

Microcontroladores

Semestre: 2021-1

Profesor: Kalun José Lau Gan

Semana 9: Lenguajes de alto nivel en microcontroladores

1

¿Preguntas previas?

- Sobre el TF
 - Se estará publicando mas tarde la “guía del trabajo final” en el AV
- ¿Cómo se trabajarán en los laboratorios siguientes?
 - De manera individual
- ¿Se requerirán mas materiales para los laboratorios?
 - Solo los de la lista publicada a inicios del curso
- El hecho de usar C lo hace mas fácil?
 - El proceso de diseño es el mismo tanto para lenguaje Assembler como en lenguaje C, se tiene que implementar primero el hardware, desarrollar el algoritmo y finalmente aterrizar el algoritmo en un lenguaje de programación.
 - El lenguaje XC se va a permitir el uso de librerías y funciones especializadas, además de manejar mejor las operaciones matemáticas y manipulación de datos

2

Agenda:

- Resolución del EA
- Lineamientos de TF
- Los lenguajes de alto nivel, en nuestro caso el XC8
- El MPLAB Xpress
- Desarrollo de una plantilla en el MPLAB X para el XC8
- Ejemplos iniciales en XC8

3

Panorama de los lenguajes de alto nivel para microcontroladores

Cada dispositivo microcontrolador tendrá sus propias plataformas de lenguaje de programación.

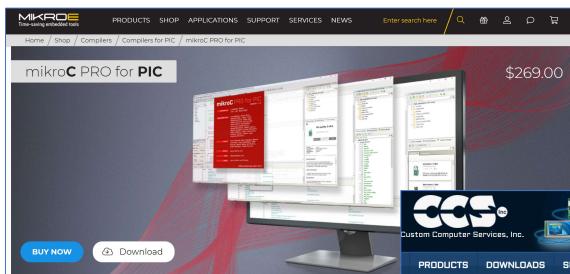
Va a depender del desarrollador del lenguaje de alto nivel para determinado microcontrolador (No todos los desarrolladores soportan todos los dispositivos de un fabricante)

Hay varias compañías que desarrollan lenguajes de alto nivel para un microcontrolador.
Problema de compatibilidad entre distintas plataformas de desarrollo aún empleando un mismo lenguaje.

- Basic
- C
- Java (muy poco utilizado y soportado)
- Python

4

Panorama de los lenguajes de alto nivel para microcontroladores



mikroC PRO for PIC

\$269.00

[BUY NOW](#) [Download](#)



PCWHD IDE Compiler for Microchip PIC10/12/16/18/24/dsPIC Devices

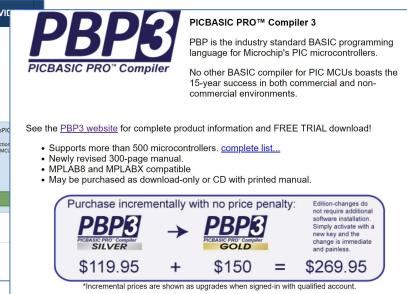
Sku: 52202-588

Devices Supported: PIC10, PIC12, PIC16, PIC18, dsPIC, PCW, PCWH, PCWHD

In stock (ships immediately)

Download version available \$600.00 [Add to Cart](#)

Starting at \$100 more, buy a complete development kit [Buy Now](#)



PBP3
PICBASIC PRO™ Compiler

PPB is the industry standard BASIC programming language for Microchip's PIC microcontrollers.

No other BASIC compiler for PIC MCUs boasts the 15-year success in both commercial and non-commercial environments.

See the [PBP3 website](#) for complete product information and FREE TRIAL download!

- Supports more than 800 microcontrollers. [complete list...](#)
- Newly revised 300-page manual.
- MPLAB® and MPLAB® X compatible
- May be purchased as download-only or CD with printed manual.

Purchase Incrementally with no price penalty:

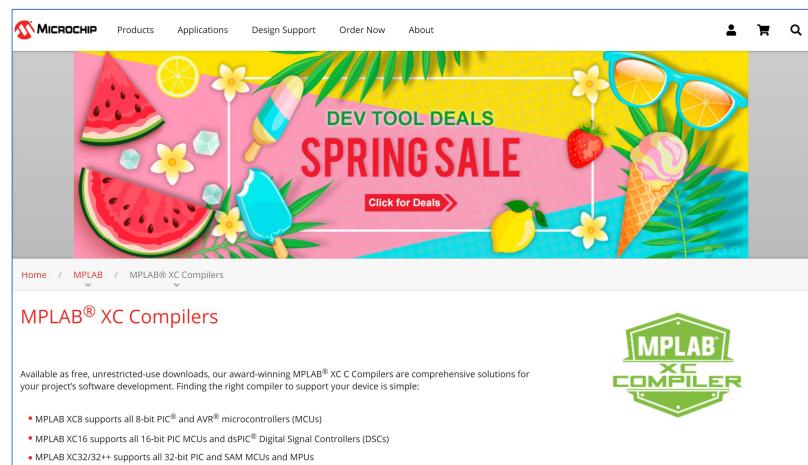
PBP3 PICBASIC PRO™ SILVER	→	PBP3 PICBASIC PRO™ GOLD		
\$119.95	+	\$150	=	\$269.95

*Incremental prices are shown as upgrades when signed-in with qualified account.

5

El compilador XC de Microchip

- Disponible versión gratis sin límite.
- La versión PRO no es gratis. Provee optimización de código compilado para que ocupe menos espacio.
- Tres versiones:
 - XC8 – PIC10, PIC12, PIC16, PIC18
 - XC16 – PIC24, dsPIC
 - XC32 – PIC32



MPLAB® XC Compilers

Available as free, unrestricted-use downloads, our award-winning MPLAB® XC C Compilers are comprehensive solutions for your project's software development. Finding the right compiler to support your device is simple:

- MPLAB XC8 supports all 8-bit PIC® and AVR® microcontrollers (MCUs)
- MPLAB XC16 supports all 16-bit PIC MCUs and dsPIC® Digital Signal Controllers (DSCs)
- MPLAB XC32/32+ supports all 32-bit PIC and SAM MCUs and MPUs

MPLAB XC COMPILER

6

El compilador XC de Microchip

- Los XC8, XC16 y XC32 vienen separados...
- Ej. El bootloader HID para el microcontrolador PIC18F4550 requiere que se compile el código en XC8 PRO para que entre en su memoria.



Part Number: SW006021-DGL - MPLAB XC8 Compiler PRO Dongle License

The MPLAB XC8 is a full-featured, highly-optimized ANSI C compiler for all 8-bit AVR® and PIC® MCUs. This compiler integrates into Microchip's MPLAB(R) X IDE, is compatible with all Microchip debuggers and emulators, and runs on Windows, Linux and Mac OS X.

The dongle license is a USB flash drive that contains a single-user encrypted license. It is a perpetual license and unlocks PRO optimizations for all versions of the MPLAB XC8 compilers, [version 1.41 and later](#), and does not include High Priority Access (HPA). The Dongle License allows a user to unlock PRO optimizations on any computer it is plugged into. If lost, the Dongle License can be replaced one time for a processing fee of \$200 and the dongle must be registered to the user.

[More Info »](#)

Standard Pricing:

Order Quantity

1+

USD per Unit

\$1,695.00

In Stock: 9

Delivery and scheduling options available in the cart [»](#)

Order now, up to 9 can ship on 20-May-2020

Lead Time For Additional Quantities [»](#)

Additional quantities can ship by 11-Aug-2020

Quantity:



7

Optimización en el compilador XC

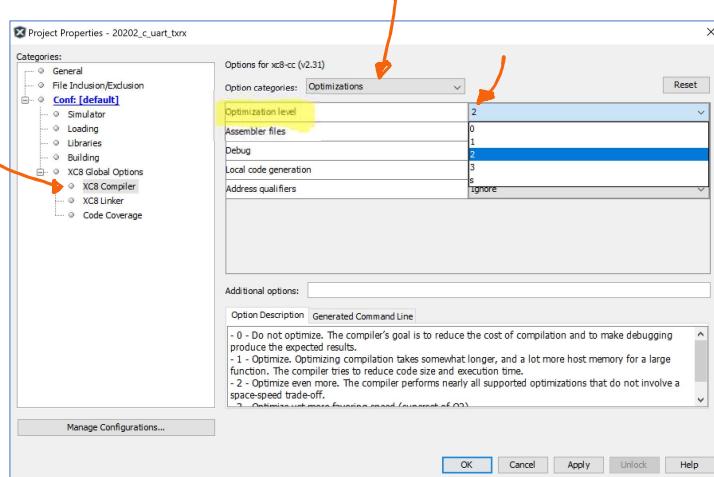
- **Optimization levels:**
 - Level 0: Fastest compilation time, minimal optimizations
 - Level 1: Optimizations with low debugging impact
 - Level 2: All optimizations that give the best balance between speed and size
 - Level 3: Program execution will be as fast as possible
 - Level s: Code size will be as small as possible
- **Where available use:**
 - Use Whole-program and Link-time setting
 - Procedural abstraction

- En la versión gratuita del XC tenemos optimización hasta el nivel 2
- Para acceder al nivel 3 o nivel s requerimos tener la licencia PRO

8

Optimización en el compilador XC

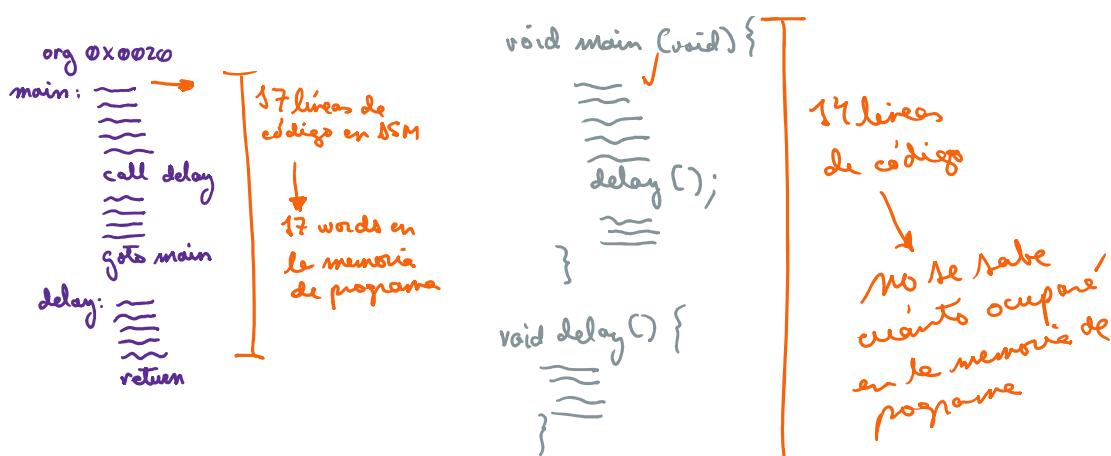
- Para acceder a las opciones de optimización ingresamos a las propiedades del proyecto:



9

Eficiencia de código en Assembler vs C

¿Por qué Assembler y por qué C?



10

El MPLAB Xpress

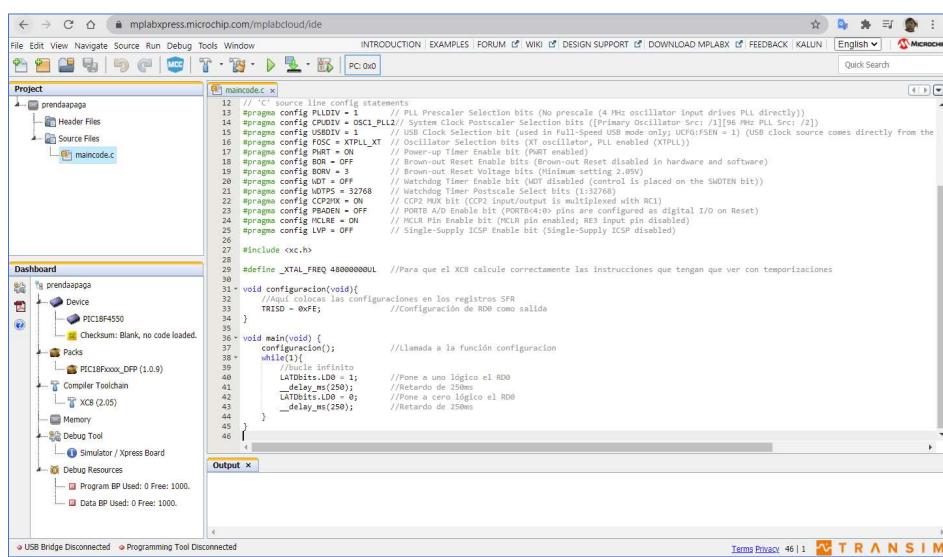
- Link:
[https://www.microchip.com
/en-us/development-tools-tools-and-software/mplab-xpress](https://www.microchip.com/en-us/development-tools-tools-and-software/mplab-xpress)
- Entorno de trabajo online
- Permite realizar proyectos en XC sin tener que instalar el MPLAB X en el computador.



11

El MPLAB Xpress

- Los proyectos se almacenan en la nube de Microchip
- Soporte solo para algunos modelos de microcontrolador PIC



12

MPLAB X: Creación de un proyecto en XC8

- Link del manual de XC8:
 - <http://ww1.microchip.com/downloads/en/devicedoc/50002053g.pdf>
- Link de descarga del XC8 v2.32:
 - <https://www.microchip.com/mplabxc8windows>

13

Plantilla de código en XC8:

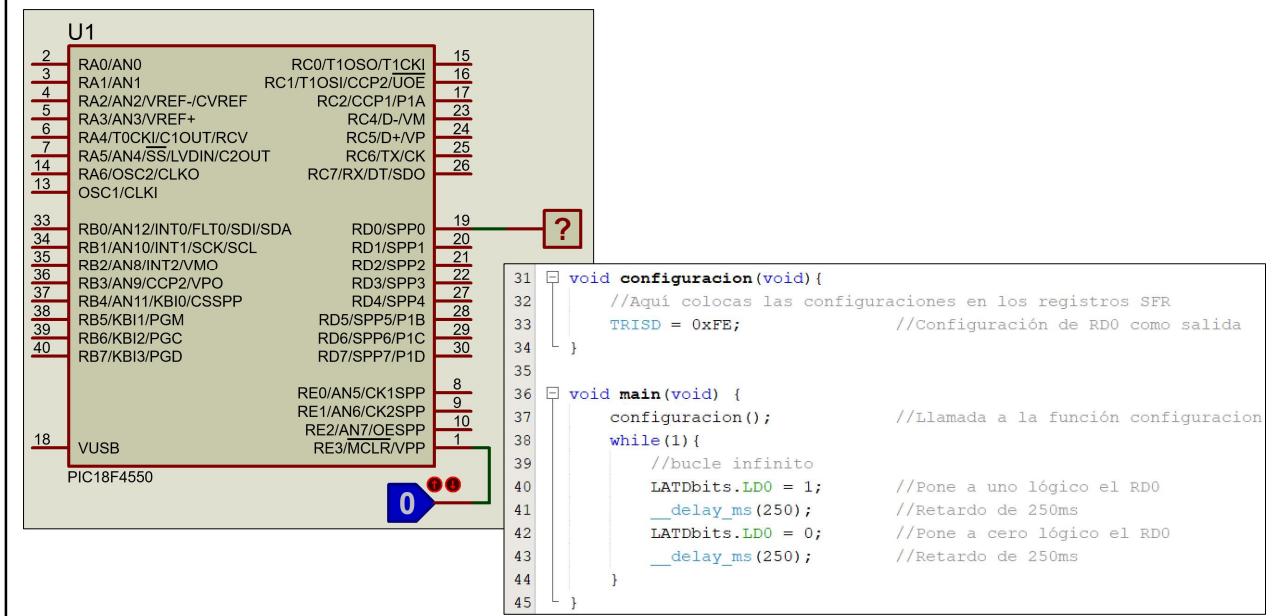
```

10  #pragma config PLLDIV = 1           // PLL Prescaler Selection bit
11  #pragma config CPUDIV = OSC1_PLL2 // System Clock Postscaler Selection
12  #pragma config FOSC = XTPLL_XT  // Oscillator Selection bits
13  #pragma config PWRT = ON          // Power-up Timer Enable bit
14  #pragma config BOR = OFF          // Brown-out Reset Enable bits
15  #pragma config BORV = 3           // Brown-out Reset Voltage bit
16  #pragma config WDT = OFF          // Watchdog Timer Enable bit
17  #pragma config WDTPS = 32768       // Watchdog Timer Postscale Selection
18  #pragma config CCP2MX = ON          // CCP2 MUX bit (CCP2 input/output)
19  #pragma config PBADEN = OFF         // PORTB A/D Enable bit (PORTE)
20  #pragma config MCLRE = ON          // MCLR Pin Enable bit (MCLR pin)
21  #pragma config LVP = OFF           // Single-Supply ICSP Enable bit
22
23  #include <xc.h>
24
25  #define _XTAL_FREQ 48000000UL    //Frecuencia de trabajo 48MHz
26
27  void configuracion(void) {
28      //Aqui colocas las configuraciones iniciales
29  }
30
31  void main(void) {
32      configuracion();
33      while (1) {
34          //Tu programa de usuario
35      }
36  }

```

14

Ejemplo en XC8: Titileo de LED



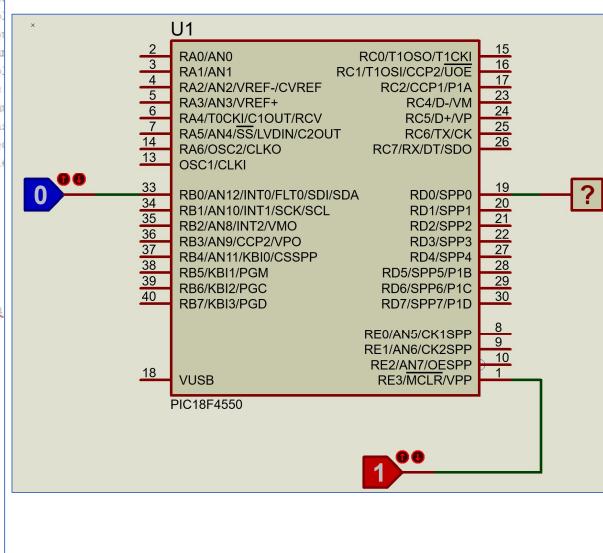
15

Ejemplo en XC8: Negador lógico de un bit

```

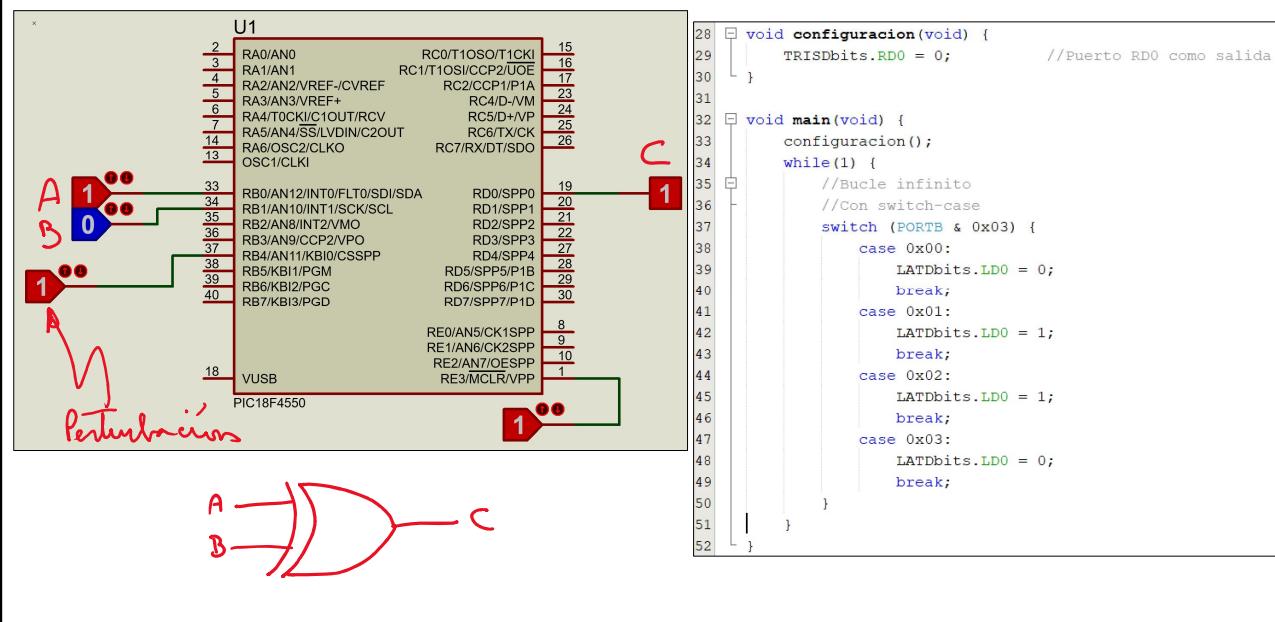
2 #pragma config PLLDIV = 1 // PLL Prescaler Selection bits (No pre
3 #pragma config CPUDIV = OSC1_PLL2 // System Clock Postscaler Selection b
4 #pragma config FOSC = XTPLL_XT // Oscillator Selection bits (XT oscill
5 #pragma config FWRT = ON // Power-up Timer Enable bit (FWRT enab
6 #pragma config BOR = OFF // Brown-out Reset Enable bits (Brown-o
7 #pragma config BORV = 3 // Brown-out Reset Voltage bits (Minimu
8 #pragma config WDT = OFF // Watchdog Timer Enable bit (WDT disab
9 #pragma config WDTPS = 32768 // Watchdog Timer Postscale Select bits
10 #pragma config CCP2MX = ON // CCP2 MUX bit (CCP2 input/output is m
11 #pragma config PBADEN = OFF // PORTB A/D Enable bit (PORTB<4:0> pin
12 #pragma config MCLE = ON // MCLR Pin Enable bit (MCLE pin enable
13 #pragma config LVP = OFF // Single-Supply ICSP Enable bit (Singl
14
15 #include <xc.h>
16 #define _XTAL_FREQ 48000000UL
17
18 void init_conf(void){
19     //Aca colocaremos las configuraciones iniciales de la aplicación
20     //TRISDbits.RD0 = 0; // RD0 como salida
21     asm("bcf TRISD, 0"); // Escribiendo instrucciones en mpasm
22 }
23
24 void main(void) {
25     init_conf();
26     while(1){
27         if (PORTBbits.RB0 == 1) {
28             LATDbits.LD0 = 0;
29         }
30         else{
31             LATDbits.LD0 = 1;
32         }
33     }
34 }

```



16

Ejemplo en XC8: Compuerta XOR



17

Ejemplo: Uso del A/D en XC8 y MPASM

- XC8 vs MPASM
- Se puede apreciar que aparentemente hay menos líneas en XC8
- Al compilar el programa en XC8 se puede ver que ocupa 60bytes en memoria de programa, en MPASM ocupa una cuarta parte

The screenshot shows two side-by-side code editors in MPLAB X.

XC8 Code:

```

1 // Zona de bits de configuración
2 #pragma config FLLDIV = 1           // PLL Prescaler Selection bits (No prescale)
3 #pragma config CPUDIV = OSC1_PLL2 // System Clock Postscaler Selection bits (P
4 #pragma config FOSC = XTPLL_XT    // Oscillator Selection bits (XT oscillator (P
5 #pragma config PWRT = ON           // Power-up Timer Enable bit (PWRT enabled)
6 #pragma config BORV = 3             // Brown-out Reset Selection bits (Brown-out Re
7 #pragma config WDT = OFF           // Watchdog Timer Selection bits (WDT disabled (c
8 #pragma config WDTPS = 32768        // Watchdog Timer Prescaler (WDTPS = 32768)
9 #pragma config CCP2MX = ON          // CCP2 MUX bit
10 #pragma config PBADEN = OFF         // PORTB A/D Enable bit (PBADEN = OFF)
11 #pragma config MCORE = ON          // MCORE Pin Enable bit (MCORE = ON)
12 #pragma config IVE = OFF           // Single-Supply ICSP Enable bit (Single-Supp
13
14 #include <xc.h>
15 #define _XTAL_FREQ 40000000UL
16
17 void init_conf(void) {
18     //Aca colocaremos las configuraciones iniciales
19     TRISD = 0x00; //RD como salidas
20     ADCON2 = 0x24; //8TAD, FOSC/4 ADFM=0
21     ADCON1 = 0x0E; //AN0 habilitado
22     ADCON0 = 0x01; //AN0 seleccionado y AD f
23
24 }
25
26
27 void main(void) {
28     //Aca va la rutina principal
29     init_conf();
30     while(1) {
31         ADCON0bits.GODONE = 1; //Inicia la conversion
32         while(ADCON0bits.GODONE == 1);
33         LATD = ADRESH;
34     }
35 }

```

MPASM Code:

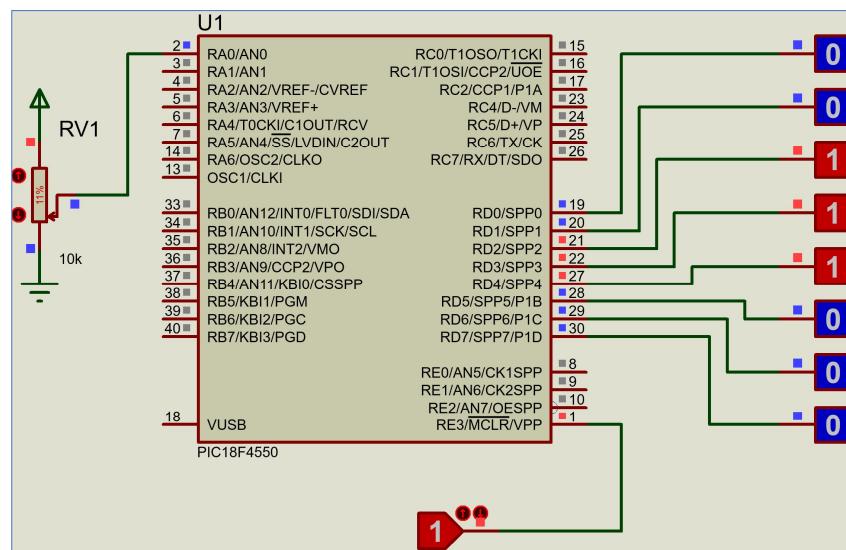
```

3 ;include <p18f4550.h> ;llamada a la librería de nombre de los registros
4
5 ;Directivas de preprocesador o bits de configuración
6 CONFIG PLLDIV = 1 ; PLL Prescaler Selection bits (No prescale
7 CONFIG CPUDIV = OSC1_PLL2 ; System Clock Postscaler Selection bits (P
8 CONFIG FOSC = XTPLL_XT ; Oscillator Selection bits (XT oscillator (P
9 CONFIG PWRT = ON ; Power-up Timer Enable bit (PWRT enabled)
10 CONFIG BOR = OFF ; Brown-out Reset Enable bits (Brown-out Res
11 CONFIG WDT = OFF ; Watchdog Timer Enable bit (WDT disabled (c
12 CONFIG CCP2MX = ON ; CCP2 MUX bit (CCP2 input/output is multiplexed)
13 CONFIG PBADEN = OFF ; PORTB A/D Enable bit (PORTB<4:0> pins are
14 CONFIG MCORE = ON ; MCORE Pin Enable bit (MCORE pin enabled; RE3
15 CONFIG LVP = OFF ; Single-Supply ICSP Enable bit (Single-Supply
16
17 org 0x0000
18 goto init_conf
19
20 org 0x0020
21 init_conf: clrf TRISD ;RD como salidas
22 movlw 0x24 ;8TAD, FOSC/4 ADFM=0
23 movwf ADCON2
24 movlw 0x0E ;AN0 habilitado
25 movwf ADCON1 ;AN0 seleccionado
26 movlw 0x01 ;AN0 seleccionado y AD funcionando
27 movwf ADCON0
28
29 loop: bsf ADCON0, 0 ;Inicio la conversion en AN0
30 otro: btfs ADCON0, DONE ;Pregunto si ya terminó de convertir
31 goto otro ;Aun el ADC se encuentra en proceso de conversion
32 movff ADRESH, LATD ;Ya terminó de convertir y envia el resultado
33 goto loop
34 end
35

```

18

Ejemplo: Uso del A/D en XC8 y MPASM



19

Cuestionario:

- Ventajas y desventajas entre: XC8, CCS-C, mikroC

20

Fin de la sesión