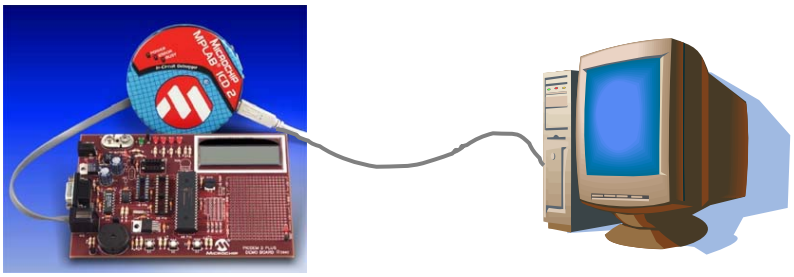
Microcontroladores PIC: Entorno de trabajo MPLAB (v7.xx)



- MPLAB-IDE es un programa software que se ejecuta sobre un PC para desarrollar aplicaciones para microcontroladores de MICROCHIP

- El MPLAB IDE constituye un entorno de desarrollo integrado distribuido gratuitamente por Microchip (fabricante de los microcontroladores PIC) desde su página web: http://www.microchip.com

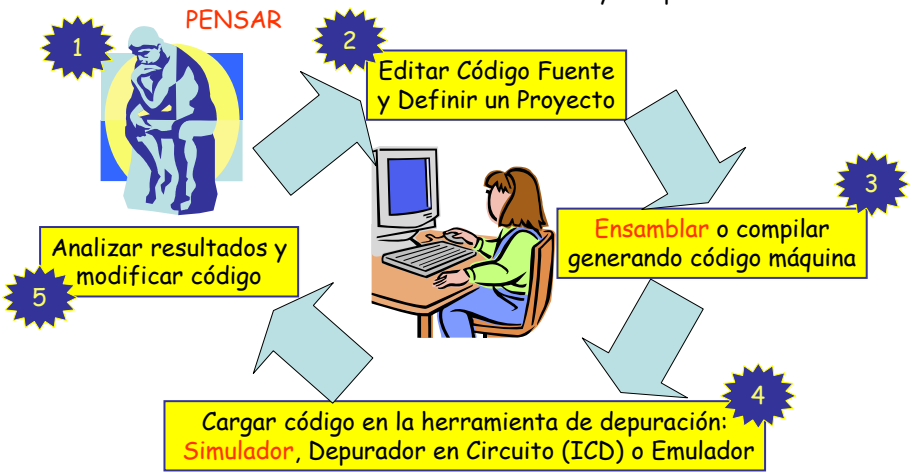
- A continuación se indican los pasos fundamentales a dar para trabajar con este entorno (versión 7.xx) hasta completar el proceso de probar y grabar un programa en la memoria del microcontrolador.

- Se va a trabajar con un código fuente (cuenta.asm), que se toma como primer ejemplo. Se realizará su edición, se definirá un Proyecto que incluirá como código fuente el programa editado y se realizará el ensamblado del mismo.

- Tras la simulación se procederá a comprobar el correcto funcionamiento del código sobre el hardware, utilizando el depurador en circuito MPLAB ICD2 junto con una tarjeta de entrenamiento PICDEM 2 plus

- Por último, se procederá a grabar este programa en un microcontrolador PIC usando también el MPLAB ICD2 como programador

El DISEÑO con microcontroladores constituye un proceso CÍCLICO:



TODO ESTO SE PUEDE HACER DESDE MPLAB

Herramientas accesibles desde MPLAB-IDE:
Integrated Development Environment

- *Project Manager*
  Gestión de Proyectos
- *Editor*
  Edición del texto de los programas escritos en ensamblador ó C
- *Assembler/Linker and Language Tools*
  Ensamblador, Compiladores y Montador de Enlaces
- *Debugger*
  Depuración del código por simulación o sobre el circuito real
- *Programmer*
  Programación o grabación final de microcontroladores

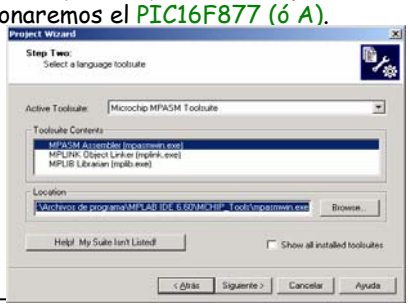
Primeros pasos...

• Una vez iniciado el programa, se utilizará el Editor de texto para escribir el fichero con el código fuente: **File>>New**
después de la edición, se salvará: **File>>Save**

• A continuación se debe crear un PROYECTO, y se sugiere utilizar el asistente (al menos al principio), para lo cual se seleccionará **Project>>Project Wizard...**

• Tras un mensaje de bienvenida, se nos solicita que indiquemos el dispositivo que pensamos utilizar. En este caso seleccionaremos el PIC16F877 (ó A).

• La siguiente ventana pregunta por la herramienta de lenguaje que se va a usar. Seleccionaremos Microchip MPASM Tool-suite (es la opción que sale por defecto) asegurándonos de que el entorno MPLAB conoce la ubicación de los tres ejecutables que necesita: mpasmwin, mplink y mplib.



• A continuación se le asignará un nombre de proyecto (por ejemplo cuenta, sin extensión) y se indicará en qué directorio se va a ubicar. Se recomienda que proyecto y fichero ensamblador se encuentren en el mismo directorio.

• En un nuevo paso, se nos pide indicar los ficheros que se van a incluir en el proyecto que estamos creando. Deberemos buscar el fichero cuenta.asm, seleccionarlo y hacer click sobre el botón **Add>>**. Si no se hubiera creado aún el fichero *.asm con el editor, simplemente haremos click sobre **Siguiente>** ya que se pueden añadir posteriormente

• Con esto ya queda creado el proyecto, y en el entorno MPLAB aparece una ventana (cuenta.mcw) en la que se muestra la información del proyecto que se acaba de definir.

• Se pueden incluir ficheros en el proyecto haciendo click con el botón derecho sobre Source Files en la ventana cuenta.mcw y seleccionar la opción *Add Files...* o bien eliminarlos con *Remove*

¿ Dónde estamos ahora dentro del proceso?

Ensamblado del programa

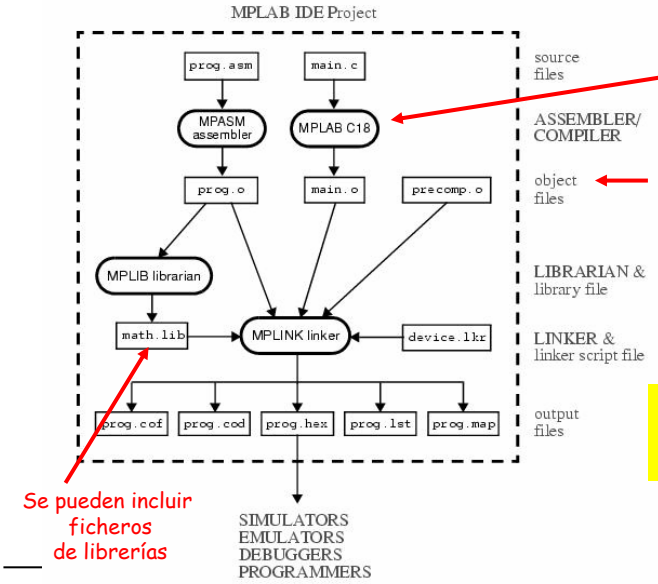
•En esta etapa se realiza el ensamblado del fichero del código fuente y el traspaso de éste a la memoria interna de simulación.

•Para ensamblar el programa basta con hacer doble click en el icono correspondiente, que equivale a seleccionar **Project>>Build All** también se puede ensamblar con **Project>>Make**
(reensambla sólo aquellos ficheros que hayan cambiado desde la última vez)

• Un proyecto puede tener más de un fichero de código fuente, se pueden ensamblar por separado generando código objeto reubicable y luego "ubicarlos" finalmente mediante una herramienta que es el montador de enlaces (MPLINK)

• No será ese nuestro caso ya que utilizaremos normalmente un solo fichero de código fuente que generará código máquina absoluto (con dirección de posicionamiento en memoria de programa ya asignada)

El Entorno MPLAB



Resultaría posible integrar ficheros en C con ficheros en ensamblador

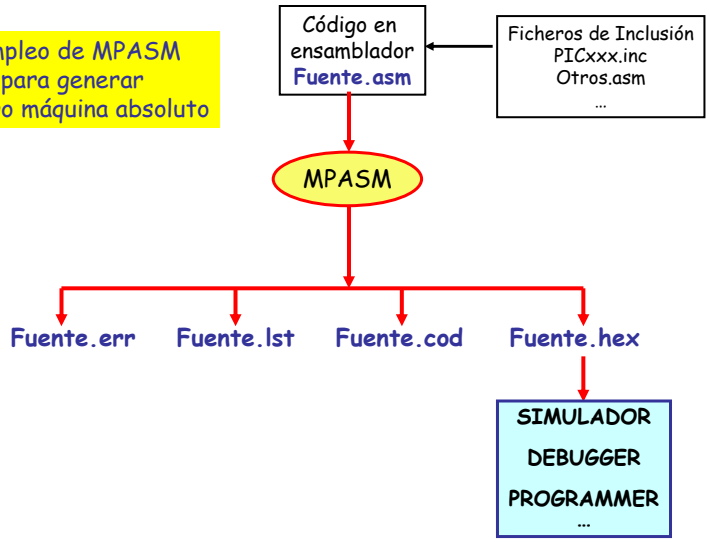
.o códigos objeto reubicables

Empleo de MPLINK para generar código máquina

Se pueden incluir ficheros de librerías

El Entorno MPLAB

Empleo de MPASM para generar código máquina absoluto



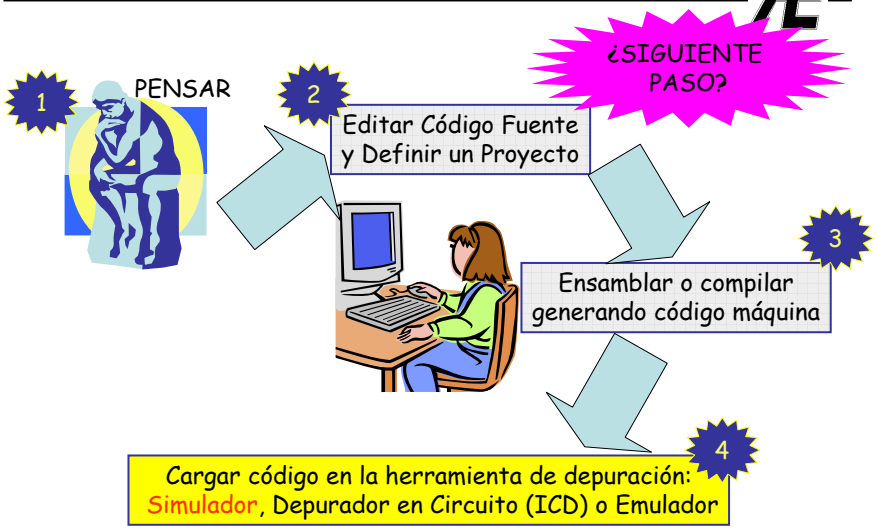
Ensamblando…

- Aparecerá momentáneamente una ventana indicando que se está desarrollando el proceso de ensamblado.

- Al final aparece la pantalla Output en la que se indica si el ensamblado se ha llevado a cabo con éxito (*BUILD SUCCEEDED*) o si, por el contrario, se han localizado fallos (*BUILD FAILED*).

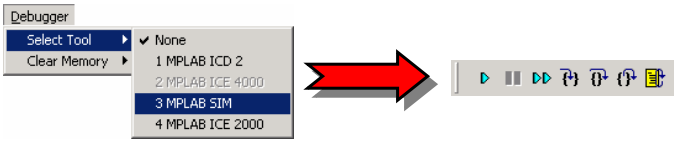
- En caso de existir fallos, se indica de qué tipo son y en qué línea están. Haciendo doble click sobre la línea en la que se muestra esta información, se accede a la posición donde se ha detectado el fallo en el fichero fuente para proceder a la corrección

- Hasta que no desaparezcan todos los errores, no se podrá generar el código máquina en un formato compatible con el resto de herramientas (extensión .hex)



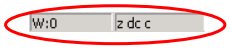
Simulación del programa

• Una vez ensamblado el programan sin errores, simularemos su comportamiento seleccionando la opción **Debugger>>Select Tool > MPLAB SIM**.



• La simulación no es útil si no se visualizan los resultados de la misma. Para ello, se activará una ventana personalizada Watch que permite supervisar el contenido de los registros de interés. Esta ventana se activa mediante **View>>Watch**. Los registros se añaden haciendo click en el botón **Add SFR** ó bien **Add Symbol**.

• En la barra de estado también se muestra información sobre el contenido del registro W y el valor de los flags de STATUS (minúscula = '0').



Condiciones de la Simulación

• MPLAB-Sim es un simulador de eventos discretos.

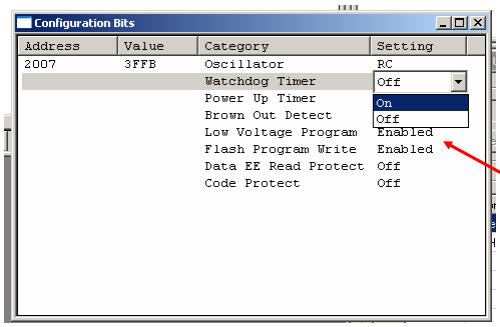
• No se trata de una simulación a la velocidad que desarrollará el microcontrolador. Las instrucciones se ejecutan tan rápido como puede la CPU del computador (PC) donde se esté ejecutando MPLAB. Esto significa que será normalmente más lento que el microcontrolador real trabajando a la frecuencia que marque su oscilador.

• La velocidad de simulación depende de la frecuencia de la CPU del PC y del número de tareas que se estén ejecutando simultáneamente. No tiene ninguna trascendencia el cambio en la frecuencia del micro que se puede configurar en la ventana Debugger > Settings…

• La simulación más rápida posible (Run) será varios órdenes de magnitud más lenta que la ejecución final en el microcontrolador. Los retardos y temporizaciones deben ser reajustados durante la simulación para evitar que se alarguen en exceso

Configuración del Microcontrolador

• Antes de iniciar la simulación, se deberían configurar los bits de configuración del microcontrolador: **Configure > Configuration bits…**



La simulación sólo es software, por lo que las condiciones hardware no pueden darse

• De todos los bits de configuración, durante la simulación sólo podría afectar el correspondiente al *Watchdog Timer* ya que podría resetear el microc. Para evitarlo, se debe desactivar el mismo

Ejecución del código

• Los comandos más importantes para la ejecución (localizados dentro del menú **Debugger**) son los que se indican a continuación.

➢ **Run** (Ejecución continua). Ejecuta el programa constantemente. La ventana Watch no se actualiza hasta que no se detiene la simulación.

➢ **Animate** (Ejecución animada). Ejecuta el programa de forma continua pero actualizando el contenido de la ventana Watch cada vez que ejecuta una instrucción.

➢ **Halt** (Paro). Detiene la ejecución del programa y actualiza todos los valores de las ventanas de visualización.

➢ **Step Into** (Ejecución paso a paso). Ejecuta una sola instrucción del programa y actualiza la información de las ventanas de visualización.

➢ **Reset**. Equivale a un reset por activación del pin /MCLR.

• Otros modos de simulación son los siguientes.

➢ **Step Over**. Ejecuta una sola instrucción del programa y actualiza las ventanas de visualización. Cuando la instrucción es una llamada a una subrutina (*call k*), se ejecuta toda la subrutina antes de actualizar las ventanas.

➢ **Step Out**. Cuando se está ejecutando una subrutina paso a paso, este modo de simulación obliga a que se ejecuten todas las instrucciones de la subrutina hasta regresar al programa principal, momento en el que se detiene la simulación y se actualizan las ventanas de visualización.

➢ **Run to cursor**. Esta opción da lugar a una ejecución continua desde la última instrucción simulada hasta la posición actual del cursor. Se entra en este modo de simulación mediante el menú que se activa con el botón derecho del ratón.

Breakpoints o Puntos de Ruptura

• Otro elemento asociado a la simulación son los puntos de ruptura o *breakpoints*, que constituyen puntos o instrucciones donde el usuario decide que debe detenerse la ejecución del programa.

• Para ubicar un *breakpoint* sobre una línea señalada por el cursor, se seleccionará la opción **Set Breakpoint** del menú que aparece al pulsar el botón derecho del ratón. Otra posibilidad es hacer doble click sobre la línea donde se quiere colocar. En cualquier caso, aparecerá una "B" de color rojo en la posición donde se ha situado el punto de ruptura.

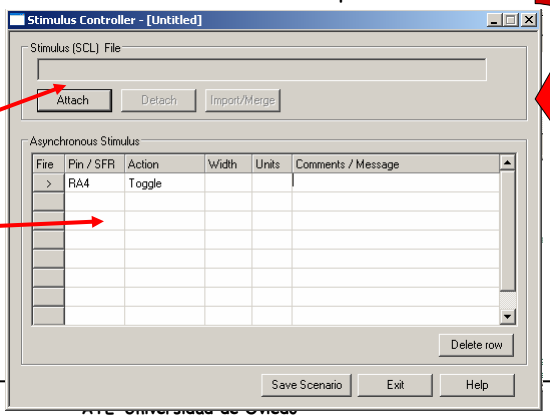
• Al simular una ejecución continua, el programa se detendrá en la instrucción que se ha marcado con el punto de ruptura. Para continuar con la simulación desde ese punto hay que volver a lanzar la simulación.

• Para eliminar puntos de ruptura, basta con hacer doble click sobre la línea en que se encuentran. Si hay muchos, puede resultar más útil seleccionar **Breakpoints ▶ Remove All Breakpoints** en el menú que se activa con el botón derecho del ratón la opción. Este menú también ofrece la opción de activar/desactivar los puntos de ruptura colocados en el programa.

Simulación de entradas

• Para comprobar el correcto funcionamiento de un programa suele ser necesario modificar el valor de determinadas entradas durante la simulación. Para editar los estímulos de una entrada de un puerto hay que seleccionar el menú **Debugger>>Stimulus Controller**
y se puede crear o abrir un "*Scenario*" de entradas para la simulación

En esta ventana se pueden definir tanto estímulos Síncronos (ficheros SCL) como Asíncronos que se definen en esta zona inferior



Estímulos Asíncronos

• Son aquellos eventos que se producen cuando el usuario ejecuta un "click" con el ratón en la zona "Fire" y por tanto, se ignora "a priori" el instante en el que van a producirse

•En la columna Pin/SFR Stimulus se define dónde se va a dar el evento

• Se definen cinco posibles acciones:
- Set High. Pone la entrada a "1".
- Set Low. Pone la entrada a "0".
- Toggle. Cambia de valor cada vez que se pulse. Es la más habitual.
- Pulse High Pulso a 1 de duración definida a la derecha.
- Pulse Low Pulso a 0 de duración definida a la derecha.

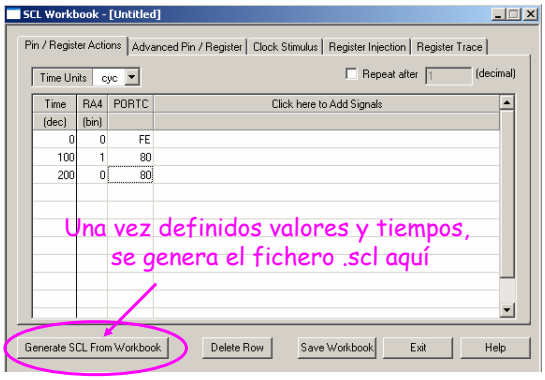
• Cada vez que se pulse al botón **Fire (durante el tiempo de simulación)**, se ejecuta la acción asociada y en el pin correspondiente.

Estímulos Síncronos

- Son aquellos eventos que se producen en un momento conocido y predeterminado por el usuario antes de iniciar la ejecución de la simulación

- Se crea un fichero de extensión .scl que se puede incorporar en el mismo *Scenario* que los estímulos asíncronos, en esa ventana: **Attach**

- Para generar el fichero: **Debugger > SCL Generator** y se puede crear o abrir un *Workbook* donde aparecen varias "pestañas" para definir de diversos modos, valores coincidentes con instantes de tiempo. Ese tiempo se puede medir en ciclos de instrucción o en unidades temporales



"Pestañas" disponibles en *SCL Workbook*:

- *Pin/Register Actions*: para introducir valores en pines o registros coincidentes con instantes de tiempo dados (simple y útil)

- *Advanced Pin/Register:* para introducir valores en pines o registros cuando se den determinadas condiciones (más complejo)

- *Clock Stimulus :* para introducir señales de "tipo reloj" en un pin dado del microcontrolador (que cambian de manera periódica entre 1 y 0)

- *Register Injection*: para introducir valores en determinados registros coincidentes con el acceso a los mismos (Trigger: Demand) o cuando el PC tiene un determinado valor

- *Register Trace*: no es propiamente para introducir estímulos, sino que se emplea para salvar en un fichero de texto el contenido de ciertos registros durante la ejecución de la simulación

EJERCICIO PROPUESTO:

Llevar a cabo la simulación del programa que cuente en binario y saque por el PORTB el nº de veces que se pulsa el pulsador conectado a RA4

- Visualizar al menos el contenido de los puertos A y B.
- Configurar la ventana de visualización para poder ver los registros anteriores en formato binario.
- Definir una entrada asíncrona en el pin 4 de PORTA.
- Simular el programa y comprobar su funcionamiento.
- Practicar distintos modos de simulación y la utilización de puntos de ruptura.

ALGORITMO

Inicialización:

- PORTB se define como puerto de salida y PORTB se pone a cero al principio
- PORTA como puerto de entrada (por defecto son todos así)

Bucle de Ejecución continua

```
Si la tecla está pulsada (RA4==0) entonces
      Llamada a Subprograma de Incremento
si no
      Volvemos al principio del bucle
```

Subprograma de Incremento

```
Sumamos 1 a la combinación presente en el PORTB
Bucle de espera a que la tecla esté liberada
Si la tecla está pulsada (RA4==0) entonces
      Seguimos esperando dentro del bucle
si no
      Retornamos del Subprograma
```

El Entorno MPLAB

CÓDIGO FUENTE

```
;Fichero CUENTA.ASM
;
;Programa de Prueba para la placa PICDEM-2 plus
;Por el Puerto B se saca en binario, el numero de veces
;que se pulsó la tecla que está conectada a la entrada RA4
;si pulsada a cero y si libre a 1
;
                LIST    P=16F877            ;Directiva para definir listado y microcontrolador
                INCLUDE P16F877.INC         ;Inclusión de fichero de etiquetas

                ORG     0
                BSF     STATUS,RP0          ;Paso al banco 1 de la memoria de datos
                CLRF    TRISB               ;para definir el PORTB como salida
                BCF     STATUS,RP0          ;Volvemos al banco 0
                CLRF    PORTB               ;Ponemos a cero el PORTB para que aparezca ese
                                            ;valor cuando se defina como salida

ESPERA          BTFSS   PORTA,4             ;Esperamos a que se pulse la tecla
                CALL    INCREMENTO          ;en cuyo caso RA4 pasa a 0 y vamos a
                GOTO    ESPERA              ;subprograma de INCREMENTO

;Subprograma de INCREMENTO

INCREMENTO      INCF PORTB,F                ;Si se pulsó incrementamos PORTB

SOLTAR          BTFSS PORTA,4               ;no salimos hasta que se haya soltado
                GOTO SOLTAR                 ;la tecla, en ese caso RA4 pasaría a 1
                RETURN                      ;y volvemos al programa principal
                END
```


El Entorno MPLAB

SI FUNCIONA SOBRE EL SIMULADOR…

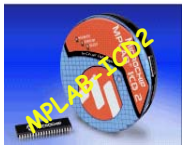
¿ Funcionará sobre la tarjeta real de la aplicación ?

NO SE PUEDE ASEGURAR

EL SIMULADOR TRABAJA SOBRE SEÑALES LÓGICAS
Y
NO TRABAJA EN TIEMPO REAL

EL MICROCONTROLADOR
DEBE TRABAJAR CON SEÑALES ELÉCTRICAS
Y EN TIEMPO REAL

¿UNA HERRAMIENTA QUE ME APROXIME MÁS AL MUNDO REAL?



¿ Qué es el MPLAB-ICD2 ?

- Es un Depurador de microcontroladores PIC para trabajar sobre el micro ya insertado dentro del circuito de la aplicación y es a la vez un Programador de esos dispositivos microcontroladores.

- Como *Debugger*: permite la ejecución (controlada desde el entorno MPLAB) de programas en tiempo real y utilizando los recursos internos del propio microcontrolador pero con posibilidad de parar, ejecutar paso a paso, ver el estado de los registros internos, establecer puntos de ruptura, etc.

- Como Programador: permite la grabación final del programa de usuario en la memoria del microcontrolador

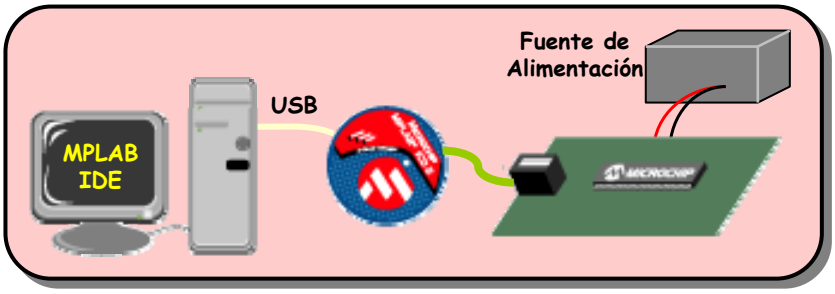
- ICD2 se conecta por un lado a la placa del microcontrolador mediante un conector RJ45 y por otro lado al PC mediante un puerto serie o USB

Depuración con MPLAB ICD 2

- Permite comprobar el funcionamiento del programa sobre la tarjeta en la que se va a colocar el microcontrolador.

- El control de la depuración se lleva a cabo con los mismos controles que se usan durante la simulación (Run, Animate, Halt, …).

- Es preciso conectar el dispositivo entre el PC y la PCB que albergará el PIC.



¿ ICD2 cómo funciona como DEPURADOR ?

- Su utilidad reside en la capacidad que tienen ciertos microcontroladores PIC para trabajar en un modo especial denominado "modo depuración".

- En ese modo, en la memoria de programa de estos dispositivos "conviven" el programa del usuario y una "ejecutiva de depuración" (*Debug executive*) que está en la parte final de la memoria

- Existe un registro interno en el MCU denominado Registro Especial de Depuración que se compara con el Contador de Programa (PC) actual.

- El Registro de Depuración se carga vía serie a través de las líneas PGC y PGD del microcontrolador desde MPLAB-IDE a través de ICD2

- Una vez que se ha ejecutado la instrucción que se encontraba en la posición señalada por el Reg. de Depuración, se "dispara" el mecanismo de depuración: se detiene el programa de usuario y el PC pasa a apuntar a la zona de código de la ejecutiva de depuración (muy parecido a una interrupción)

…¿ ICD2 cómo funciona como DEPURADOR ?

- La actuación sobre el comando Halt desde MPLAB-IDE, provoca que el contenido actual del PC se cargue en el Registro de Depuración

- MPLAB ICD2 se comunica con el microcontrolador mediante la ejecutiva de depuración y físicamente a través de las líneas PGC y PGD.

- Le transfiere al entorno MPLAB-IDE la información del estado del microcontrolador en el punto de ruptura correspondiente.

- El entorno MPLAB-IDE envía una serie de "preguntas" que son respondidas por la ejecutiva de depuración sobre el estado de los registros y del núcleo del microcontrolador y se muestran al usuario del entorno.

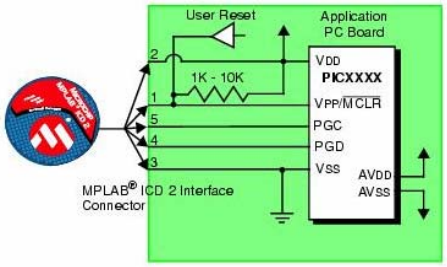
- Al ser una aplicación, la ejecutiva de depuración precisa de ciertos recursos del microcontrolador: memoria de programa, registros y una posición de la pila hardware

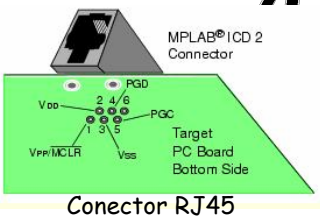
Recursos que va a usar el MPLAB-ICD2 en modo depuración

Para el caso de los microcontroladores PIC16F877:

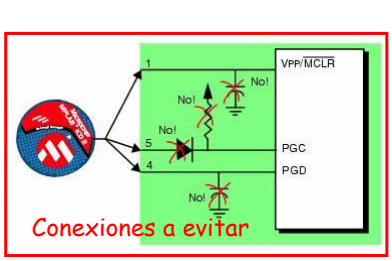
- El pin MCLR está reservado para depuración
- No está permitida la programación a baja tensión (Low Voltage ICSP)
- Pines RB6/PGC y RB7/PGD para transferencia serie
- Un nivel de la pila hardware
- Las 256 últimas posiciones de la memoria de programa (0x1F00 a 0x1FFF)
- 15 Registros RAM, direcciones: 0x70, 0xF0, 0x170, 0x1F0, 0x1E5-0x1EF

Conexiones del MPLAB-ICD2 al microcontrolador

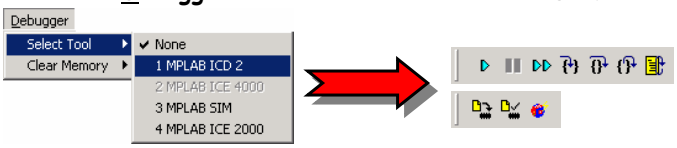




Conector RJ45



• Para lanzar el depurador es preciso seleccionar la opción
Debugger>>Select Tool ▶ MPLAB ICD 2.



• Antes de establecer la conexión, puede resultar adecuado configurar el equipo entrando en **Debugger>>Settings**… y seleccionando las siguientes pestañas.

- **Communication.** Indicamos que vamos a conectar el ICD 2 al puerto USB. También podría utilizarse alguno de los puertos serie (COMn).
- **Program.** Comprobamos que está activa la opción *Allow ICD 2 to select memories and ranges* para hacer que sea el depurador el que decida qué zonas de memoria deben grabarse en el PIC en función del tamaño del programa ensamblado. Se evita así tener que grabar toda la memoria de programa del microcontrolador (8K)

• A continuación es necesario establecer la conexión con el depurador, para lo cual se selecciona **Debugger>>Connect** o se hace click sobre el icono.
ES IMPORTANTE QUE ESTÉ SELECCIONADO EL DISPOSITIVO QUE ESTÁ EN LA PLACA REALMENTE: **Configure>>Select device**…

• Si ha habido errores de conexión, deberá verificarse el cableado del equipo y la configuración indicada en **Debugger>>Settings**… , fijándose ahora también en las pestañas

- **Status.** Nos indicará si la conexión se ha hecho de manera adecuada. Permite además configurar el sistema para que esta conexión se establezca de manera automática.
- **Power.** Indica los valores de las tensiones del sistema. Para que todo funcione correctamente se necesita Target Vdd≈5V (si no es así, la placa está mal alimentada), Target Vpp≈13V y MPLAB ICD 2 Vpp≈13V.

• Una vez establecida la conexión, el MPLAB ICD 2 ya está listo para transferir nuestro programa al microcontrolador que se encuentra en la placa de la aplicación, lo que permitirá llevar a cabo la **depuración**.

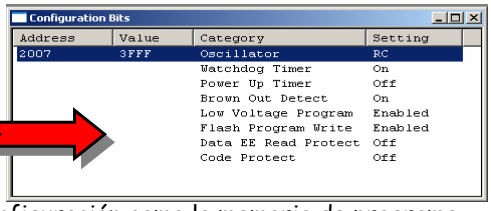
• En modo depuración se puede ejecutar, parar, transferir contenidos, etc.

• La **depuración con MPLAB ICD2** se puede realizar porque el microcontrolador admite trabajar en modo depuración cuando el **bit DEBUG** de la palabra de **configuración** se graba como **'0'**

• Si el microcontrolador está trabajando en **modo depuración** hay ciertos **recursos** que necesita este modo de trabajo: 2 pines (RB6 y RB7), 1 posición de la pila hardware del PC, las últimas 256 posiciones de la memoria de programa y varias posiciones de RAM 0x070 (0x0F0, 0x170, 0x1F0), 0x1EB-0x1EF

• Se deben cargar los bits de la palabra de configuración:

Configure >> Configuration bits…



• Para grabar tanto la palabra de configuración como la memoria de programa (tras haber ensamblado) se seleccionará **Debugger>>Program**.

PROGRAMAR

• Una vez programado el dispositivo, se procede del mismo modo que se hizo con el simulador con la diferencia de que ahora, además de poder ver el valor de los registros en las ventanas de visualización, **comprobaremos el funcionamiento real en la placa**.

Run : para lanzar la ejecución continua del programa

Animate : ejecución con animación (actualización de posición y reg.)

Halt : parada de la ejecución

Reset : reinicio del microcontrolador

Step into : ejecución paso a paso

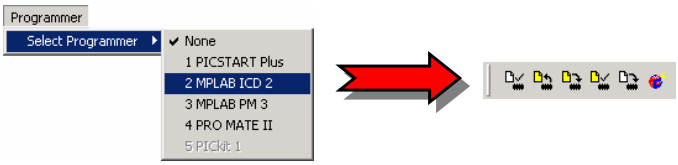
Step over : paso a paso ejecutando los subprogramas completos

Step out : ejecución hasta que se sale de subprograma

Grabado de PICs con MPLAB ICD 2

• Además de llevar a cabo procesos de depuración, el MPLAB ICD 2 permite grabar los microcontroladores que se incluirán en la placa final.

• Para ello, hay que seleccionar este dispositivo como elemento programador, lo cual se consigue con **Programmer>>Select Programmer ▶ MPLAB ICD 2**.



• El MPLAB ICD 2 no puede funcionar como depurador y programador al mismo tiempo. Por ello, para seleccionar el MPLAB ICD 2 como programador, deberemos asegurarnos de que no está seleccionado como depurador (y viceversa).

• Como Programador no necesita transferir al micro la ejecutiva de depuración

• Las opciones de grabación que se ofrecen son las siguientes:

- **Program**. La que se usa para grabar el PIC. Es la más habitual.
- **Read**. Lee el programa que está grabado en el PIC y lo carga en la memoria de programa del entorno MPLAB.
- **Verify**. Comprueba que la grabación se ha efectuado correctamente.
- **Erase Part**. Borra completamente el PIC antes de programarlo.
- **Blank Check**. Comprueba que el PIC está borrado.

• Después de programado, el PIC puede desconectarse del MPLAB-ICD2 y empezará a trabajar como dispositivo autónomo que es

• Todos los recursos del microcontrolador estarán disponibles

• El programa comenzará a ejecutarse en cuanto el microcontrolador reciba la tensión de alimentación.