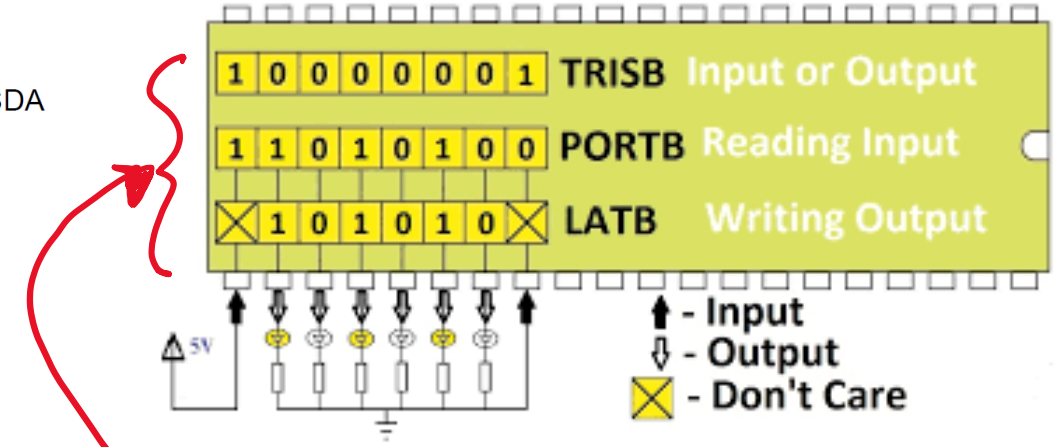
Labcenter: Proteus VSM (virtual system modelling)

Se harán las simulaciones

Manejo de puertos de E/S en el microcontrolador PIC18F4550:



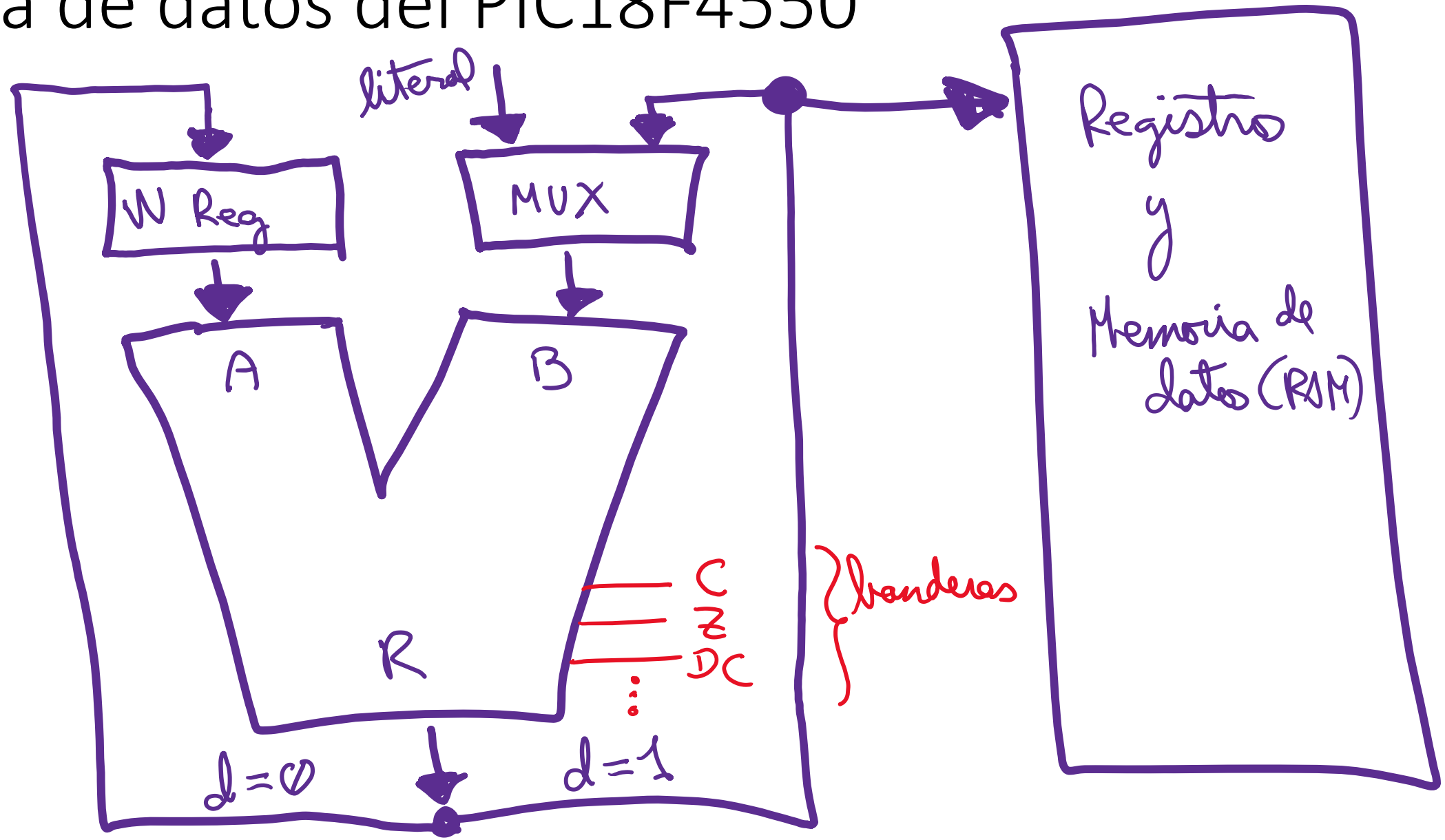
TIP: Declarar si es E o S antes de usar el puerto.



Registros para manejar los puertos de E/S.

- En modo digital E/S: RXY
  - X: letra del puerto
  - Y: número del puerto
- Algunos puertos son o solo "E" o solo "S".

Ruta de datos del PIC18F4550



MPASM: Instrucciones de movimiento de datos

MOVLW	**Move literal to W**
Syntax:	[*label*] MOVLW k
Operands:	$0 \le k \le 255$
Operation:	$k \rightarrow W$
Status Affected:	None
Encoding:	0000 1110 kkkk kkkk
Description:	The eight-bit literal 'k' is loaded into W.
Words:	1
Cycles:	1

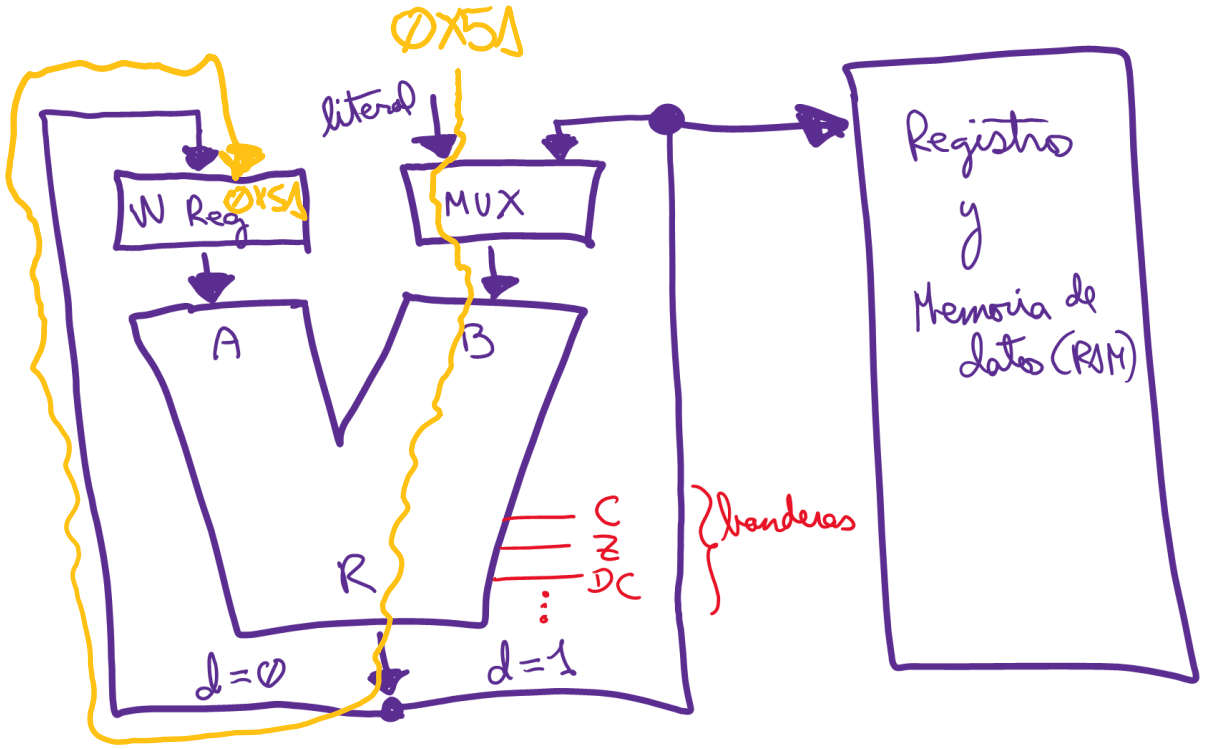
Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `MOVLW 0x5A`

After Instruction

W = 0x5A



mueve el numero ØX51 a Wreg

literal

MPASM: Instrucciones de movimiento de datos

MOVWF **Move W to f**

Syntax:	[*label*] MOVWF f [,a]
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	(W) → f
Status Affected:	None
Encoding:	0110 111a ffff ffff
Description:	Move data from W to register 'f'. Location 'f' can be anywhere in the 256 byte bank. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).
Words:	1
Cycles:	1

Q Cycle Activity:

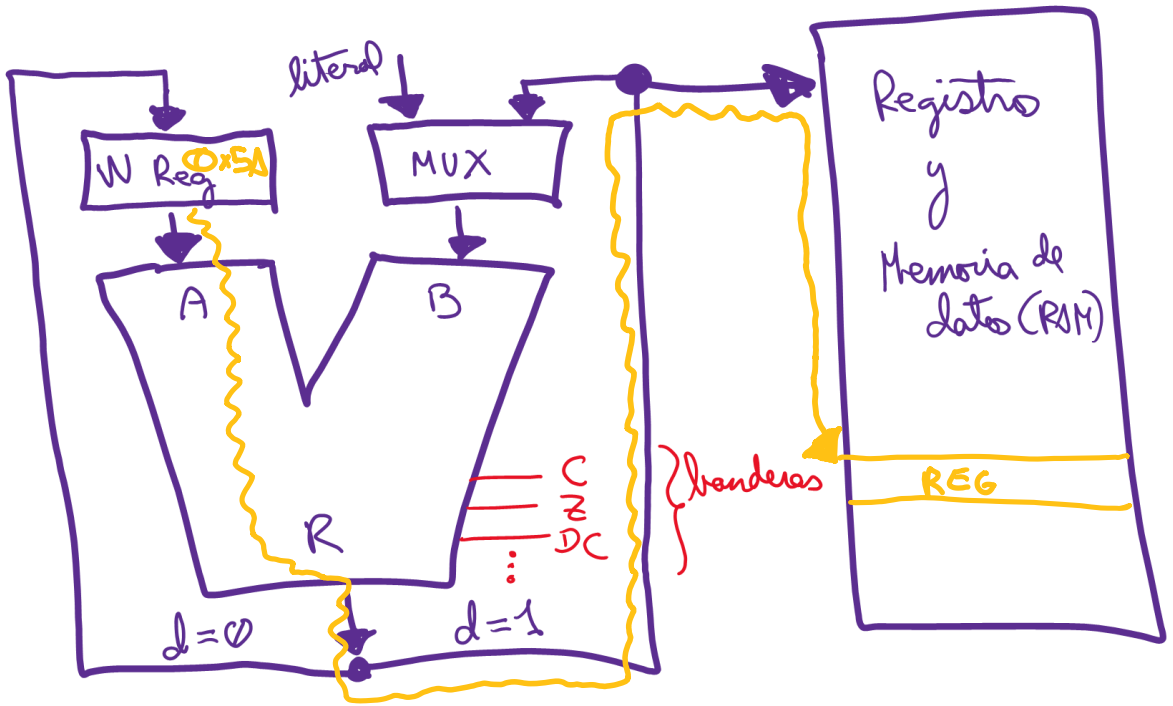
Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `MOVWF REG, 0`

Before Instruction
W = 0x4F
REG = 0xFF

After Instruction
W = 0x4F
REG = 0x4F

mueve el contenido de W hacia REG



MPASM: Instrucciones de movimiento de datos

MOVF **Move f**

Syntax:	[*label*] MOVF f [,d [,a]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	f → dest
Status Affected:	N, Z
Encoding:	0101 00da ffff ffff
Description:	The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). Location 'f' can be anywhere in the 256 byte bank. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).
Words:	1
Cycles:	1

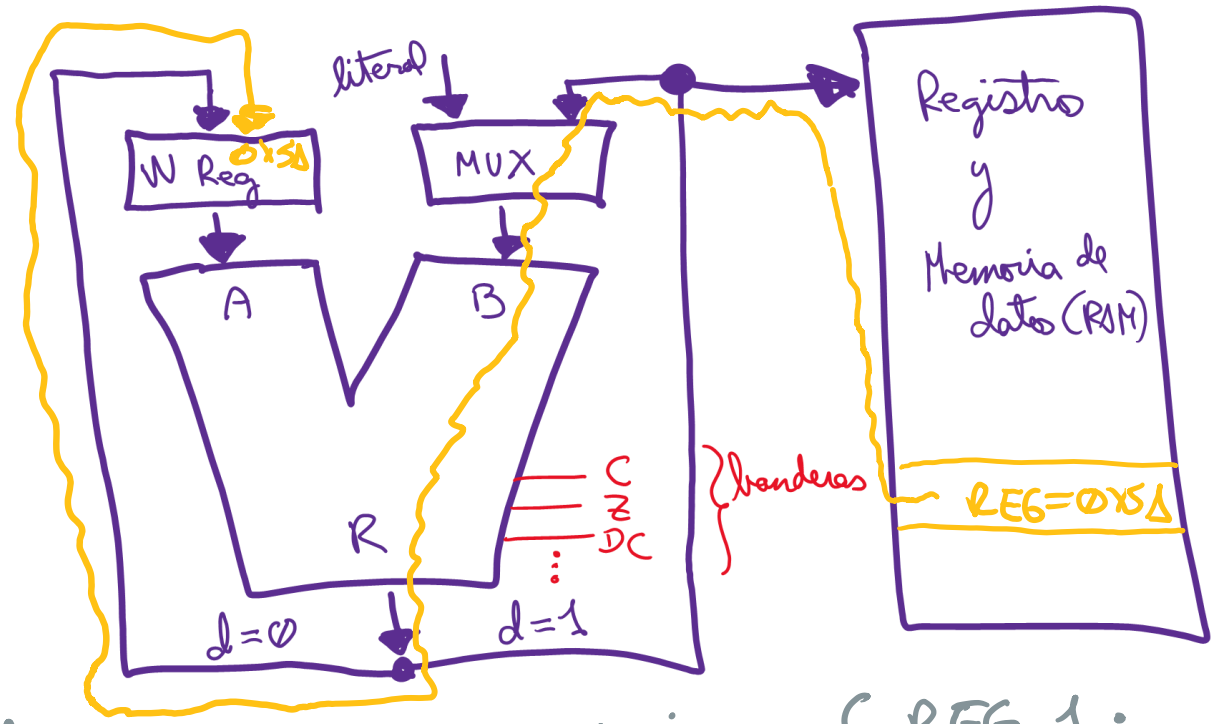
Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write W

Example: `MOVF REG, 0, 0`

Before Instruction
REG = 0x22
W = 0xFF

After Instruction
REG = 0x22
W = 0x22



mueve el contenido de REG hacia W (d=0)

si movf REG,1: moverá el contenido hacia si mismo! ¡actualizará el estado de las banderas!

MPASM: Instrucciones de movimiento de datos

MOVFF — **Move f to f**

Syntax: [*label*] MOVFF f_s,f_d

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	1100	ffff	ffff	$ffff_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of source register 'f_s' are moved to destination register 'f_d'. Location of source 'f_s' can be anywhere in the 4096 byte data space (000h to FFFh), and location of destination 'f_d' can also be anywhere from 000h to FFFh. Either source or destination can be W (a useful special situation). MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

Note: The MOVFF instruction should not be used to modify interrupt settings while any interrupt is enabled. See Section 8.0 for more information.

Words: 2

Cycles: 2 (3)

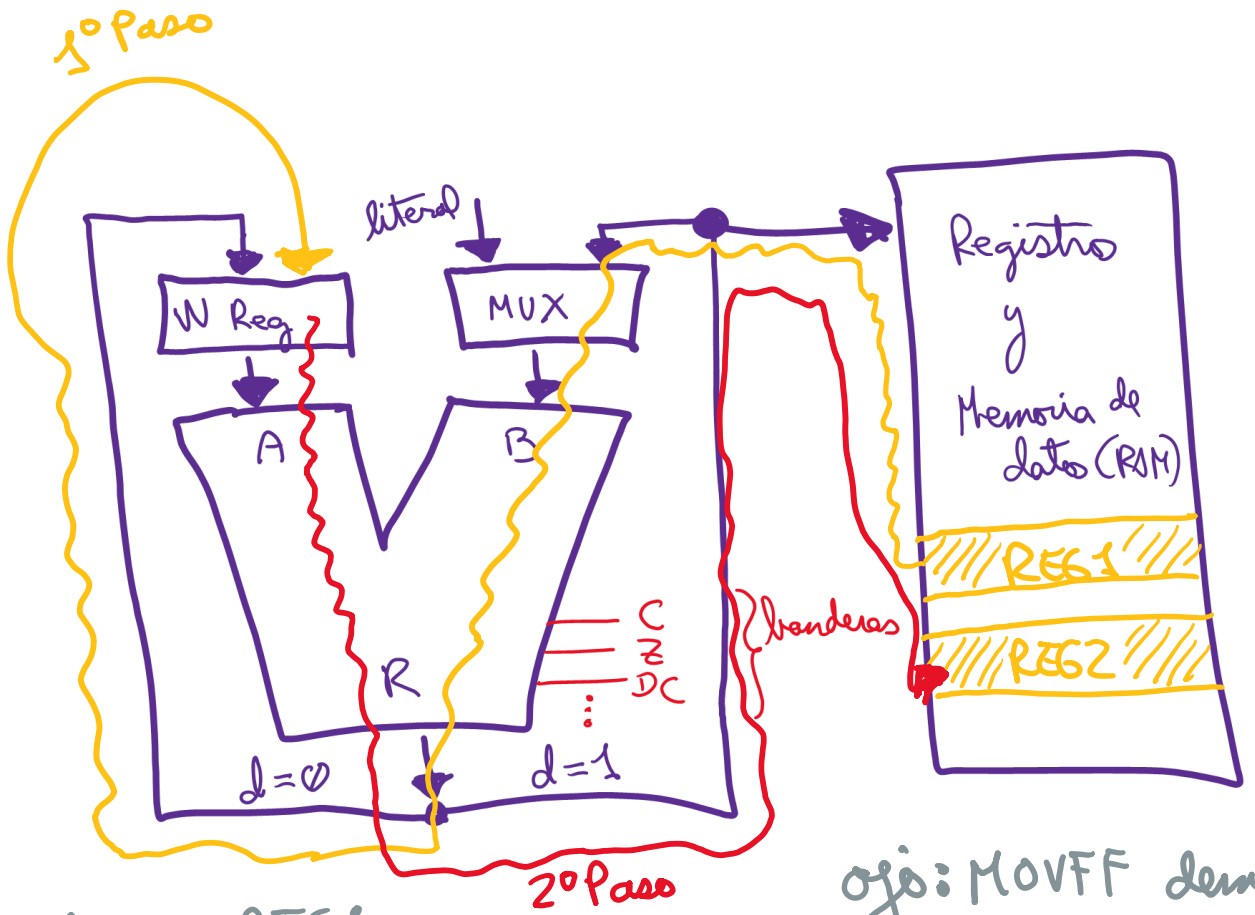
Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: `MOVFF REG1, REG2`

Before Instruction
REG1 = 0x33
REG2 = 0x11

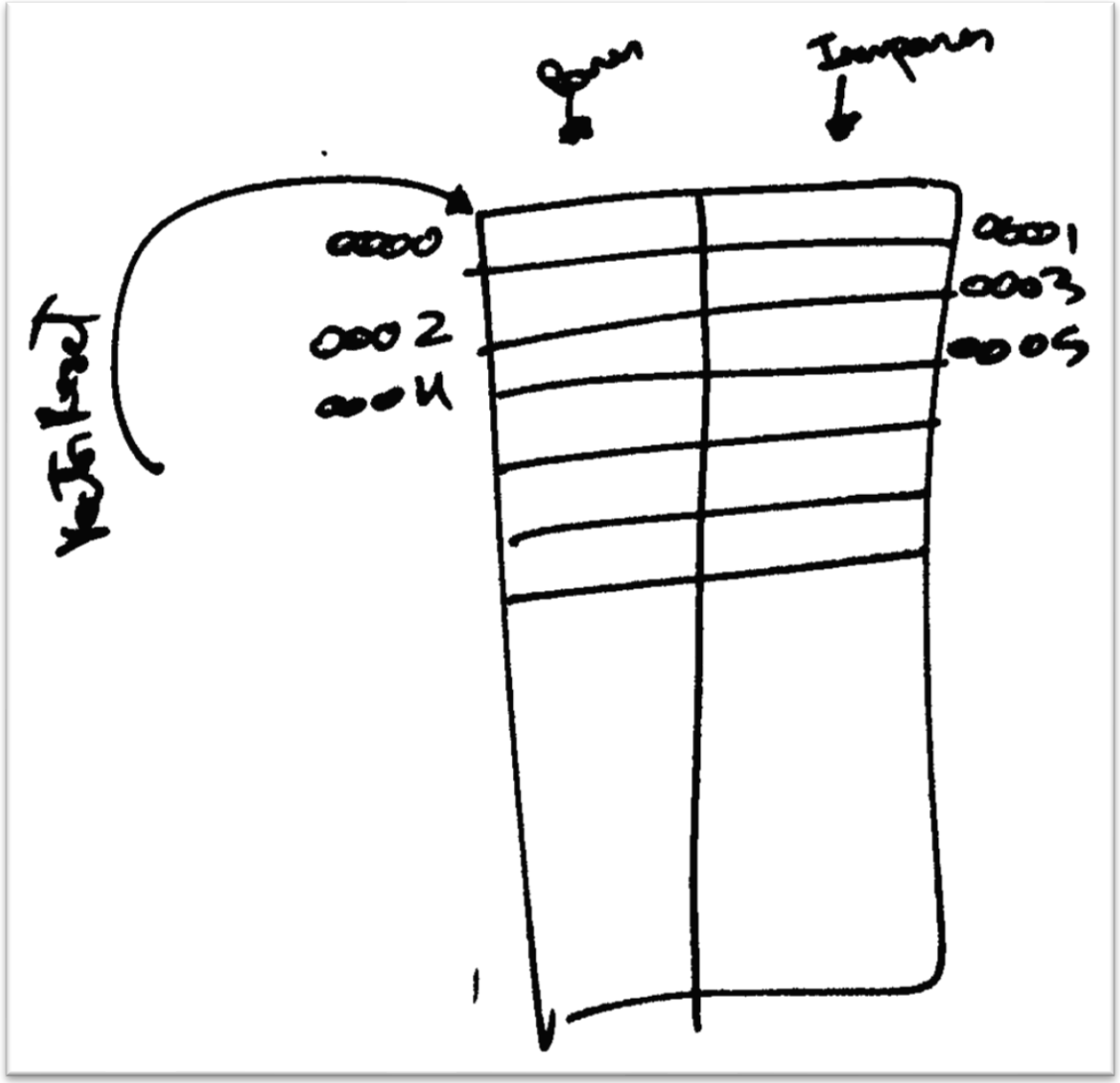
After Instruction
REG1 = 0x33,
REG2 = 0x33



Mueve el contenido de REG1 hacia REG2

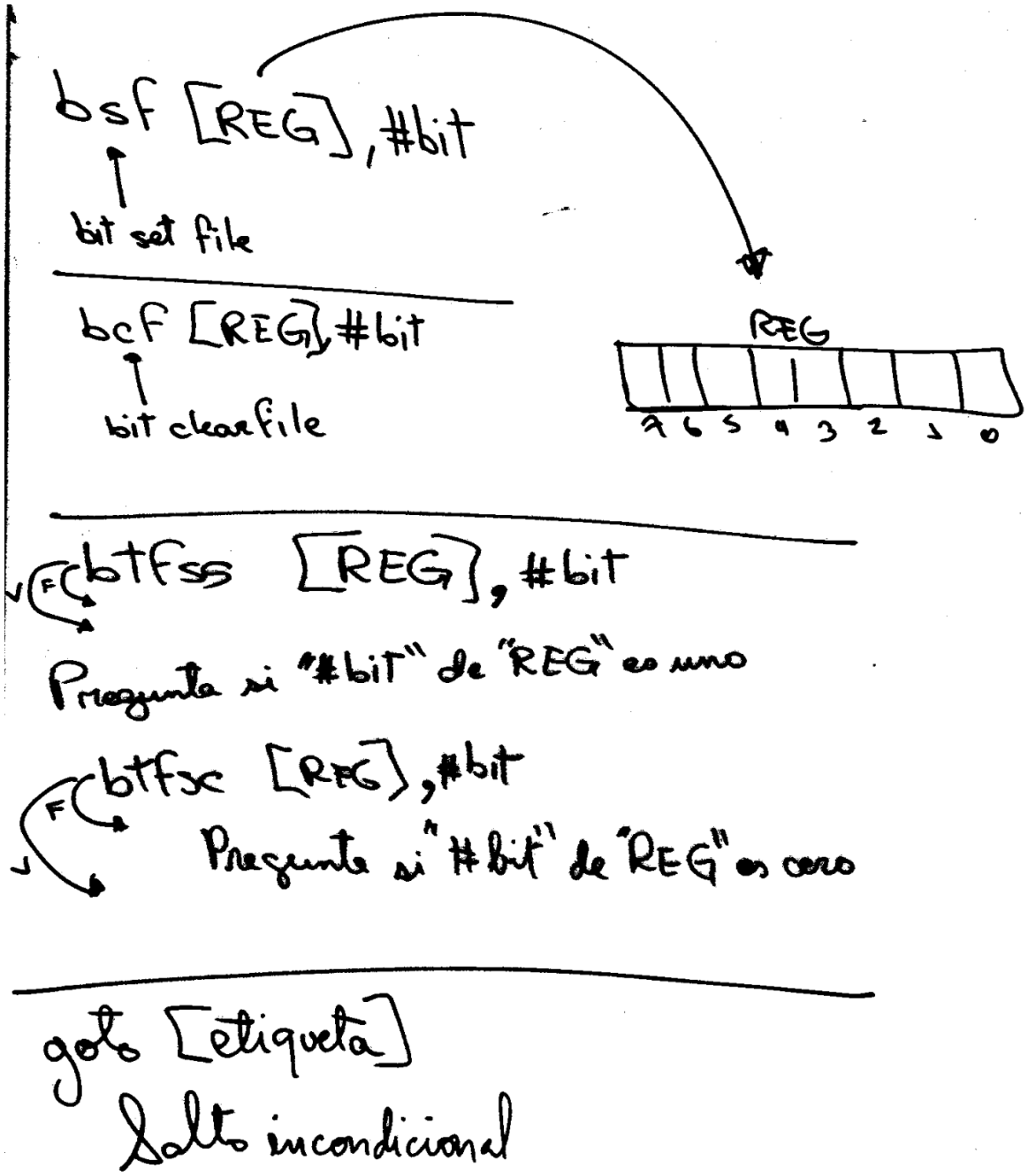
ojo: MOVFF demora dos ciclos en ejecutarse

Estructura de la memoria de programa

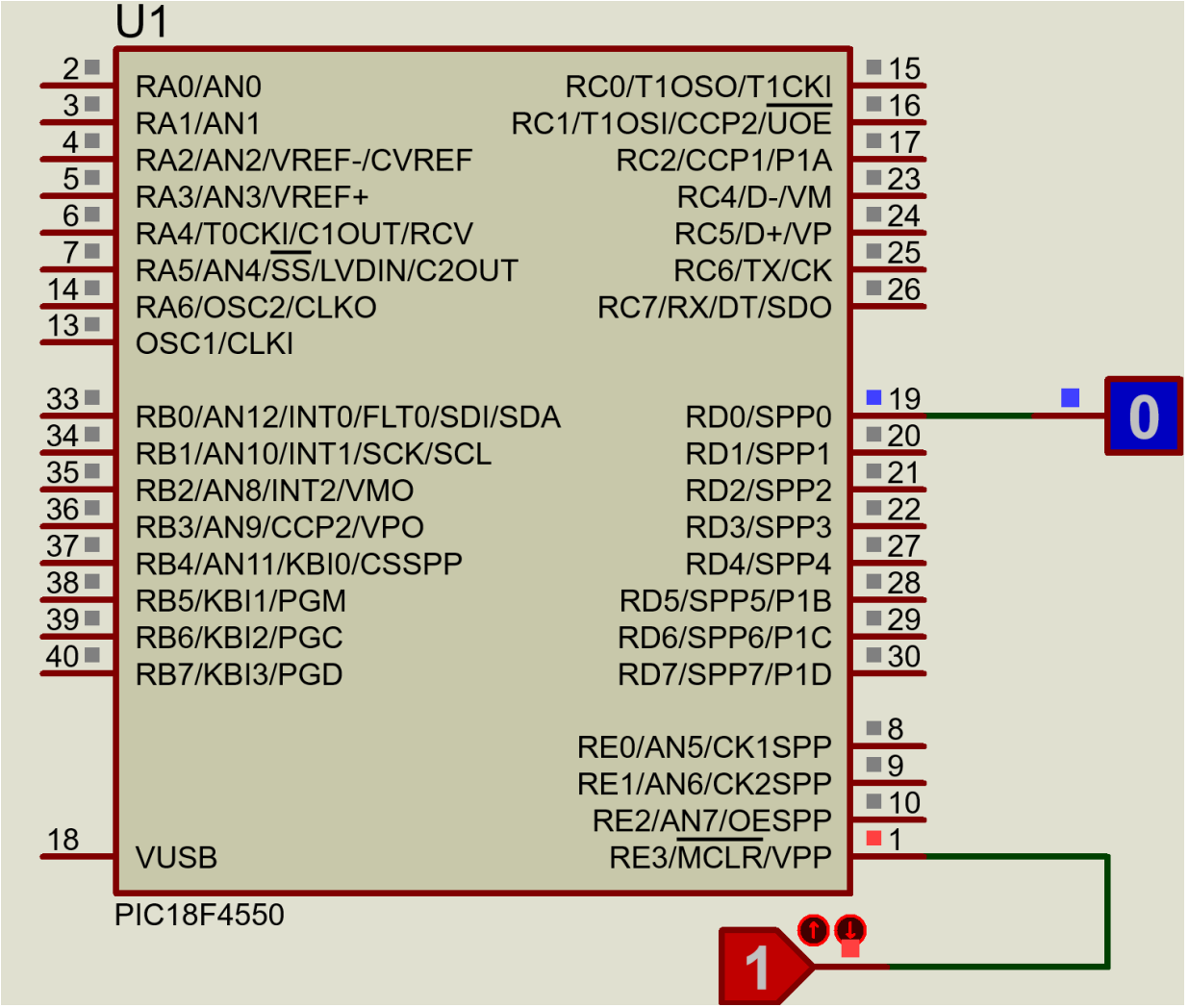


- Memoria de programa de 8 bits de ancho, 32K de capacidad.
- Las instrucciones ocupan 2 bytes (cada una)
- El contador de programa (PC) se incrementa de 2 en 2.
- Se puede usar la memoria de programa para almacenar datos (8 bits)

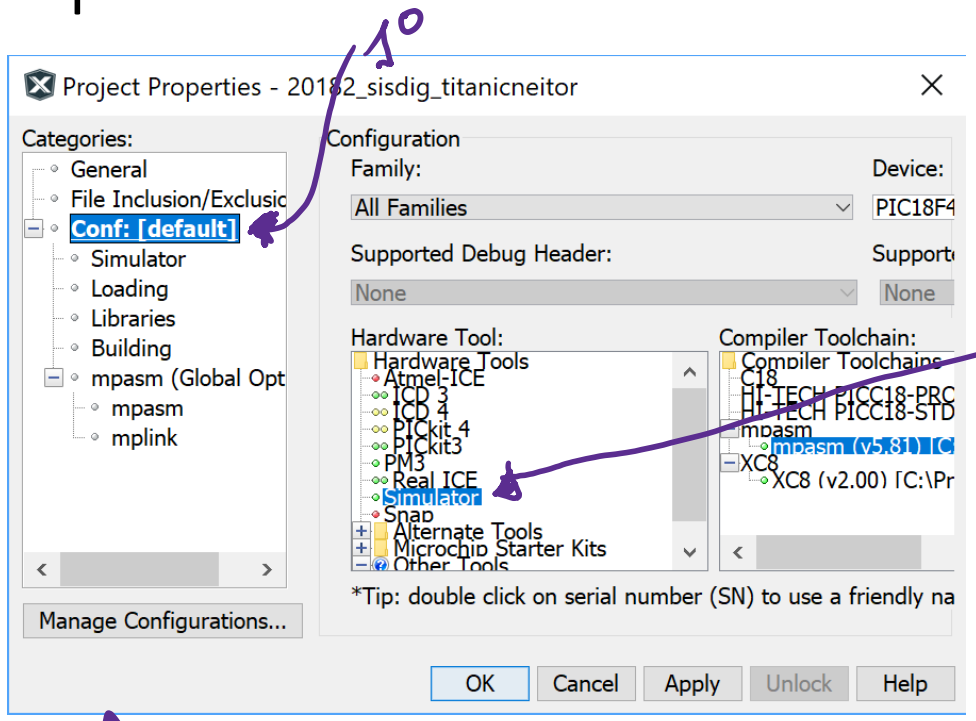
MPASM - Instrucciones
-Manipulación de bit en un registro (bsf, bcf)
-Pregunta de bit en un registro (btfss, btfsc)
-Salto incondicional (goto)



Simulación en Proteus



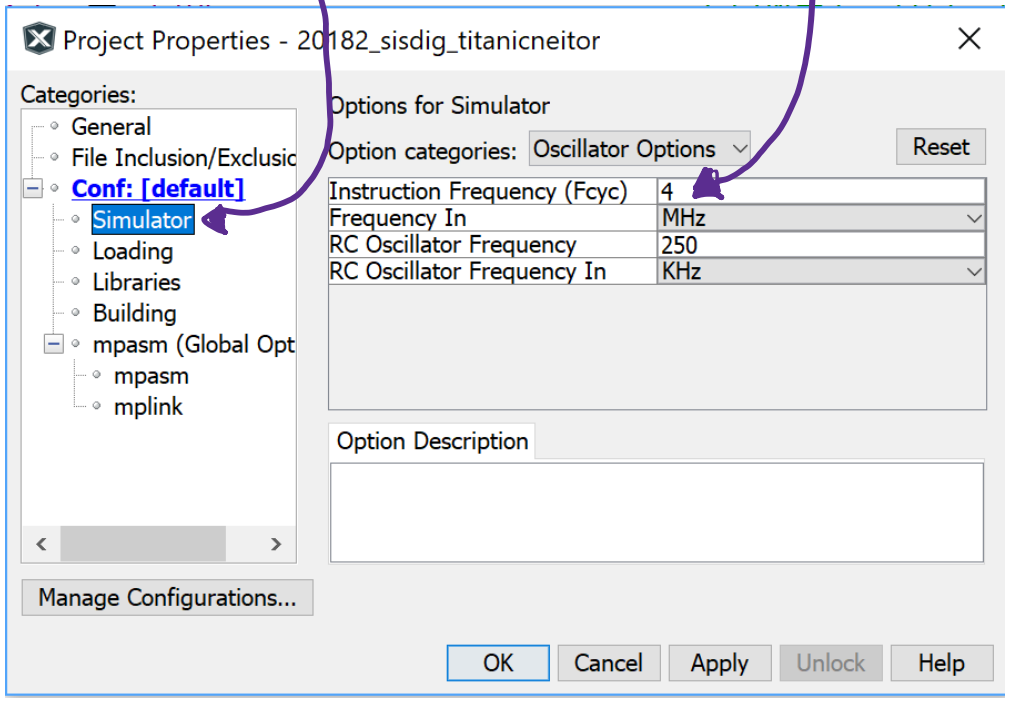
Usando el “stopwatch” del modo simulador del MPLAB X para obtener la duración de la subrutina de retardo



Seleccionas "Simulator"

Frecuencia del

2º



Ventana de propiedades del proyecto