

Práctica 1

Introducción al laboratorio y al ensamblador del PIC

1. Objetivos

El objetivo de esta primera sesión es que el alumno aprenda el manejo de las herramientas que se utilizarán en el laboratorio durante el transcurso de la asignatura. Para ello se mostrará como crear un proyecto mediante el software de desarrollo MPLAB y se realizan las primeras experiencias básicas de programación del microcontrolador usando lenguaje ensamblador.

En concreto, en esta primera sesión se utilizarán los puertos de entrada/salida digitales. El programa propuesto tomará las entradas por el puerto A y mostrará la salida por la barra de leds y el display simple de 7 segmentos conectado al puerto B. También se realizarán algunas pruebas con operaciones en ensamblador, lectura de tablas y su simulación.

2. Introducción

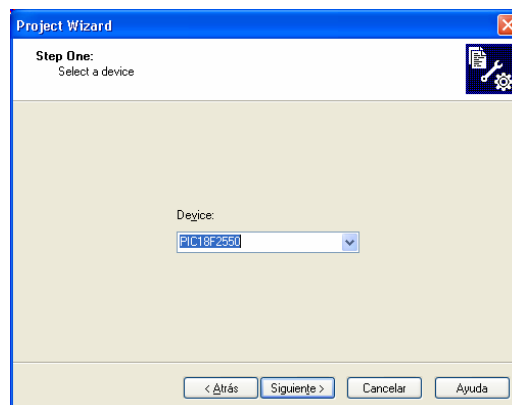
En esta primera sesión utilizaremos las herramientas de Microchip para ensamblar y simular los programas en ensamblador. También se utilizará la tarjeta EduMic junto con su programa de descarga, desarrollados en la Universidad de Valencia, para realizar los diferentes experimentos de todas las sesiones prácticas. También se discute en esta sección la teoría básica para la realización de los diferentes trabajos que se piden.

2.1 Ensamblar y simular

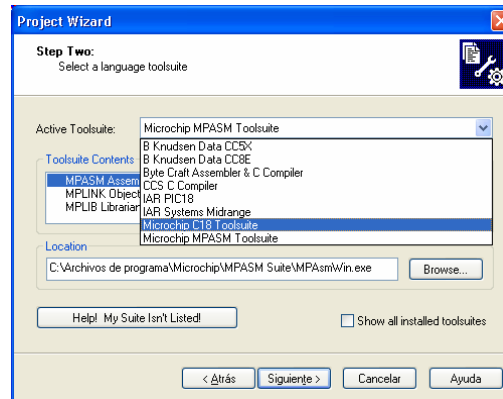
Microchip tiene la herramienta MPLAB de compilación y simulación integrada de manera que la realización de programas en ensamblador y su simulación es muy simple. Es una herramienta para MS Windows y su funcionamiento es tan simple como crear un nuevo proyecto, añadirle el fichero en ensamblador que se quiera compilar y finalmente compilarlo para generar el fichero en formato HEX que es el que se guarda en el PIC para hacerlo funcionar. La versión que se describe a continuación es la 7.01.

Crear un nuevo proyecto:

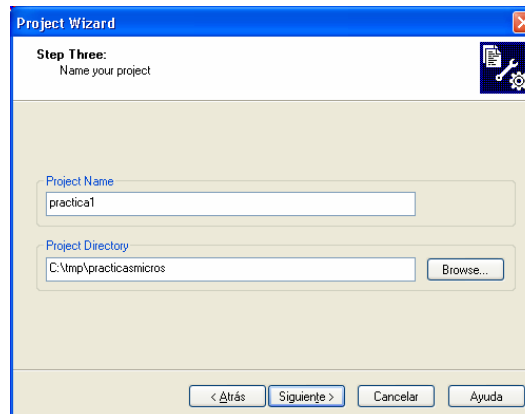
Para crear un nuevo proyecto se puede utilizar el asistente (*wizard*) *Project > Project wizard* que nos va pidiendo paso a paso toda la información para crear el proyecto. Lo primero que nos pide es el microcontrolador que vamos a usar. En las prácticas de este curso deberemos seleccionar el PIC18F2550.



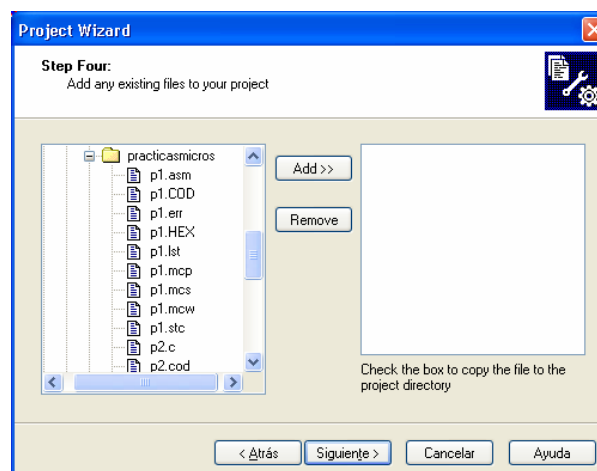
Después de pulsar siguiente, la segunda opción que nos pide es la herramienta que vamos a utilizar para compilar el proyecto. Si se utiliza ensamblador la opción seleccionada debe ser *Microchip MPASM Toolsite*, si utilizamos el compilador de C (opción que se usará en próximas prácticas) *Microchip PIC18 Toolsite*.



También se pueden modificar los directorios de búsqueda para los ejecutables del programa compilador, enlazador, ensamblador y librarian (no debe ser necesario). El siguiente paso pide el nombre del proyecto y el directorio donde se almacenarán todos los ficheros asociados al proyecto, por ejemplo podemos utilizar “practica1”. Según el nombre que hayamos elegido así se llamará el fichero en formato HEX que se genere.



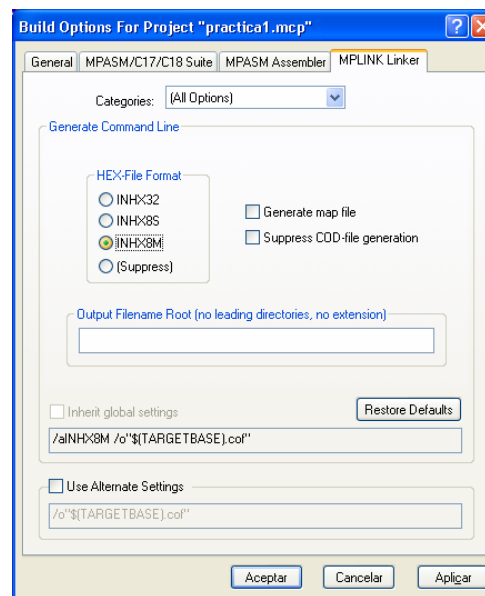
A continuación nos pide que incluyamos los ficheros fuente que forman parte del proyecto. Si todavía no hemos creado los ficheros con el código ensamblador o C del proyecto esta opción la dejaremos vacía.



Al finalizar el asistente para la creación del proyecto se crean los ficheros *.mcp y *.mcw donde se guardan las opciones asociadas.

A continuación se crearán los ficheros fuentes nuevos fichero haciendo *File>New*. En este fichero meteremos el programa en ensamblador y lo salvaremos con cualquier nombre siendo la extensión "asm" para indicar que se trata de un fichero en ensamblador. Cuidado al salvar este fichero y el proyecto para hacerlo en nuestro directorio de trabajo y no en otro lugar fuera de él. Una vez salvado el fichero lo podemos añadir al proyecto haciendo *Project > Add Files to Project...*

También es necesario cambiar el formato del fichero de salida que crea el linker. Para ello se selecciona en la ventana *Project > Build Options > Project* de edición del proyecto para hacer una pequeña modificación al elegir el formato de salida Intel (HEX) adecuado. Dentro de las opciones referentes al MPLINK Linker seleccionaremos dentro del menú "Hex-file Format" la opción **INHx8M**.



Llegados a este punto se puede ensamblar el proyecto con *Project > Build All*. Si todo ha ido bien se genera un fichero con extensión HEX que será el que se utilice en la descarga al PIC.

Para simular el programa:

Para simular el programa recién realizado hay que habilitar el herramienta de simulación MPLAB-SIM dentro del menú *Debugger > Select Tool > 3 MPLAB-SIM*. Ahora se puede simular la ejecución del programa paso a paso con *Debugger > Step Into (F7)* o todo de golpe con *Debug > Run (F9)*. Para ver los diferentes elementos del PIC podemos ir al menú *View* donde podremos seleccionar una serie de ventanas que nos muestran los registros, la memoria de programa, los registros especiales, etc. y que nos van a permitir averiguar si nuestro programa hace lo que esperábamos.

2.2 Estructura básica de un programa en ensamblador

En el siguiente ejemplo se muestra un pequeño programa en ensamblador que lee el puerto A y copia los 4 bits más bajos en el puerto B, los 4 bits más altos del puerto B siempre se ponen a cero. Para hacer esto primero programa el puerto A de entrada y el puerto B como salida. Se muestran también las directivas básicas de compilación que deberemos usar en cualquiera de nuestros programas:

```
; Programa básico de ejemplo
```

```

list      p=18F2550      ; Directiva para indicar el tipo de PIC

include   "P18F2550.INC"      ; Definiciones de registros internos

org       0x00           ; Siempre empieza en la direccion cero
goto     inicio         ; Saltamos al inicio del programa

org       0x50           ; Dejamos espacio para las interrupciones
inicio

movlw    0x0F           ; Con un 0x0F en ADCON1 se programan
movwfm  ADCON1,0       ; todos los bits de PORTA y PORTB como digitales.
clrf    TRISB,0        ; Puerta B se configura como salida
movlw   b'00011111'    ; Solo cambiamos los bits presentes
movwfm  TRISA,0        ; Puerta A se configura como entrada

repite

movf     PORTA,W,0     ; Lee el Puerto A y lo pone en W
andlw   0x0F          ; W = W(AND)00001111, dejamos solo 4 bits mas bajos
movwfm  PORTB,0       ; Sacar por el Puerto B el registro W
goto    repite        ; Se repite para siempre

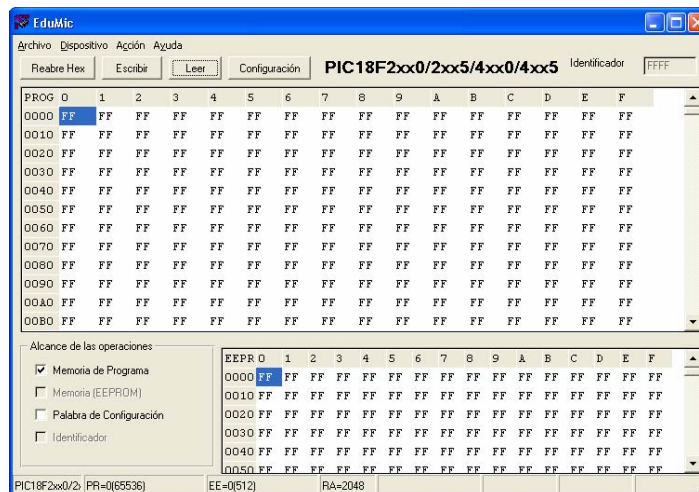
end      ; Fin del programa fuente

```

El propio programa se explica por sí mismo. Las directivas al inicio del programa nos indican el PIC que se está utilizando, sirven para especificar la palabra de configuración, etc. Es conveniente ponerlas todas aunque no es imprescindible. El `include` es muy importante, ya que en él se encuentran todas las definiciones referidas a este procesador; si no se incluyera ese fichero no se podrían utilizar los nombres TRISB, PORTA, etc.

2.3 Programar el PIC

En la página web de la asignatura se puede encontrar el manual completo de la tarjeta EduMIC que incluye el software de programación. Este manual es muy interesante consultarlo para comprender el funcionamiento de la tarjeta y todos sus componentes. El programa de descarga se encuentra en el menú de programas en EduMIC. Al ejecutarlo aparece una ventana con diferentes opciones. Lo primero que hay que configurar en el programa es el dispositivo, para ello hay que desplegar el menú dispositivo y seleccionar el microcontrolador **PIC18F2xx0/2xx5/4xx0/4xx5**.



A continuación hay que cargar el fichero HEX (menú Archivo) y establecer la palabra de configuración (menú Configuración). Es importante que la palabra de configuración **config1** sea **'0C00'**. De esta forma se selecciona el oscilador primario sin PLL y sin predivisor, y la señal del microcontrolador será directamente la generada por el oscilador primario de las tarjetas, que es de 4 Mhz. En todas las prácticas habrá seguir manteniendo este valor de configuración a **'0C00'**.

Una vez cargado se puede programar el PIC de la tarjeta. Para ello hay que asegurarse que el conmutador de programación de la tarjeta EduMIC se encuentra en modo "programa". Después sólo tenemos que hacer

“Escribir”, comprobando que todo funciona correctamente. Por defecto sólo se programa la memoria de programa, pero se debe activar aunque sólo sea una vez la escritura de la palabra de configuración.

Una vez programado el PIC hay que poner el conmutador de programación en modo normal para ver si funciona correctamente.

2.4 Barra de Leds, display de 7 segmentos e interruptores de entrada

Estos son los periféricos básicos que se manejan en la sesión de hoy. La barra de leds y el display de 7 segmentos en la tarjeta EduMIC está conectada al puerto B del PIC, mientras que los cinco interruptores digitales están conectados a los bits bajos del puerto A.

La figura 1 muestra la conexión del display de 7 segmentos y la barra de leds al puerto B. Esto nos permite saber qué bit del puerto B enciende qué led o qué segmento en la barra y el display respectivamente.

Para que estos dispositivos estén disponibles hay que poner el conmutador correspondiente (DISPLAY/LED) en su posición ON (encendido).

La figura 2 muestra los interruptores que ponen a cero o uno las entradas del puerto A. Es muy importante darse cuenta de que en la tarjeta EduMIC al puerto A llegan tanto estas entradas digitales como las entradas analógicas, por lo que es imprescindible poner los conmutadores de selección (ANALOG/LIBRE/DIGITAL) en su posición (DIGITAL). El resto de dispositivos es conveniente tenerlos desconectados para no interferir.

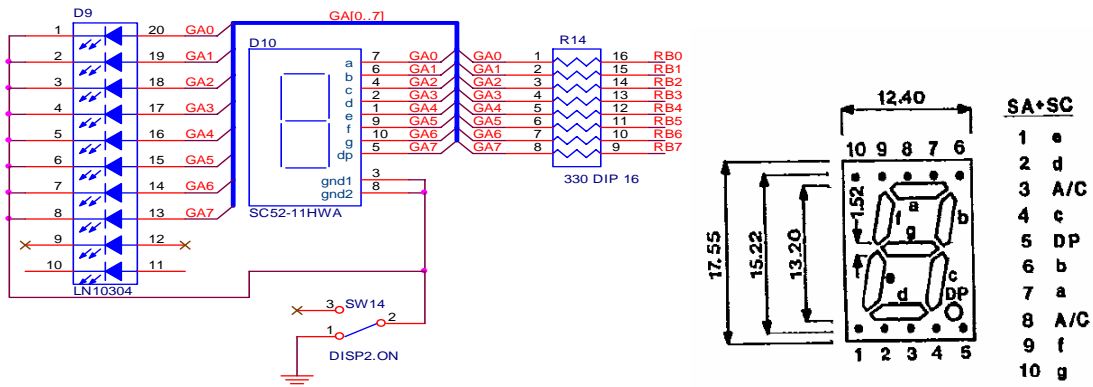


Figura 1: Conexión del display de 7 segmentos y la barra de leds

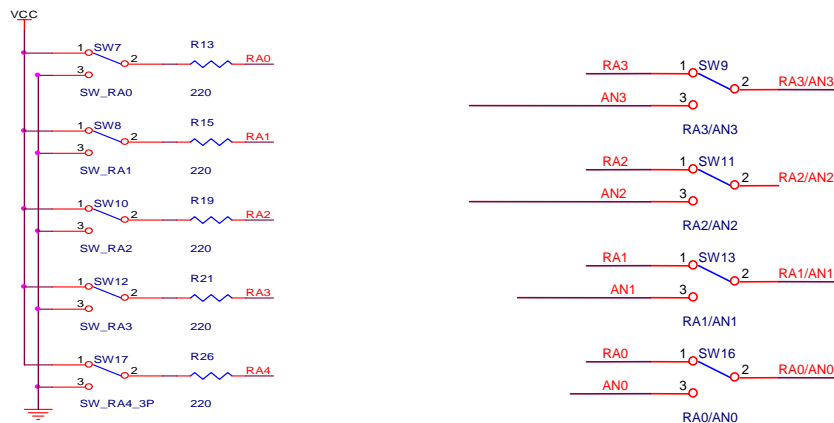


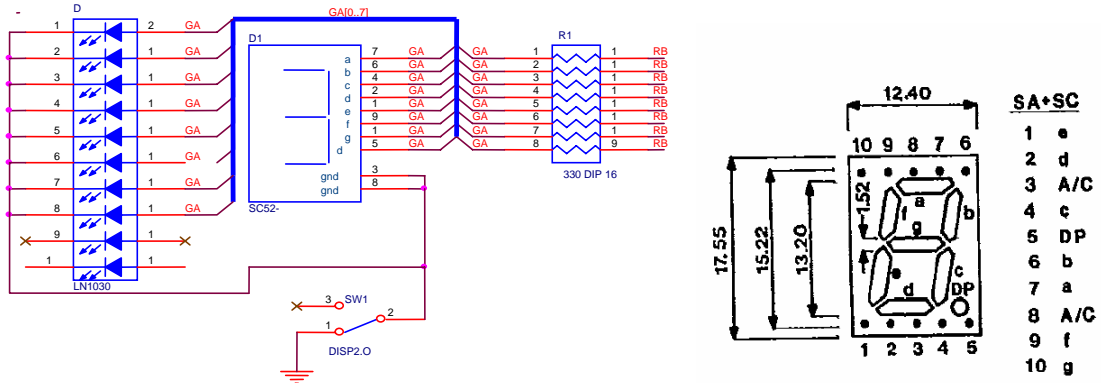
Figura 2: Interruptores de entrada digital (izq.) y selectores Analógico/Digital (der.) para el puerto A

3. Trabajos a realizar

En todas las sesiones de laboratorio se pide la realización de unos programas o tareas. **Es imprescindible que se tengan preparados, o al menos esbozados, los programas al principio de cada sesión.** Al inicio de la sesión el profesor comprueba de que efectivamente los programas han sido preparados y se pone la nota correspondiente. Tener los programas preparados al inicio de la sesión es muy importante, de lo contrario, es muy difícil que en una misma sesión se puedan terminar los programas habiendo probado su funcionamiento.

Programa opcional: Operaciones básicas de entrada/salida

Se pide realizar un programa que lea el puerto A y muestre el valor binario de los 4 bits bajos por el display de 7 segmentos conectado al puerto B. Por ejemplo si en el puerto A se tiene el valor de entrada '0000 0011' en el display de 7 segmentos conectado al puerto B se mostrará:



Para resolver este programa se puede utilizar el acceso indexado a una tabla (se muestra un ejemplo a continuación) donde se guardan los valores que se desean obtener indexados por un índice. En el siguiente ejemplo se muestra una rutina que devuelve un valor dependiendo del índice que se pase a la subrutina tabla dentro del registro WREG, el valor de WREG debe estar en un intervalo de de 0 a 30 y sólo podría tomar valores pares. Esta misma rutina, con pequeñísimas variaciones, puede ser usada para devolver los leds que se deben encienden en el display de 7 segmentos dependiendo del valor de entrada en el puerto A.

```
; Ejemplo de acceso a tabla
; ...

                ; Supongamos que inicialmente WREG=3
                ; el valor en el acumulador se multiplica x2
                ; mediante un desplazamiento lógico a izquierdas
                ; ya que cada retlw ocupa 2 posiciones de memoria
                ; al final:-> (WREG=6)

                call    tabla    ; Llamamos a tabla y nos devuelve en W
                                ; el contenido de la posición 3.

org    0x100
tabla
movff PCL,0    ; antes de saltar hay que leer PCL e iniciar PCLATH:PCLATU
addwf PCL,F    ; Sumamos PCL y W, el resultado a PCL-> PCL=PCL+W

retlw 0x00    ; con lo que se salta a la posición
retlw 0x01    ; indicada por W
retlw 0x01
retlw 0x02----> ; PCL=PCL+W Este (0x02) es el valor devuelto en WREG.
retlw 0x01
retlw 0x02
retlw 0x02
retlw 0x03
retlw 0x01
retlw 0x02
retlw 0x02
retlw 0x03
retlw 0x02
retlw 0x03
retlw 0x02
retlw 0x03
retlw 0x02
retlw 0x04
```

Es necesario en el acceso a tabla tener en cuidado al incrementar PCL (instrucción `addwf PCL,F`) para que no se produzca un acarreo, ya que este incremento se debería llevar sobre PCLATH y PCLATU para saltar a la dirección correcta. Se puede evitar este desbordamiento situando la tabla en una posición múltiplo de 256=> `org 0x___00`, cualquier dirección cuyos dos dígitos hexadecimales más bajos sean '00', y limitando el tamaño de la tabla para que junto con las posiciones que ocupan las dos primeras instrucciones (`movff PCL,0` y `addwf PCF,F`) la tabla no supere los 256 bytes.