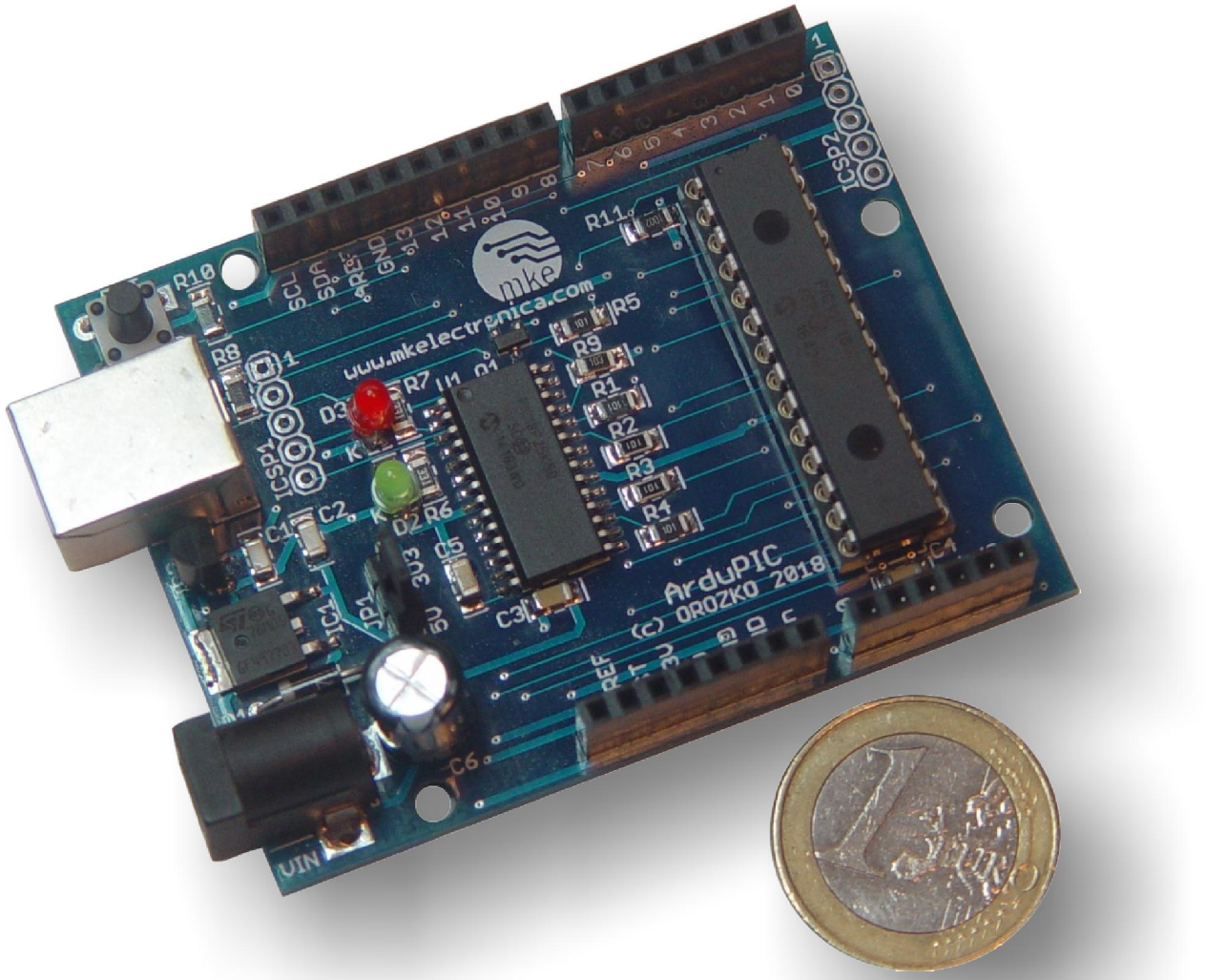
Ardu PIC

Manual de usuario



Rev. Febrero 2018

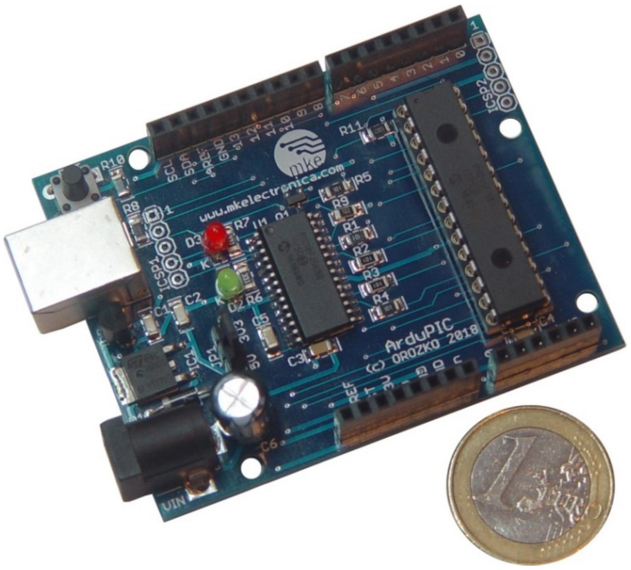
MK Electrónica

C/ Ipergorta, 11 - 3ºB
48410 Orozko - Bizkaia
Tfno.: 34-944 04 80 89

email: info@mkelectronica.com
www.mkelectronica.com

ArduPIC: Manual de usuario

1. INTRODUCCIÓN

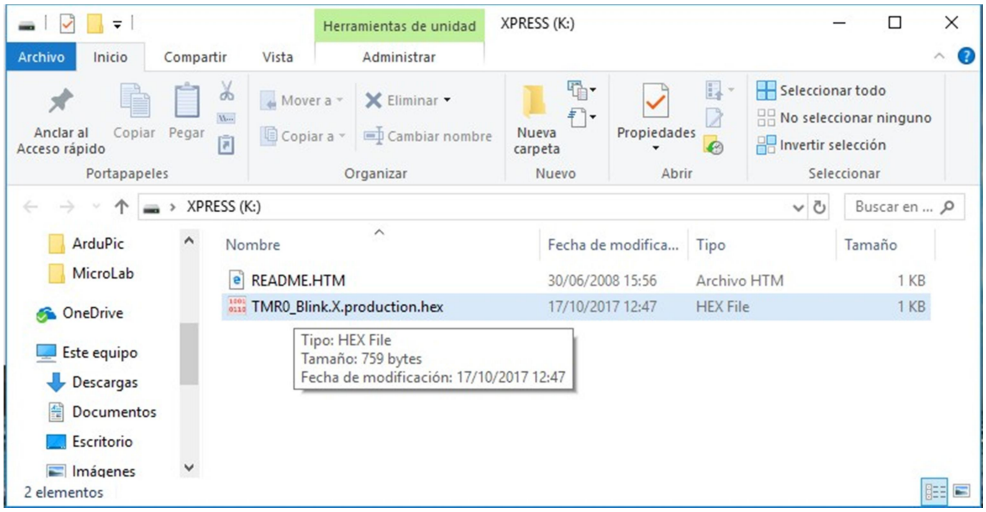


¿Eres un enamorado entusiasta de los microcontroladores PIC de Microchip? Si la respuesta es afirmativa nuestra tarjeta controladora **ArduPIC** diseñada, fabricada y comercializada por MK Electrónica, te puede servir para cubrir buena parte de tus necesidades e inquietudes.

Incluye un microcontrolador de 8 bits de última generación como es el PIC16F1855 que integra un gran número de periféricos y recursos internos. Además, como ArduPIC es una tarjeta controladora pin compatible con Arduino, podrás utilizar la mayor parte de los shields disponibles en el mercado.

Por otra parte, para el diseño, edición y puesta a punto de tus programas podrás utilizar la herramienta MPLABX IDE original de Microchip. Esta herramienta 100% gratuita incluye un completo entorno de desarrollo o IDE, con gestor de proyectos, editor de programas fuentes, simulador/depurador y mucho más. También incluye el lenguaje ensamblador MPASM, la versión gratuita del compilador XC8 y el generador de códigos de configuración MCC.

El sistema de grabación del controlador es muy novedoso. Cuando conectas ArduPIC a tu PC a través de un puerto USB, ésta aparece como una nueva unidad de disco como por ejemplo **XPRESS(K:)**. Basta con que arrastres y sueltes sobre ella ("*Drag and Drop*") el fichero ejecutable *.HEX que deseas grabar.

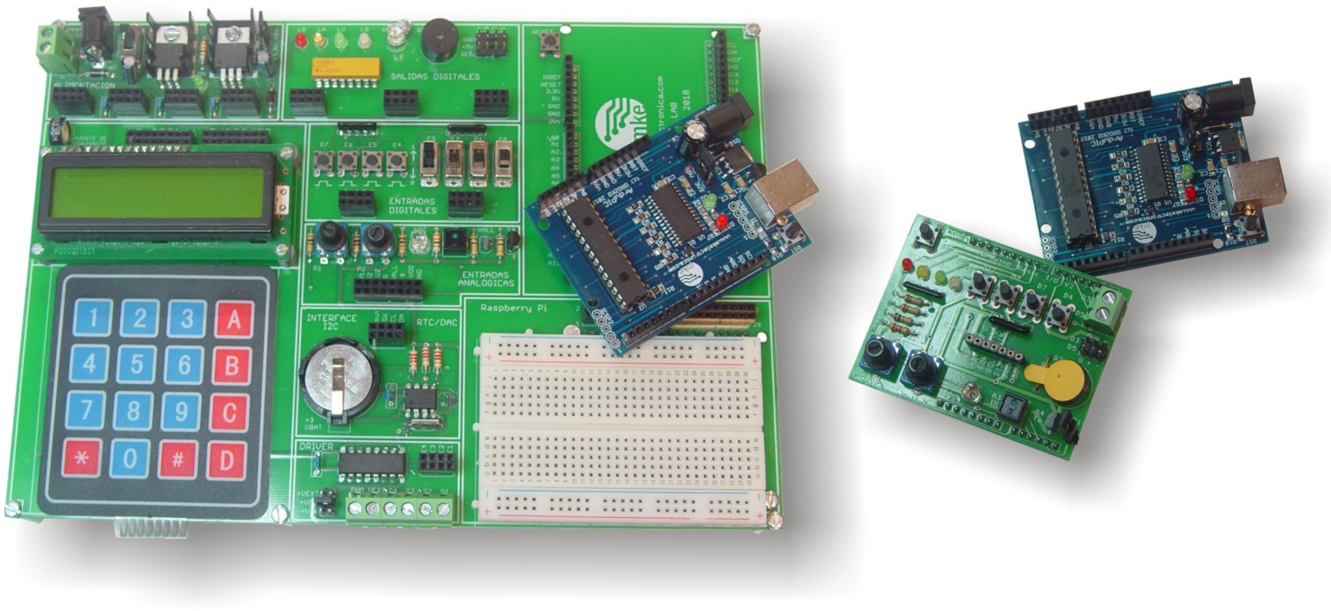


Esos ficheros ejecutables (*.HEX) los puedes obtener directamente mediante el ensamblador MPASM o el compilador de lenguaje C XC8 de Microchip. También los puedes obtener empleando otros lenguajes de otros fabricantes como el lenguaje C de CCS, el lenguaje PIC BASIC de Micro Engineering Labs, o el lenguaje gráfico FlowCode de Matrix Multimedia entre otros.

2. CARACTERISTICAS GENERALES

Las características más importantes de ArduPIC se pueden resumir en:

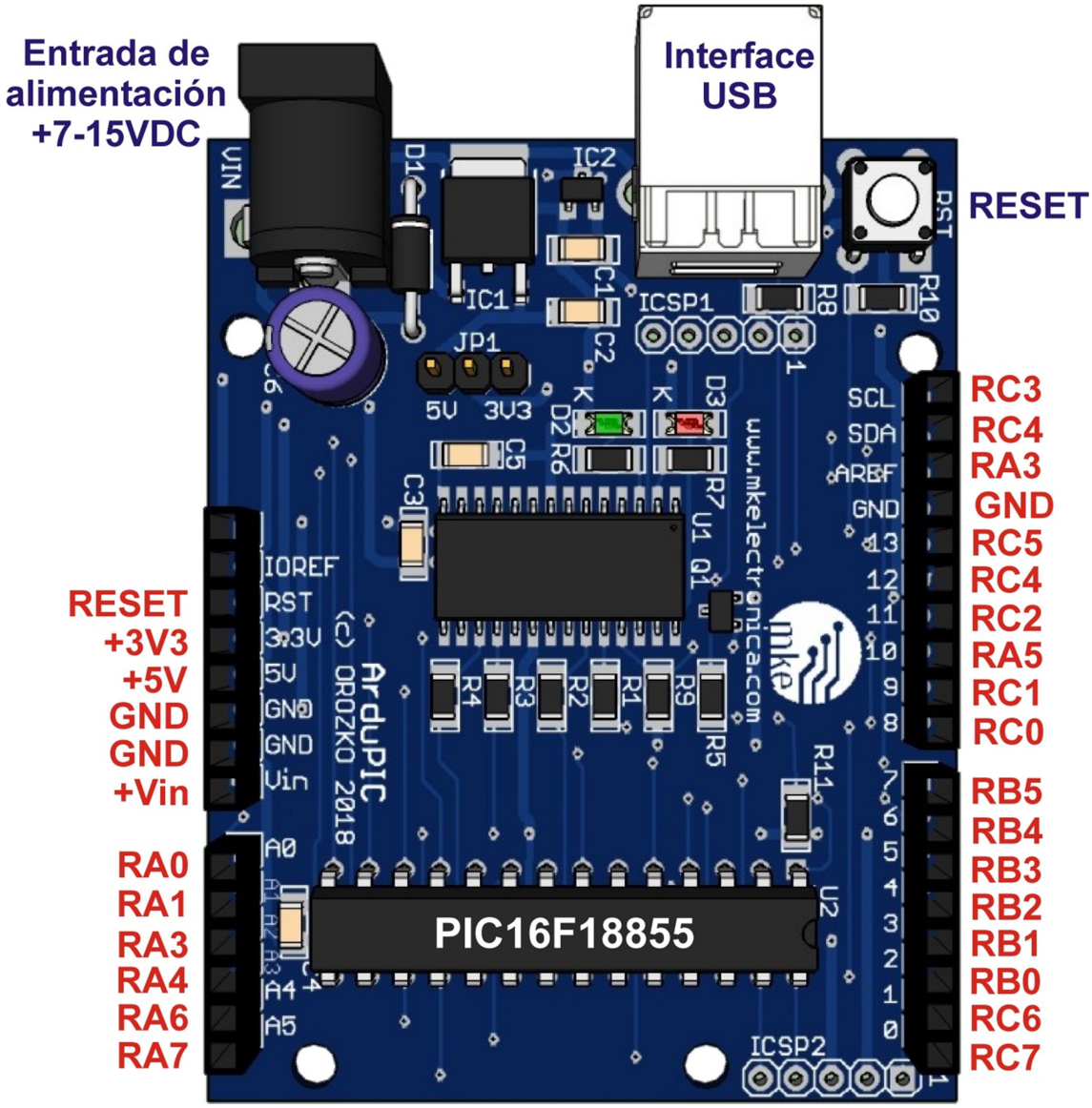
- Incluye un controlador de 8 bits PIC16F18855 de última generación
- Mediante un Jumper se puede seleccionar la tensión de trabajo entre 3.3V y 5V (por defecto).
- Interface USB con el PC mediante cables adaptador estándar de USB tipo A a USB tipo B.
- Tensión de alimentación externa de 7 a 15 VDC. Se puede aplicar desde un alimentador AC/DC de pared o desde un sistema de pilas/baterías.
- Pulsador RESET para reiniciar la comunicación entre la tarjeta ArduPIC y el PC.
- Grabación del controlador mediante la técnica "*Drag and Drop*" de arrastrar y soltar.
- Pin compatible con Arduino UNO lo que implica que podrás usar la mayor parte de shields y tarjetas de expansión disponibles.
- Por este mismo motivo ArduPIC puede trabajar con la tarjeta básica de entradas y salidas BASIC I/O y la plataforma MicroLab. Ambas de MK Electrónica.



3. CONEXIONES

Cuando se dice que una tarjeta es pin compatible con, Arduino UNO, simplemente significa que tiene la misma distribución de patillas, el mismo tipo de conectores, las mismas alimentaciones, etc… Pero las señales como tal no tienen porqué ser las mismas. El microcontrolador PIC16F18855 de ArduPIC no tiene nada que ver con el microcontrolador At328 de Arduino UNO. Tampoco se emplean las mismas herramientas de programación, ni el mismo lenguaje, ni los mismos recursos internos.

Vamos a ver la relación entre las conexiones compatibles de Arduino UNO con las señales correspondientes al microcontrolador de ArduPIC.



ARDUINO	ARDUPIC	DESCRIPCIÓN
RESET	RESET	Reinicia la comunicación USB entre ArduPIC y el PC
+3V3	+3V3	Salida de tensión de alimentación estabilizada a 3,3V / 250 mA máximo
+5V	+5V	Salida de tensión de alimentación a 5V / 500mA máximo
GND	GND	Señal de tierra de alimentación
+Vin	+Vin	Entrada de tensión externa de alimentación de +7V a 15VDC
A0-A5	RA0-RA7	Entradas analógicas
D0-D7	RC7-RC6 y RB0-RB5	Entradas/Salidas digitales
D8-D13	RC0, RC1, RA5, RC2, RC4 y RC5	Entradas/Salidas digitales
AREF	RA3	Entrada de tensión de referencia
SDA	RC4	Bus I2C, señal de datos
SCL	RC3	Bus I2C, señal de reloj

4.- EL PIC16F18855

¿Porqué hemos elegido este microcontrolador para la tarjeta ArduPIC? Pues bien, alguno había que elegir, pero este modelo en concreto nos parece de lo más completo dentro de la enorme familia de microcontroladores PIC de Microchip. Si ya estás familiarizado con sus predecesores de la subfamilia PIC16, en este modelo encontrarás nuevas y potentes prestaciones.

Toda la información técnica, abundante, completa y de primera mano para este dispositivo, la puedes encontrar en el Data Sheet que facilita Microchip. Para que te hagas una breve idea vamos a destacar sus características más relevantes.

4.1 Características del núcleo o CPU

- Arquitectura RISC optimizada para los compiladores C
- Juego reducido con 49 instrucciones

- Velocidad máxima de trabajo de 35MHz con un ciclo de instrucción de 125nS
- Capacidad de interrupción y memoria stack de 16 niveles
- Tres temporizadores de 8 bits (TMR2/4/6)
- Cuatro temporizadores de 16 bits (TMR0/1/3/5)
- Power On Reset (POR) de bajo consumo
- Power Up Timer (PWRTE) configurable
- Brown Out Reset (BOR) con recuperación rápida
- Opción de Brown Out Reset de bajo consumo (LPBOR)
- Temporizador watchdog con ventana temporal (WWDT)
- Protección de código programable

4-2 Memoria

- 8K de memoria FLASH de programa
- 1K de memoria SRAM de datos
- 256 bytes de memoria EEPROM de datos

4-3 Alimentación y modos de trabajo

- Tensión de alimentación de 2,3V hasta 5.5V
- Modo DOZE: Es la posibilidad de que el núcleo de la CPU ejecute a menor velocidad que el sistema de reloj.
- Modo IDLE: Es la posibilidad de detener la CPU mientras los periféricos continúan trabajando.
- Modo SLEEP: Todo queda detenido y el consumo se reduce a la mínima expresión.
- Desconexión de los Módulos Periféricos (PDM): Posibilidad de desconectar aquellos periféricos que no se estén usando y reducir así el consumo general.

4-4 Periféricos digitales

- Cuatro Células Lógicas Configurables (CLC) independientes que integran funciones lógicas combinacionales y secuenciales similares a las de las PLD's, PAL, GAL, etc….
- Tres Generadores de Ondas Complementarios (CWG) con control de banda muerta por flanco ascendente y descendente, puente en full H, half H y múltiples fuentes de señales.
- Cinco módulos para la captura, comparación y generación de PWM (CCP)
- Resolución PWM de 10 bits
- Oscilador Controlado numéricamente (NCO) que produce una frecuencia muy lineal desde 0 hasta 32MHz y alta resolución

- Dos temporizadores para medición de señales (SMT) de 24 bits y hasta 12 modos diferentes de adquisición.
- Sistema de chequeo CRC de 16 bits.
- Comunicación mediante EUSART (RS232, RS485 y LIN), dos módulos SPI y dos módulos I2C compatibles con SMB y PMB.
- Hasta 25 patillas de E/S con cargas pull-up, interrupción por cambio de estado, control de nivel lógico (ST y TTL), configuración en colector abierto, etc…
- Selección de Pin de los periféricos (PPS) que permite reconfigurar cómo se conectan las patillas físicas del controlador con las señales de E/S de los periféricos.
- Modulador de Señal de Datos (DSM) capaz de modular una señal portadora con datos digitales.

4-4 Periféricos analógicos

- Hasta 24 canales analógicos de entrada de 10 bits y con capacidad de cálculo.
- Dos comparadores.
- Un convertidor digital analógico (DAC) de 5 bits.
- Generador de tensión de referencia con salidas de 1.024V, 2.048V y 4.096V.

4-5 Oscilador flexible

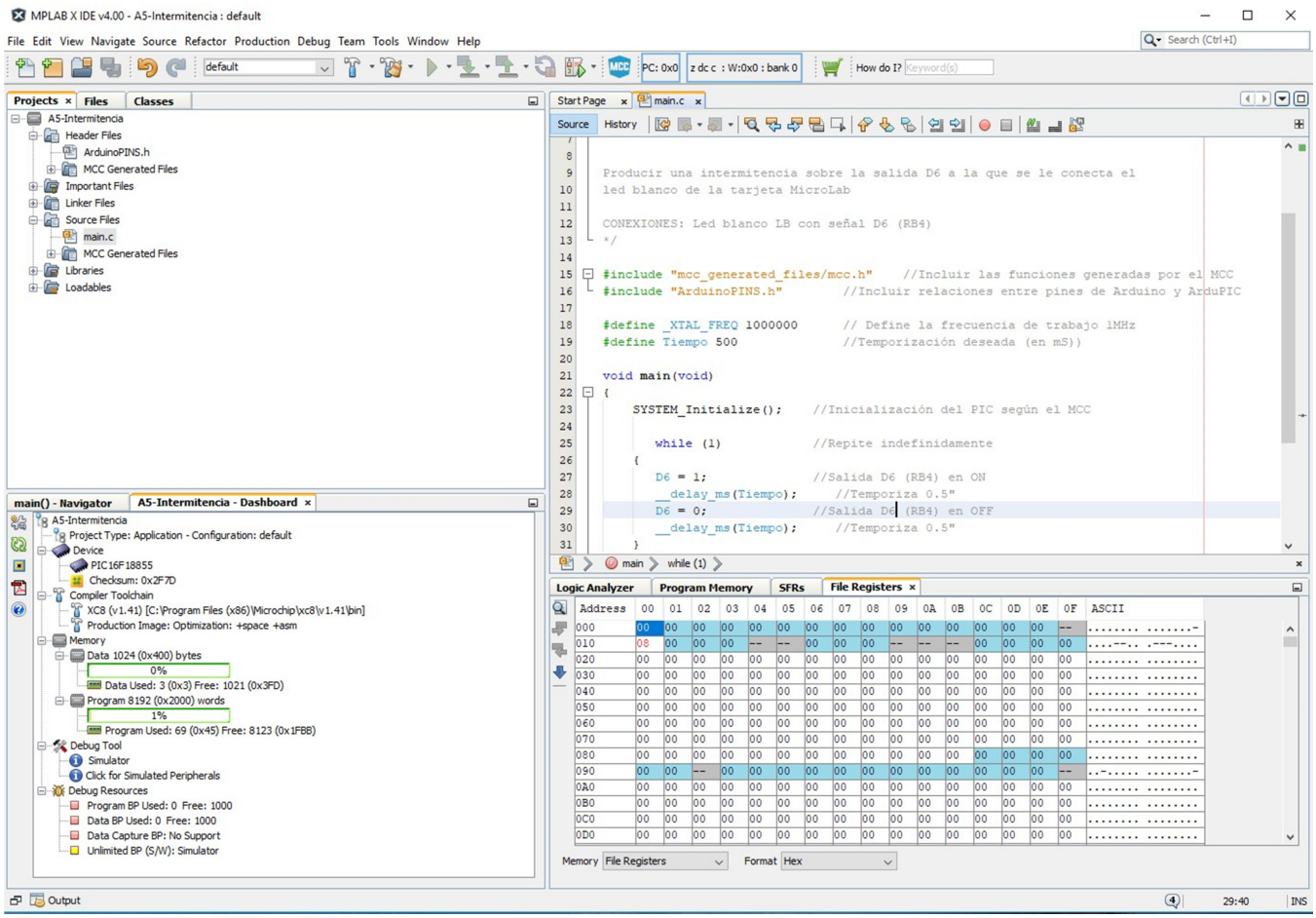
- Oscilador interno de alta precisión con rangos de frecuencia de hasta 32MHz seleccionable por software.
- PLL de x2 y x4 con fuente de reloj externa e interna.
- Oscilador interno de bajo consumo a 32KHz.
- Boque oscilador externo con tres modos de cristal/resonador de hasta 20MHz y tres modos de reloj externo de hasta 20MHz.
- Monitorización de fallo de reloj.
- Temporizador Start-Up (OST).

5.- HERRAMIENTAS SOFTWARE

Al igual que ocurre con la mayoría de los dispositivos PIC se dispone de gran cantidad de herramientas software que facilitan el diseño y puesta a punto de los programas de aplicación. Algunas son originales de Microchip y otras las facilitan distintos fabricantes. Vamos a comentar algunas de ellas.

5-1 MPLAB X IDE

Es el entorno de trabajo o IDE original de Microchip. Se trata de una potente plataforma que te permite trabajar con todos los dispositivos PIC disponibles. Es gratuita y la puedes descargar desde: http://www.microchip.com/mplab/mplab-x-ide



El MPLAB-X IDE proporciona todo lo necesario para desarrollar cualquier proyecto de principio a fin. Un pequeño resumen de lo que incluye:

- Completo gestor de proyectos.
- Potente editor de texto para que escribas tus programas fuente.
- Admite diferentes lenguajes de programación como el **Ensamblador** (incluido) o el **C** de Microchip, o de terceros fabricantes. Algunos son gratuitos y otros de pago.
- Incluye un simulador que te permite ejecutar los programas y depurarlos: ejecución paso a paso, breakpoints, tiempo real, etc…
- Permite editar/modificar los diferentes tipos de memoria del controlador: memoria FLASH de programa, RAM de datos, la EEPROM, los registros FSR, la memoria de configuración y la de identificación ID.

- Dispone de recursos como analizador lógico y generación de estímulos para la simulación.
- Puede trabajar con todas las herramientas hardware de Microchip como los depuradores y emuladores en tiempo real: PICkit-3, ICD-3, ICD-4, Real Ice, etc…
- MPLAB-X IDE es multiplataforma y se puede ejecutar bajo Windows, Mac OS y Linux.

5-2 COMPILADORES DE MICROCHIP

Bajo la denominación MPLAB XC se agrupan diferentes compiladores para todas las subfamilias de microcontroladores PIC:

- XC8: Soporta a todos los micros de 8 bits: PIC10, PIC12, PIC16 y PIC18
- XC16: Soporta todos los dispositivos de 16 bits y dsPIC
- X32/32++ Soporta todos los dispositivos de 32 bits.

Todos ellos se integran con MPLAB-X y se pueden descargar de forma gratuita y totalmente funcional: http://www.microchip.com/mplab/compilers

Si se desea se puede adquirir la versión PRO de pago de cada uno de ellos. Con esta licencia el compilador genera un código mucho más optimizado que mejora el rendimiento general de la aplicación.

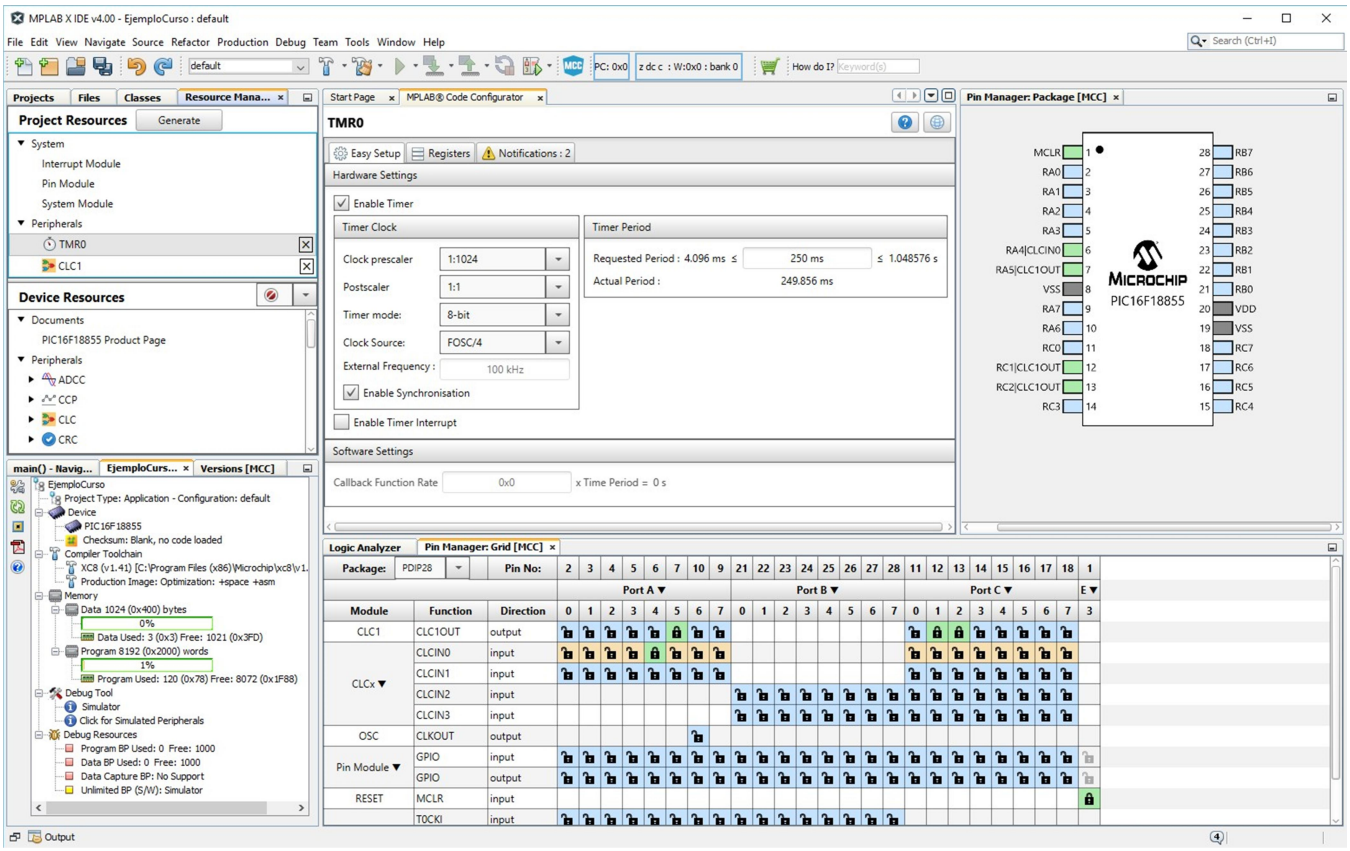
Este compilador, al igual que la mayoría, genera un fichero ejecutable con extensión *.HEX que es el que debes de grabar en la memoria del PIC16F1885 de nuestra tarjeta ArduPIC.

5-3 GENERADOR DE CÓDIGO DE CONFIGURADOR (MCC)

Se trata de una herramienta gratuita que se integra en MPLAB-X. Es un entorno gráfico que genera el código C para la configuración de los periféricos con el que normalmente comienza cualquier programa.

Emplea un interface sencillo e intuitivo con el que puedes configurar y determinar cómo deben actuar

los diferentes módulos o recursos internos del controlador en una determinada aplicación: Patillas de E/S, Timers, ADC, USART, I2C, CCP, etc… Seguidamente, al pulsar el botón "**Generate**", se obtiene automáticamente el código en lenguaje XC necesario para realizar todas las configuraciones establecidas.

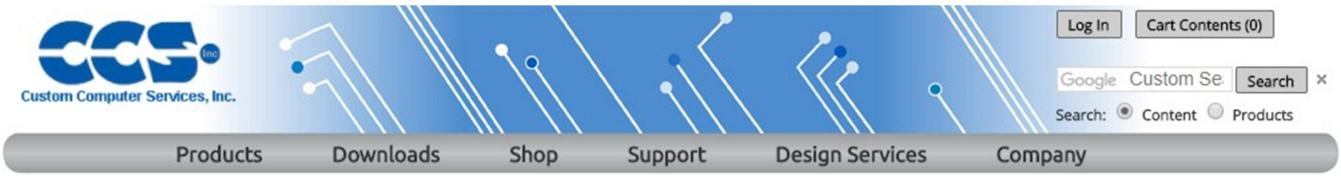


El MPLAB Code Configurator (MCC) se puede descargar desde: http://www.microchip.com/mplab/mplab-code-configurator

El PIC16F18855 incorporado en la tarjeta controladora ArduPIC puede usar estas tres potentes herramientas gratuitas de Microchip: el entorno MPLAB-X IDE, el compilador XC8 y el generador de código de configuración MCC. Si eres un entusiasta de los microcontroladores PIC, podrás disfrutar con ellas. ¡¡ Esto es lo que yo considero que es trabajar con micros en mayúsculas !!

5-4 COMPILADORES CCS

Otro conocido fabricante de compiladores C y herramientas de desarrollo es la firma Custom Computer Services, Inc (CCS) que puedes localizar en www.ccsinfo.com



Su producto más conocido es la gama de compiladores de lenguaje C para PIC:

- CCS-PCW para dispositivos PIC10/12/16 de 12 y 14 bits
- CCS-PCWH para dispositivos PIC10/12/16/18 de 12, 14 y 16 bits
- CCS-PCWHD para dispositivos PIC10/12/16/18/24/dsPIC de 12, 14, 16 y 24 bits
- CCS-PCDIDE solo para dispositivos PIC24 y dsPIC de 24 bits

Son compatibles con el entorno MPLAB-xIDE de Microchip y es posible descargar una versión gratuita y totalmente funcional durante 45 días:

http://www.ccsinfo.com/ccsfreedemo.php

Lo importante es que este, al igual que otros compiladores, produce un fichero ejecutable de extensión *.HEX que es el que garbarás en la memoria de programa del controlador PIC15F18855 de la tarjeta ArduPIC.

5-5 PIC BASIC

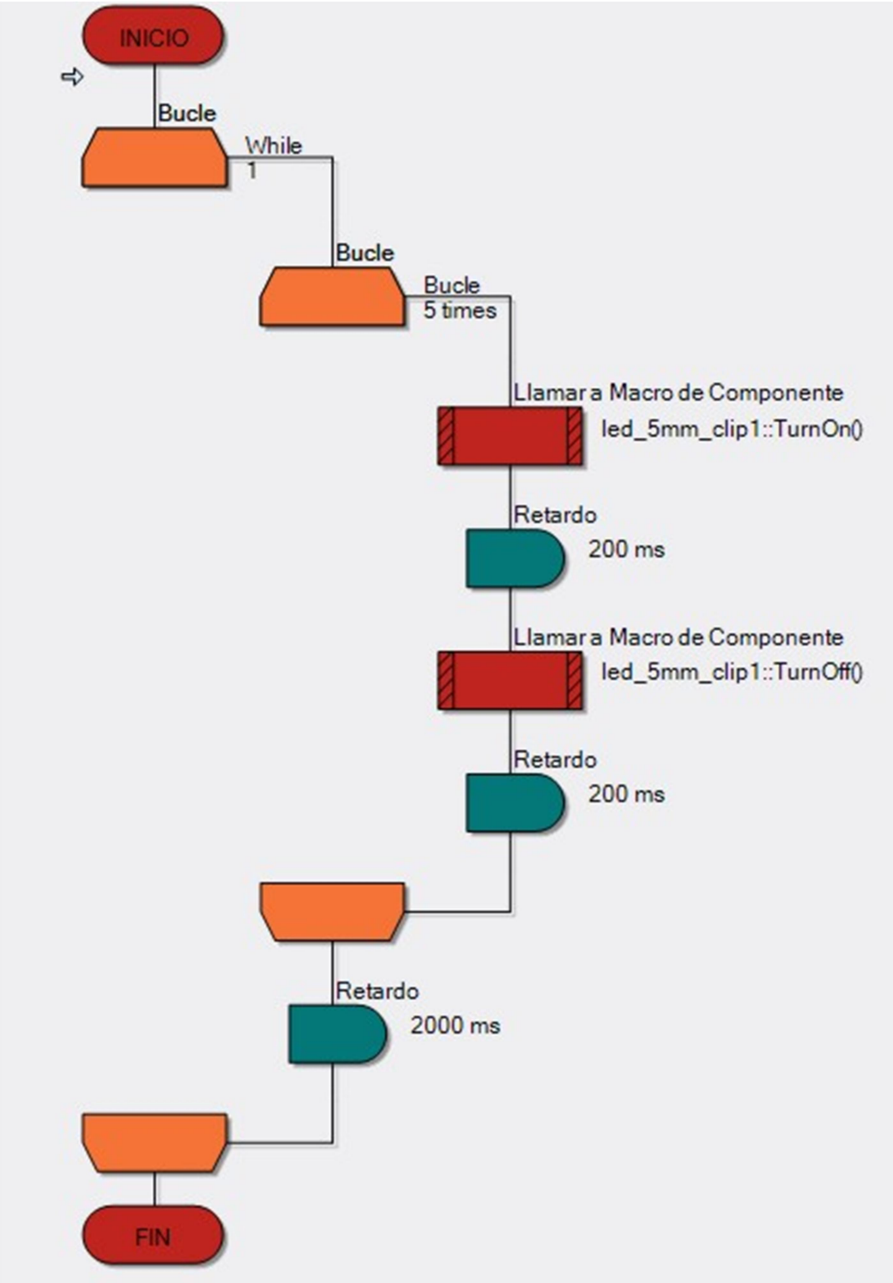
Aquí tenemos otro conocido lenguaje de programación como es el BASIC adaptado a los microcontroladores PIC. Desarrollado por la firma ME Labs, Inc puedes descargar una versión gratuita desde www.melabs.com

Su producto principal, el compilador PICBASIC PRO (PBP3), es compatible con el entorno MPLA-X IDE de Microchip.

PBP3
PICBASIC PRO™ Compiler

5-6 FLOW CODE

Otro interesante lenguaje para la programación de PIC's en general y del PIC16F18855 en particular. En esta ocasión se trata de un lenguaje gráfico que hace que la programación sea más sencilla e intuitiva. Está desarrollado y comercializado por la firma Matrix Multimedia: http://www.matrixtsl.com



El programa se diseña dibujando un diagrama de flujo cuyas piezas conforman las tareas a ejecutar. Ese diagrama se convierte a su equivalente en lenguaje C, precisamente al XC8, y de aquí al ejecutable *.HEX que grabarás en la memoria del controlador PIC16F18855 de la tarjeta ArduPIC.

FlowCode tiene además otras muchas funcionalidades: Simulación de ejecución, panel de instrumentos, creación y representación de objetos en 3D, implementación de un gran número de periféricos, etc…

Se puede descargar una versión gratuita totalmente funcional pero limitada en el tiempo.

6.- GRABACIÓN

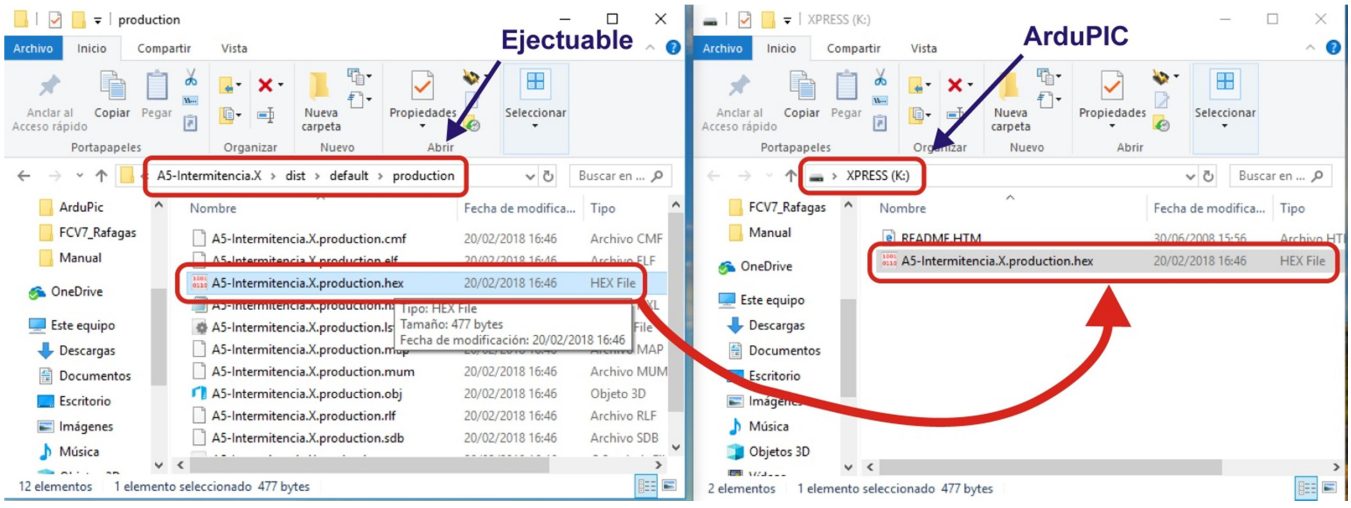
Como se ha comentado al principio de este documento, el sistema de grabación del PIC16F18855 de la tarjeta ArduPIC es conocido como "*Drag and Drop*" (arrastrar y soltar), una auténtica novedad.

Cada vez que conectas la tarjeta controladora ArduPIC a un puerto USB de tu PC, o accionas el pulsador RESET de la misma, se abre una ventana representando a una nueva unidad de almacenamiento: **XPRESS (:)**.

Por otra parte debes localizar en qué lugar o carpeta ha dejado tu compilador, el que sea que hayas utilizado, el fichero ejecutable con extensión *.HEX. En el caso de utilizar el gestor de proyectos de MPLAB-X IDE y el compilador XC8, el ejecutable se suele dejar por defecto en la carpeta:

…\nombre proyecto.X\dist\default\production

Una vez localizado el *.HEX basta con hacer click sobre él con el botón izquierdo y arrastrarlo hacia la ventana con la nueva unidad que representa a la tarjeta ArduPIC.

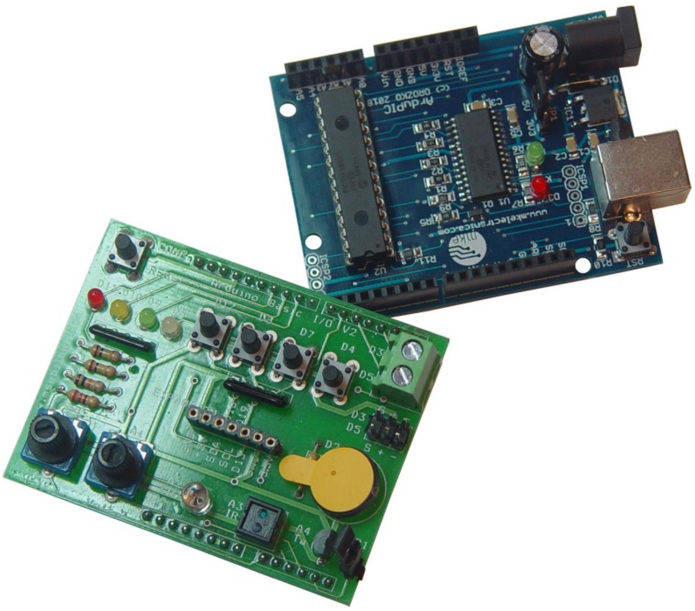


NOTA. Piensa que la ventana con la nueva unidad de almacenamiento que representa a la controladora ArduPIC (XPRESS) no es realmente una unidad de disco. ¡¡ En la memoria física del PIC16F18855 de que consta esta tarjeta solo cabe un programa cada vez !! Si tratas de arrastrar a ella varios ficheros ejecutables *.HEX distintos, lo más probable es que te aparezca una notificación y tengas que borrarlos de la ventana.

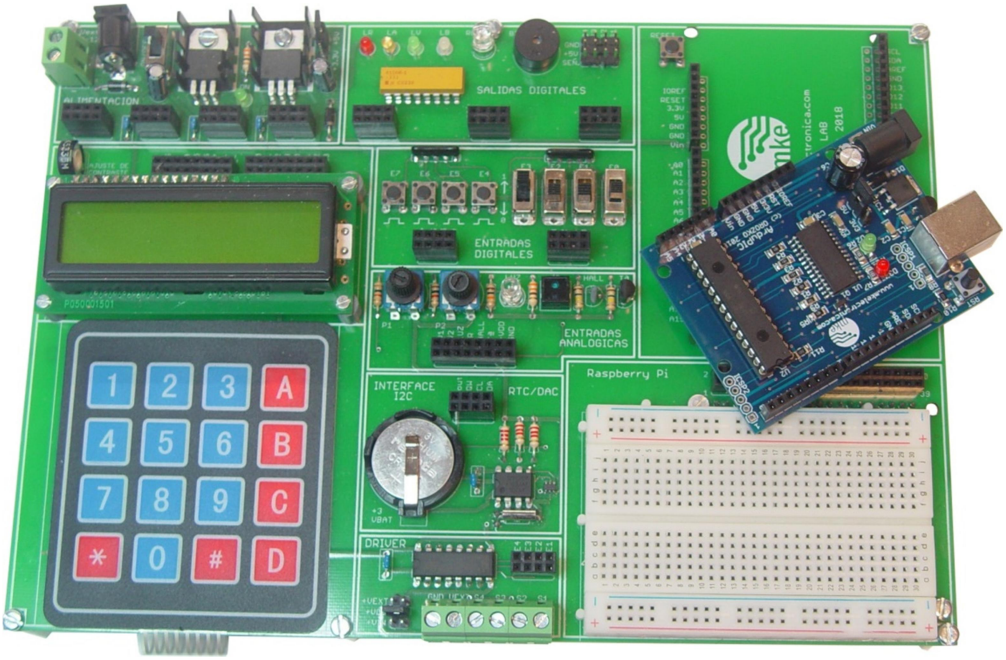
7.- SHIELDS COMPATIBLES

ArduPIC es una tarjeta pin compatible con Arduino UNO. Esto quiere decir que emplea los mismos conectores, con el mismo número de señales, la misma distribución de las tensiones de alimentación, las mismas dimensiones mecánicas, etc…

Aunque no se puede garantizar al 100%, la mayor parte de shields diseñados para Arduino UNO deben servir también para la tarjeta controladora ArduPIC. ¡¡ Este es otro de los atractivos de esta tarjeta !! Puedes usar con ella la mayor parte de dispositivos que usabas con Arduino.



Como es natural, en MK Electrónica hemos probado la controladora ArduPIC con nuestro shield básico de entradas y salidas BASIC I/O. Entre ambos, dispones de los medios suficientes para empezar a programar el PIC16F18855 y realizar tus primeros proyectos.



Si tus inquietudes, tus ansias de aprender o tus proyectos son más ambiciosos, ArduPIC es compatible con la plataforma Micro LAb, que dispone de un mayor número de periféricos y recursos para cubrir tus expectativas.

Ahora bien, no debes olvidar que el controlador PIC16F18855 de la tarjeta ArduPIC no es el mismo (ni parecido) al controlador AT328 de la tarjeta Arduino UNO. Cuando hablamos de compatibilidad lo hacemos a nivel de hardware, nunca a nivel de software.

En las siguientes tablas de conexiones se muestra la relación que hay entre las señales originales de Arduino UNO y las señales que proporciona el PIC16F18855. Imprímelas y tenlas siempre a mano.

PIC	ARDUINO
NC	NC
+5V	IOREF
RESET	RESET
+3V3	+3V3
+5V	+5V
GND	GND
GND	GND
+Vin	+Vin

PIC	ARDUINO
RA0	A0
RA1	A1
RA2	A2
RA4	A3
RA6	A4
RA7	A5

ARDUINO	PIC
SCL	RC3
SDA	RC4
AREF	RA3
GND	GND
D13	RC5
D12	RC4
D11	RC2
D10	RA5
D9	RC1
D8	RC0

ARDUINO	PIC
D7	RB5
D6	RB4
D5	RB3
D4	RB2
D3	RB1
D2	RB0
D1/TX	RC6
D0/RX	RC7

NOTA. Un de detalle muy importante del PIC16F18855 es su capacidad para remapear o reubicar sus patillas de E/S. Esto implica que si por ejemplo vas a usar el EUSART para realizar una comunicación serie, las patillas que usa el PIC por defecto para la transmisión (Tx) y la recepción (Rx) son RC6 y RC7. Ahora bien, estas patillas puedes reubicarlas por software y hacer que, por ejemplo, tanto Tx como Rx puedan ser cualquier patilla de la puerta B (RB0-RB7) o de la puerta C (RC0-RC7)

8.- EJEMPLOS

Vamos a acabar este documento presentando una serie de sencillos ejemplos resueltos con la controladora ArduPIC y su microcontrolador PIC16F18855. Como periféricos vamos a emplear los de nuestra tarjeta o shield BASIC I/O.

Insistimos en que son simples ejemplos y que este documento no pretende enseñar a programar PIC's ni a emplear las diferentes herramientas software disponibles. Esta tarea te corresponde a ti y para ello deberás contar con la información técnica que facilita Microchip sobre el PIC16F18855, el MPLAB-X y el compilador XC8, o bien otros fabricantes como CCS y sus compiladores de C, Micro Engineering Labs y su compilador PIC BASIC, Matrix y su lenguaje gráfico FlowCode, o cualquiera de los muchos que existen.

En nuestro caso todos los ejemplos se han resuelto mediante el entorno MPLAB-X IDE, la versión gratuita del compilador XC8 y el generador de código de configuración MCC.

8-1 A1-Encendido de un led-Función SI

Es el más sencillo que se nos ocurre y, por ser el primero, lo explicaremos con un poco más de detalle. Cada vez que se acciona el pulsador D4 conectado con RB2 del PIC, se activa el led blanco D6 conectado con la patilla RB4. Consulta las tablas de conexiones de la página anterior.

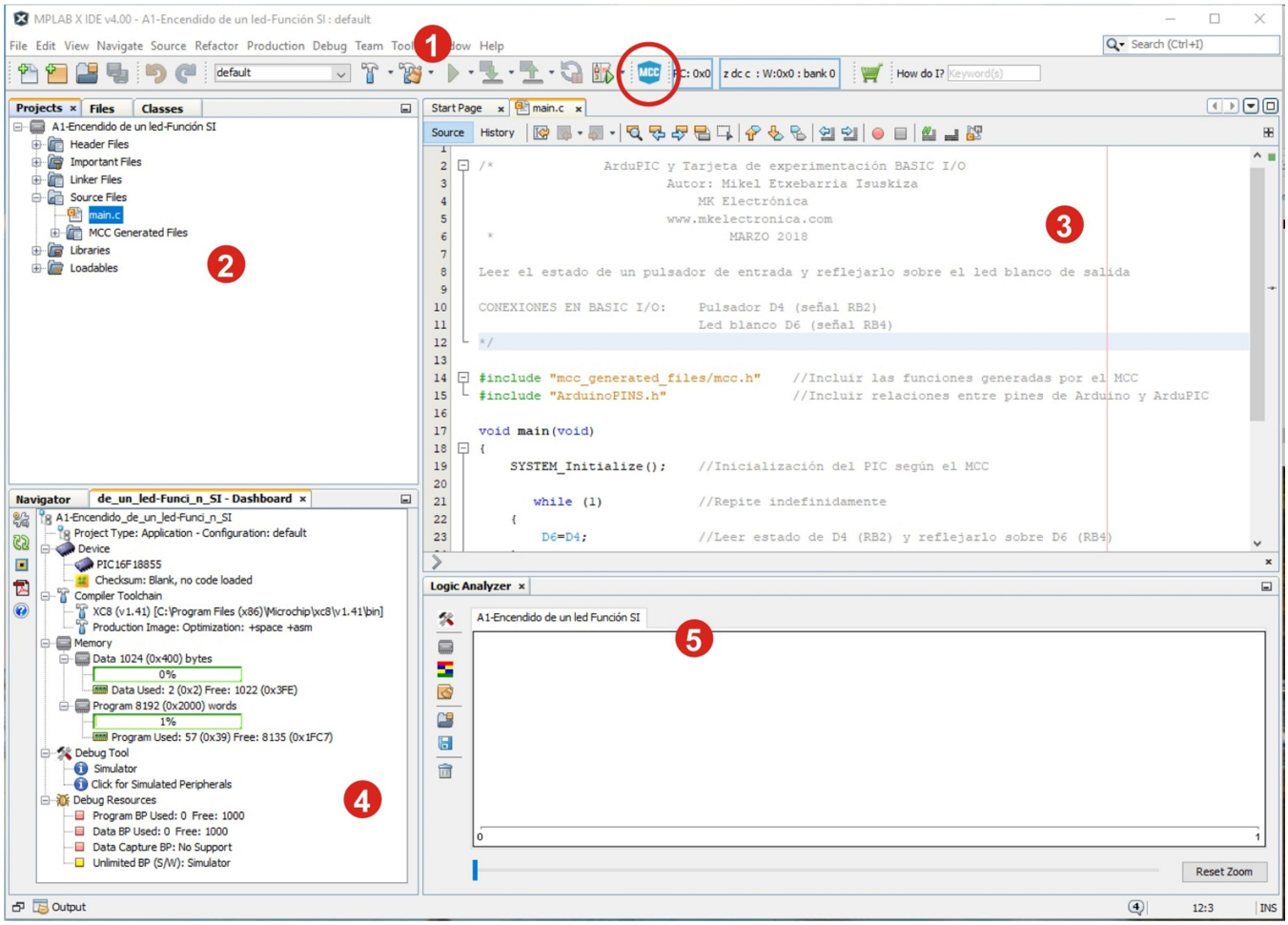
8-1-1 Abrir la aplicación MPLAB-X IDE

Suponiendo que ya tienes instalado el MPLAB-X IDE, junto con el compilador XC8 y el generador de código de configuración MCC, ejecuta la aplicación y abre este primer proyecto: "*A1-Encendido de un led-Función SI.X*"

Abre y coloca las ventanas como mejor te parezca. La imagen te puede sugerir la disposición de ventanas con las que yo me siento más cómodo: En la parte superior tenemos el menú general de opciones y los botones más usados (1). En la ventana de arriba a la izquierda se encuentra el explorador de proyectos, archivos y clases (2). A la derecha me gusta tener la ventana(s) con los programas fuente, inlcudes, cabeceras, etc… (3)

Abajo, en la ventana de la izquierda hay un resumen del proyecto actual: tipo de controlador, compilador, consumo de memoria, herramienta de depuración, etc… (4) Abajo a la derecha se encuentra una ventana donde se van presentando mensajes de interface

con el usuario, contenidos de memoria, resultados de la ejecución, etc…, en función del contexto en que te encuentres (5).

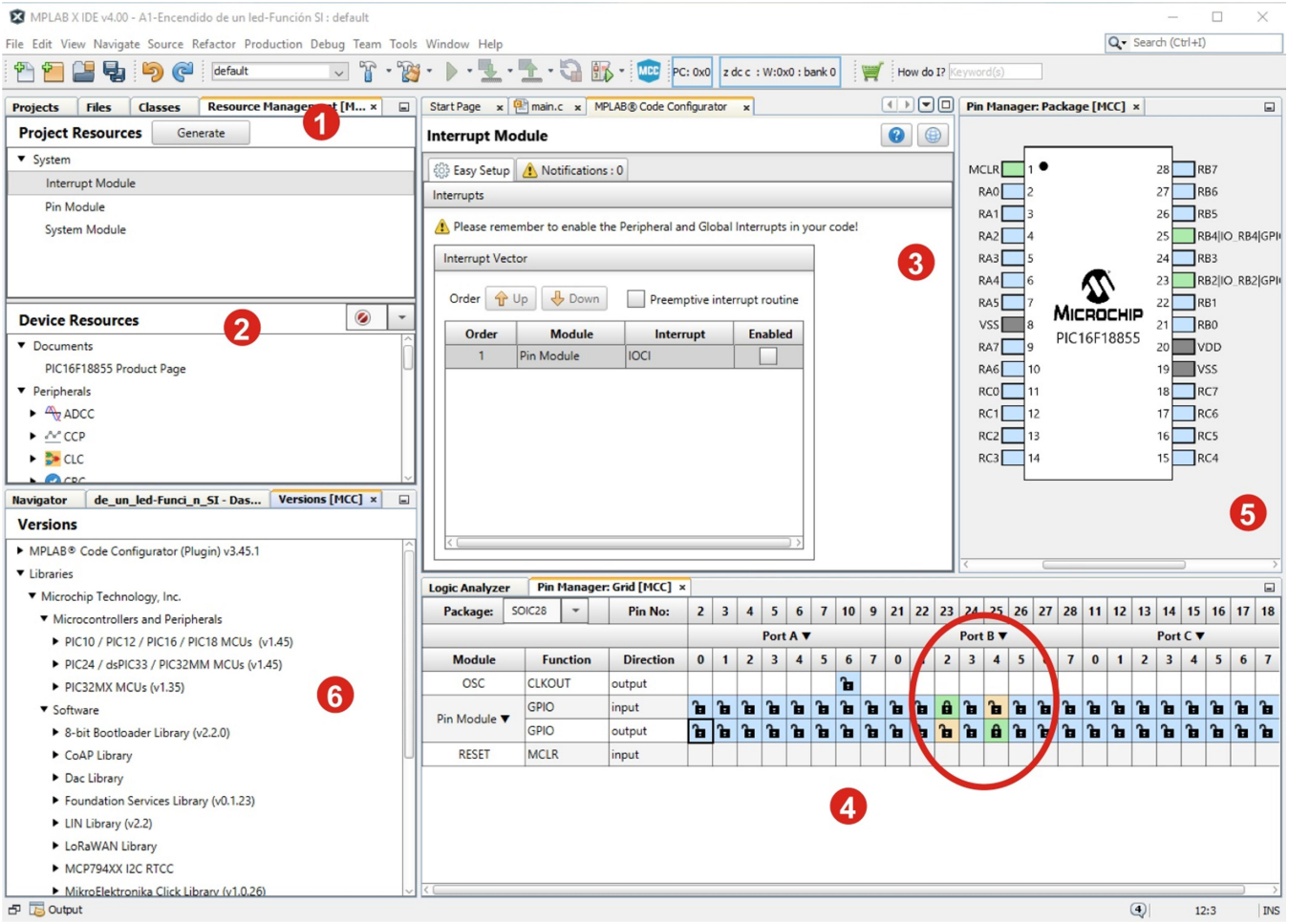


Ahora pulsa el botón MCC (círculo rojo) para activar el MPLAB-X Code Configurator (Generador de código de configuración de MPLAB-X). Con ello podrás configurar de forma gráfica e intuitiva todos los recursos del controlador: las patillas de E/S, los Timers, los convertidores ADC, las señales PWM y mucho más.

8-1-2 El Generador de código MCC

Al activar esta aplicación instalada dentro del MPLAB-X, aparecen nuevas ventanas y/o pestañas dentro de las que ya había:

1.- Resource Management: Aquí se colocan los diferentes periféricos o recursos internos del controlador y que vas a emplear en el proyecto actual. Por defecto siempre hay tres: **Interrupt Module**, **Pin Module** y **System Module**. Con el botón **Generate** se obtiene el código necesario para configurar todos los recursos colocados en esta ventana.



2.- Device Resources. En esta ventana tienes todos los periféricos o recursos internos disponibles según el controlador que hayas elegido usar en tu proyecto. ArduPIC emplea el PIC16F18855. Haciendo doble click sobre cualquiera de ellos pasan a la ventana 1 donde quedan así incluidos en el proyecto.

3.- Interrupt Module. Aquí puedes configurar individualmente todos y cada uno de los recursos o periféricos que forman parte de tu proyecto y que has colocado en la ventana 1. En el ejemplo actual se va a configurar el módulo de interrupciones si es que se van a usar (Interrupt Module) y también podrás configurar, cuando los seleccionas, el de pines (Pin Module) y el del sistema (System Módule). Enseguida lo vas a ver.

4.- Pin Manager: Grid. Aquí es donde puedes configurar, de forma muy sencilla e intuitiva, si las patillas de E/S de los diferentes puertos actuarán como entrada o salida. Observa el círculo. La patilla RB2 se configura (candado cerrado) como entrada y RB4 como salida. Precisamente estas patillas son las que se conectan con el pulsador D4 y el led blanco D6 respectivamente de la tarjeta BASIC I/O, según determina el ejemplo en cuestión.

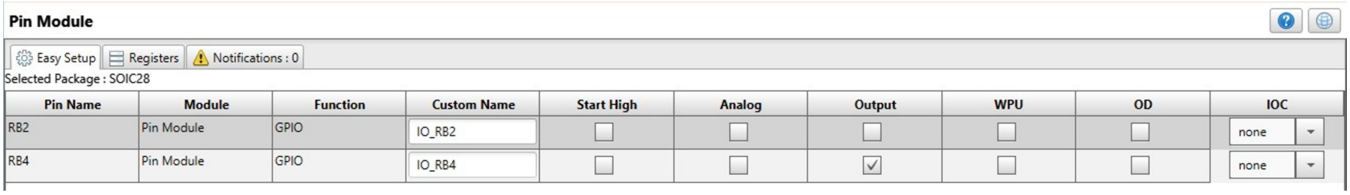
5.- Pin Manager: Package. Es una representación gráfica del encapsulado del controlador PIC16F886 y de las patillas de E/S que acabamos de configurar.

6.- Versions: Se trata de una pestaña puramente informativa que nos muestra las diferentes versiones de las herramientas software y librerías empleadas.

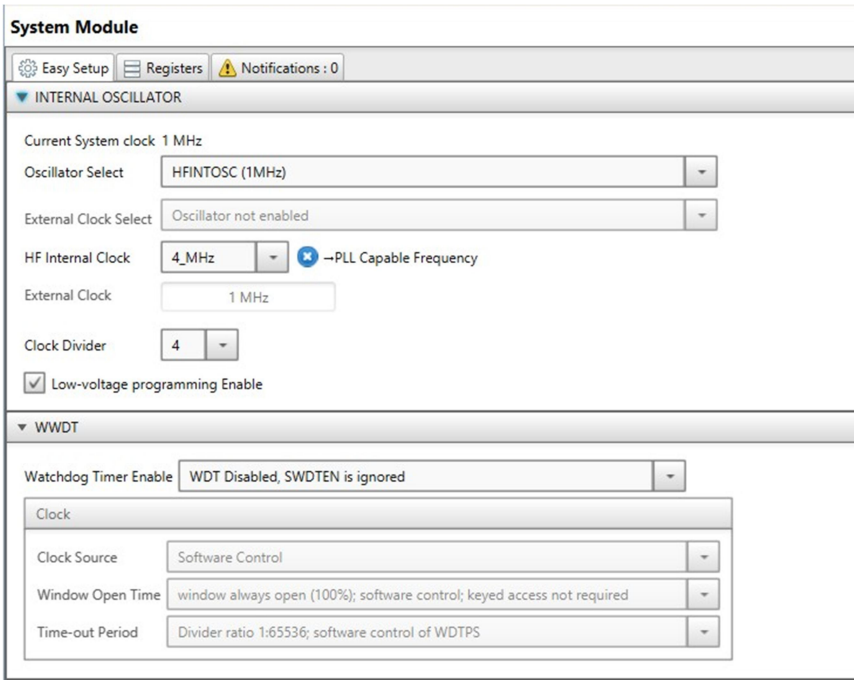
8-1-3 El programa fuente y el ejecutable

De todas las ventanas y pestañas disponibles, vamos a resumir las más importantes empleadas para generar nuestro programa fuente y el ejecutable (*.HEX) equivalente.

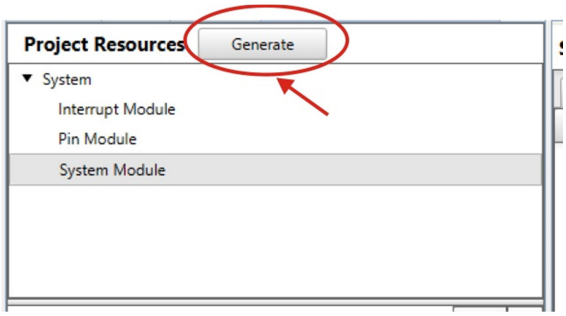
1.- Pin Module. Permite detallar la configuración de las patillas de E/S que habías configurado en el **Pin Manager: Grid** del punto 4 del apartado anterior (RB2 y RB4): nivel lógico inicial, si es una entrada analógica, si es salida, activación de Pull Up (WPU), si es en colector abierto y si se activa o no la interrupción por cambio de estado (IOC).

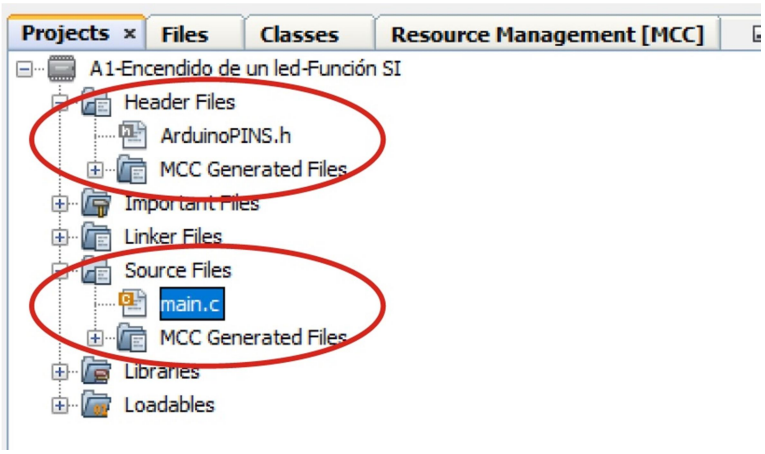


2.- System Module: Básicamente permite configurar las múltiples opciones del módulo oscilador del sistema, así como del watchdog (WDT). En nuestros ejemplos vamos a emplear un oscilador interno de 4MHz con un divisor de 4, por lo que la frecuencia de trabajo es de 1MHz y el ciclo de instrucción de 1µS. El WDT lo dejamos desconectado.



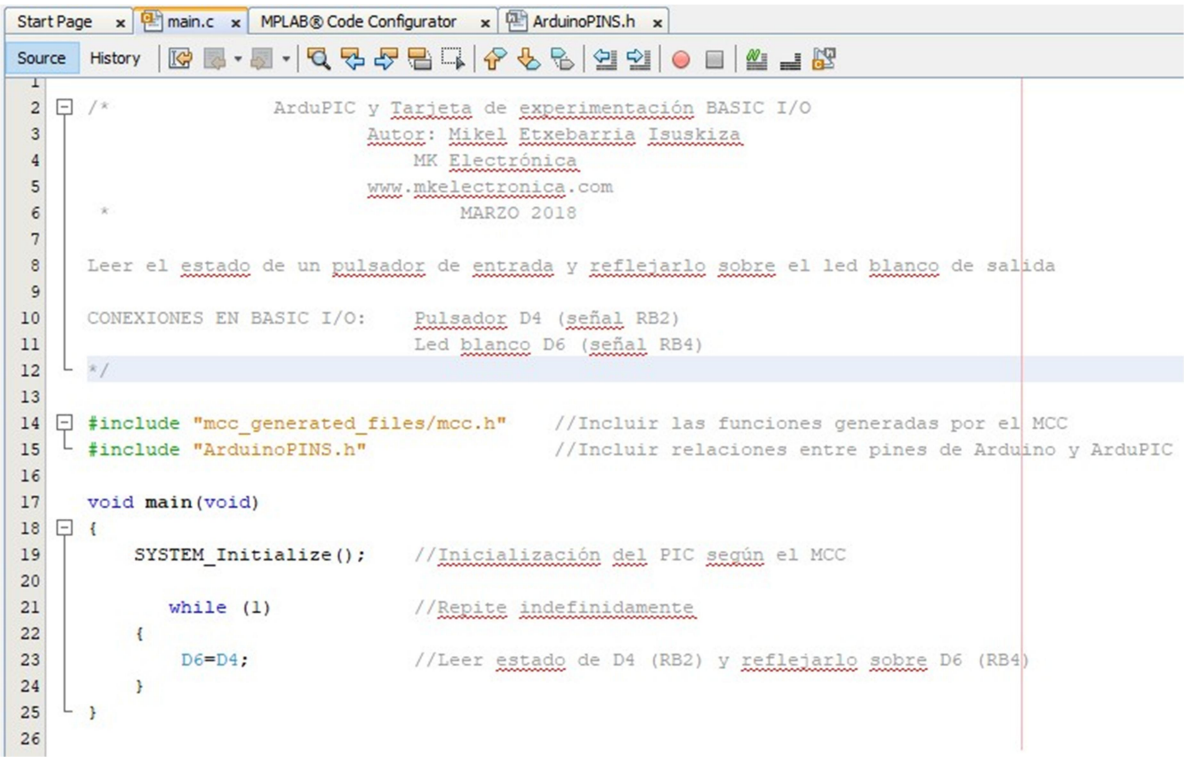
3.- Generar el código: Al pulsar el botón "**Generate**" se obtiene el código fuente necesario para configurar todos y cada uno de los módulos anteriores según las preferencias que hemos establecido. Como es lógico el código generado es válido para los compiladores de Microchip.



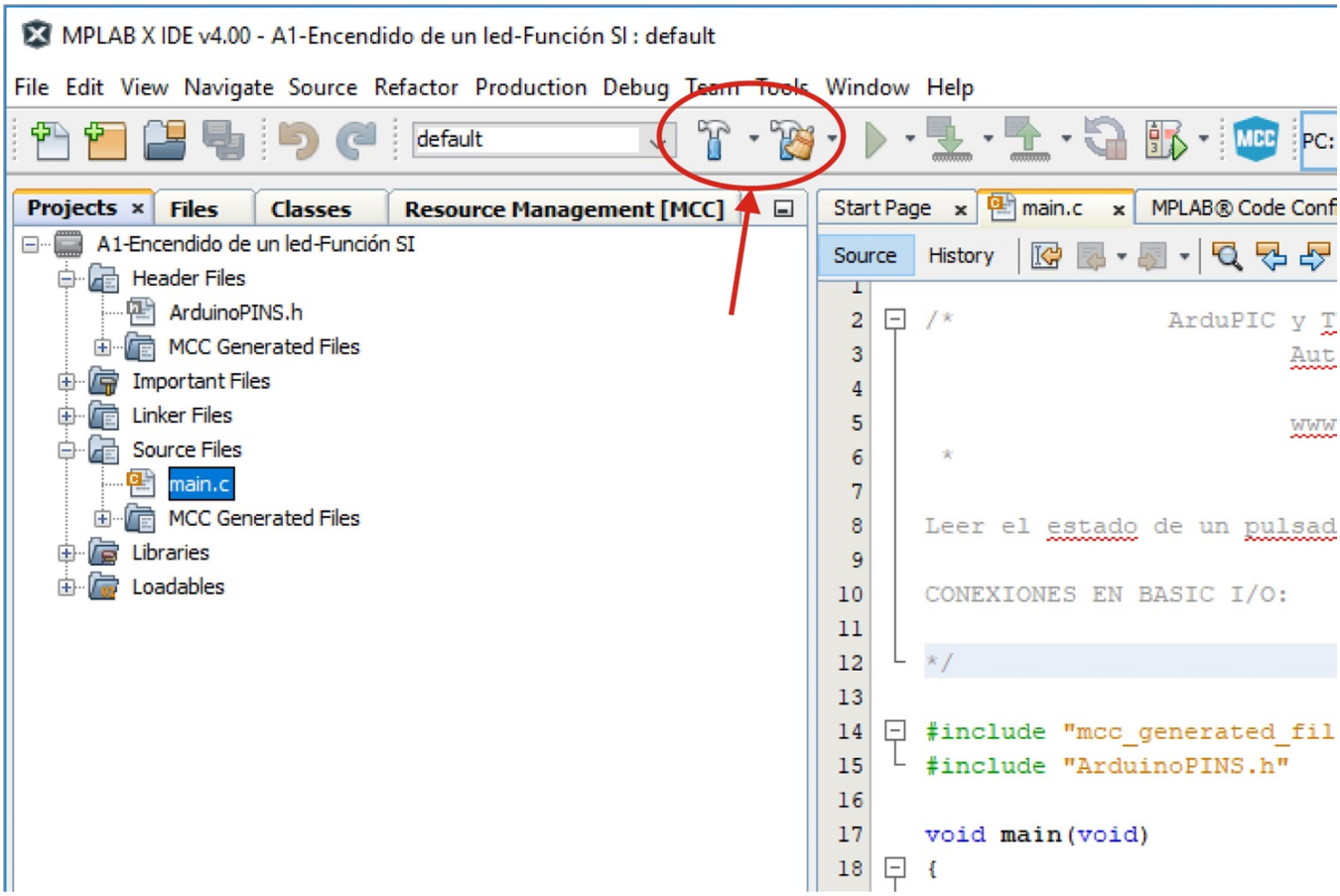


4.- Exploración: Es muy ilustrativo e interesante que explores y analices los diferentes ficheros que se han generado. Especialmente importantes son los de cabecera o Headers Files: **ArduinoPINS.h** y **MCC Generated Files**, y los ficheros fuente o Source Files: **main.c** y **MCC Generated Files**. Todos ellos los puedes abrir y modificar mediante el editor de textos del MPLAB-X.

5.- **El main.c**: Es el programa fuente principal. Si lo abres con el editor podrás ver cómo se incluye el código producido por el generador de código de configuración MCC, así como el código propio del ejemplo.



6.- Compilación: Es el proceso por el cual el programa fuente escrito, en nuestro caso, en el lenguaje C de alto nivel XC8, se convierte en código máquina o binario que se grabará en la memoria del controlador. Es muy sencillo, basta con pulsar uno de estos dos botones.



8-1-4 Grabación

Es el momento de grabar el fichero ejecutable (*.HEX) sobre la memoria del controlador PIC16F18855 disponible en nuestra tarjeta ArduPIC. Recordemos que MPLAB-X IDE y el compilador XC8 deja el ejecutable obtenido tras la compilación en una carpeta por defecto:

.../nombre del proyecto.X/dist/default/production

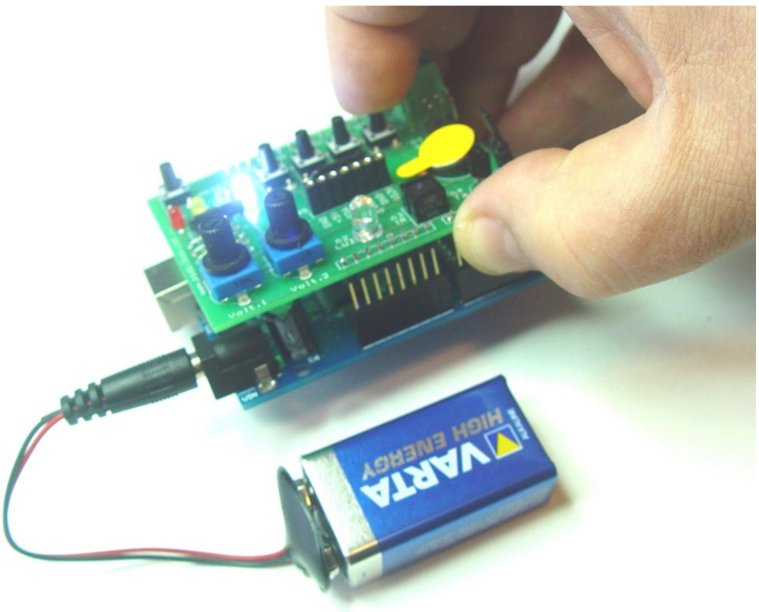
...que, en este caso sería:

.../Encendido de un led Funcion SI.X/dist/default/production

Basta con arrastrar el fichero y soltarlo sobre la unidad XPRESS(:) que representa a nuestra tarjeta ArduPIC.

8-1-5 Comprobación

Insertada la tarjeta de periféricos BASIC I/O sobre nuestra controladora ArduPIC, comprobamos que el programa recién grabado funciona correctamente. Al pulsar D4 se enciende el led blanco D6.



También puedes comprobar que el programa se ejecuta independientemente del ordenador. Desconécta ArduPIC del puerto USB y conecta una alimentación externa a base de pilas, baterías o un alimentador de AC/DC.

8-2 A2-Encendido invertido-Función NO

Es un ejemplo muy parecido al anterior. En esta ocasión cada vez que se activa el pulsador D4 el led blanco D6 se apaga. Es decir, funciona justo al contrario que el ejemplo anterior. Sigue los mismos pasos que hicimos con el ejemplo anterior.

8-3 A3- Funciones lógicas

También es muy parecido a los anteriores. Se emplean dos pulsadores de la tarjeta BASIC I/O, D4 y D7, como señales de entrada, y el led blanco D6 como salida. Recordemos que el pulsador D4 está conectado con la patilla RB2, D7 con RB5 y el led blanco D6 con la patilla RB4 del PIC. Consulta las tablas de conexiones del apartado 7

El mismo ejemplo trata de emular las funciones lógicas AND, OR, NAND, NOR, XOR y XNOR usadas en la electrónica digital.

Modificando **#define** en el programa fuente **main.c** puedes seleccionar la función lógica que vas a emular. Recuerda que cada vez que hagas una modificación debes compilar nuevamente todo el programa.

```
#define AND     //Indicar la función lógica deseada: AND, OR, NAND, NOR, XOR,
                //XNOR
```

8-4 A4-Alarma Sencilla

En este ejemplo se pretende emular una sencilla alarma. Los cuatro pulsadores D4, D7, D8 y D12 de la tarjeta BASIC I/O actúan como sensores. Ya sabes que están conectados con las señales RB2, RB5, RC0 y RC4 del PIC respectivamente.

Cuando se acciona cualquiera de ellos se activa el led rojo D11, conectado con la señal RC2 a modo de alarma. En realidad el programa ejecuta una función lógica OR de cuatro entradas.

No pierdas de vista cómo se hace la configuración de las patillas de E/S con ayuda del **Pin Manager: Grid** del generador de código de configuración MCC.

8-5 A5-Intermitencia

Otro ejemplo muy sencillo. En esta ocasión se trata de producir una intermitencia sobre el led D6 de la tarjeta BASIC I/O conectado con la patilla RB4 del PIC.

8-6 A6- Juego de luces

Partiendo del ejemplo anterior se trata en esta ocasión de realizar un juego de luces por el cual los leds blanco (D6), verde (D9), ámbar (D10) y rojo (D11) de BSIC I/O, conectados con las patillas RB4, RC1, RA5 y RC2 del PIC respectivamente, realizan una intermitencia secuencial que produce un efecto de luces en movimiento.

Variando la constante **Tiempo** puedes aumentar o disminuir la velocidad del movimiento, por ejemplo:

```
#define Tiempo 1000
```

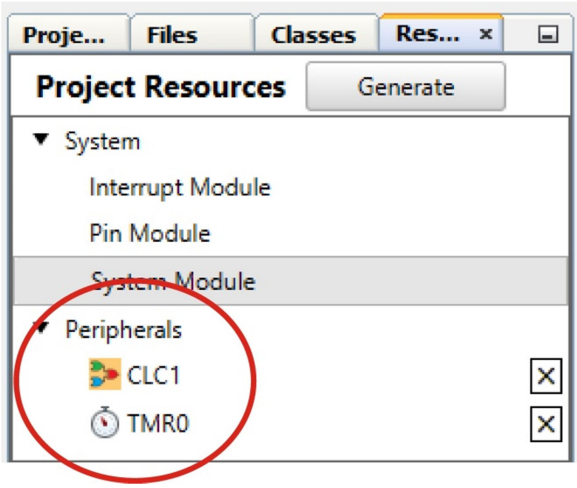
8-7 A7 TMR0:Blink

Otro ejemplo que consiste en producir una intermitencia sobre los leds rojo (D11) y verde (D9) de la tarjeta BASIC I/O, conectados con las patillas RC2 y RC1 del PIC.

¿Qué hay de nuevo en este ejemplo? Pues que vamos a emplear una serie de recursos que te darán una idea (eso espero) de la potencia del PIC16F18855 de nuestra tarjeta ArduPIC así como de la facilidad de la herramienta MCC de MPLAB-X para generar automáticamente código de configuración. Presta atención…

8-7-1 Project Resources

Activado el MCC colocamos en esta ventana los periféricos que van a formar parte de este proyecto: el módulo TMR0 y el módulo CLC1 con sus células lógicas configurables. Basta con hacer doble click sobre cada uno de ellos desde la ventana **Device Resources**.

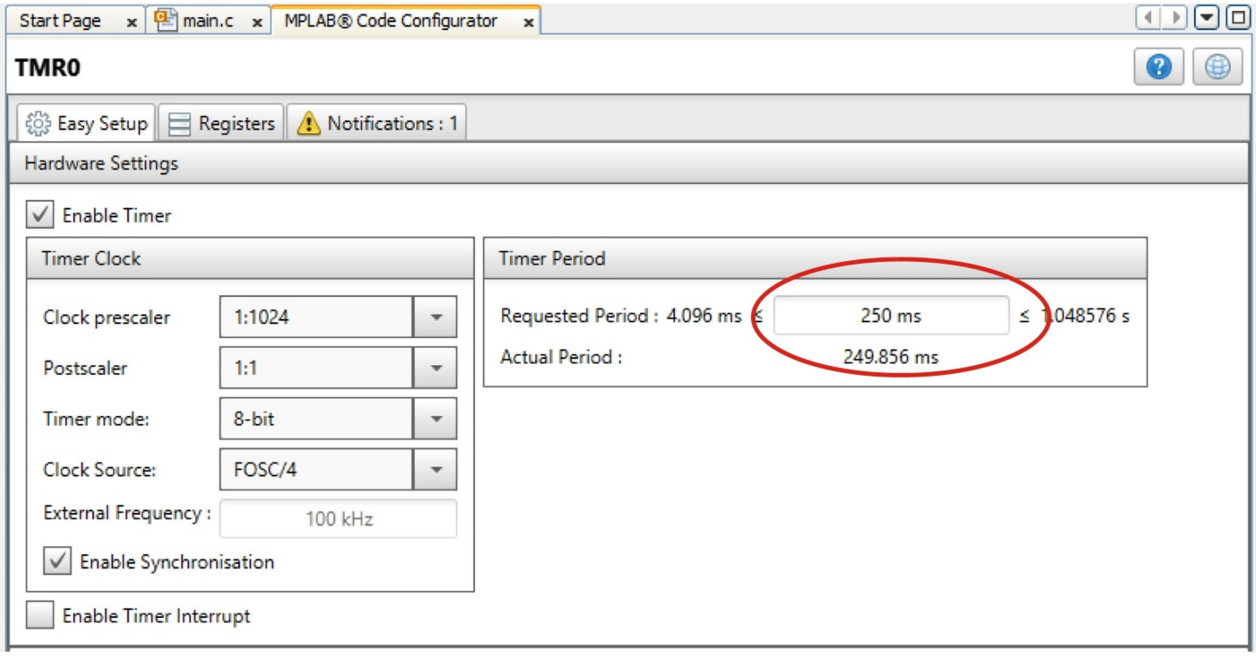


Además de estos módulos, ya sabes que por defecto también están los módulos de interrupción, de pines y del sistema. Los vamos a dejar como están.

8-7-2 Configuración del TMR0

Se selecciona desde la ventana **Project Resources**. En la ventana central del MPLAB-X IDE puedes configurar todas las prestaciones de este módulo: el preescaler, el postcaler, el modo de trabajo, la fuente de reloj y el periodo. Debes consultar en el Data Sheet del PIC16F18855 todas las funcionalidades de este módulo TMR0.

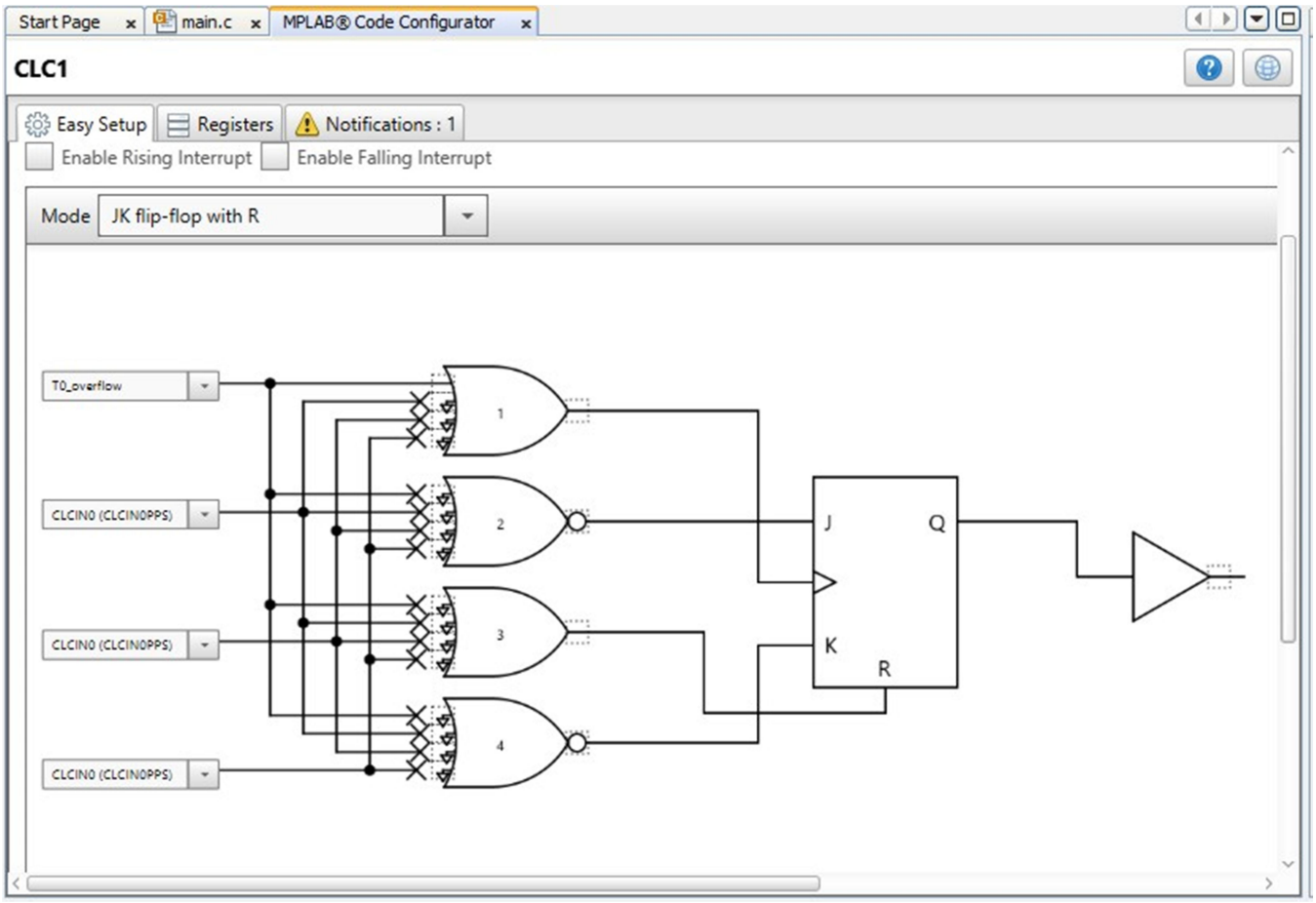
En la figura puedes ver la configuración que hemos empleado en este ejemplo para producir un periodo o desbordamiento cada 250 mS. Cuando pruebes el ejemplo puedes probar a cambiar cualquier de estos valores.



8-7-3 Configuración del módulo CLC1

Lo seleccionas también desde la ventana **Project Resources**. Este módulo consiste en una Célula Lógica Configurable (CLC) similar a las que tienen los dispositivos lógicos programables como las PLD's, GAL, PAL, etc… Nuestro PIC tiene cuatro de estas células totalmente independientes (CLC1-CLC4). ¡¡ Una auténtica y maravillosa novedad !!

Una vez seleccionas una célula, la CLC1, puedes proceder a configurar todas sus prestaciones: el modo de trabajo, las señales de entrada, polaridad, conexiones y salida.



En el ejemplo hemos configurado:

- Modo Flip-Flop JK con Reset
- Entrada desde el TMR0 para la puerta lógica 1 que va a parar a la señal de reloj del flip-flop JK
- Las entradas JK del flip-flop están a nivel "1" permanentemente. Su salida Q cambia de estado con cada señal de reloj recibida.

- La entrada RESET del flip-flop está permanentemente a nivel "0". No tiene efecto alguno.
- La salida Q del flip-flop, que representa a la salida del módulo CLC1 no está invertida.

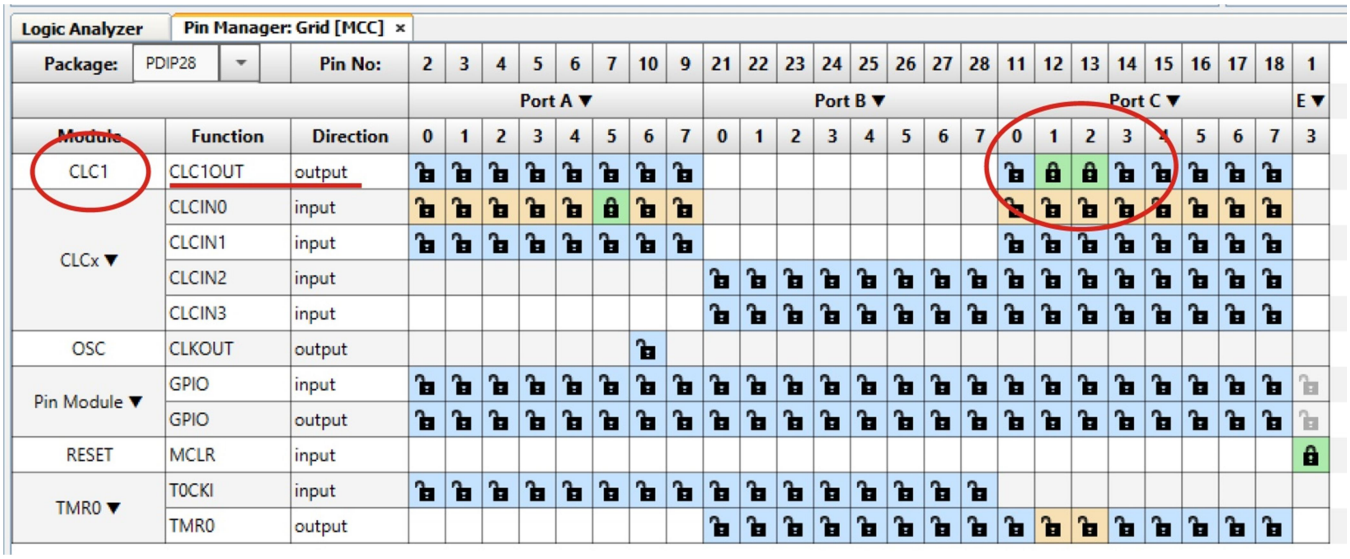
La conclusión de esta configuración del módulo CLC1 es la siguiente: Cada vez que el TMR0 desborda se produce una señal de reloj en el flip-flop. Este cambia de estado (J=K= nivel "1") y la salida Q pasa de "0" a "1" o viceversa.

Te recuerdo que el TMR0 lo habíamos configurado para que desborde cada 250 mS.

8-7-4 Configuración de las E/S

Esto lo puedes hacer desde el **Pin Manager: Grid**. Otra de las maravillosas novedades que aporta el PIC16F18855 de nuestra controladora ArduPIC es la posibilidad de re mapear gran parte de las patillas de E/S asociadas a los diferentes módulos.

En este ejemplo hemos configurado que la señal de salida del módulo CLC1, la que cambiaba de estado cada 250 mS, se obtenga por las patillas RC1 y RC2 del puerto C, que precisamente están conectadas con los leds verde (D9) y rojo (D11).



8-7-5 Compilación y comprobaciones

Configurados los periféricos que vamos a emplear en este proyecto, el módulo TMR0 y el módulo CLC1, pulsamos **Generate** de la ventana **Project Resources** para generar automáticamente el código que configura el funcionamiento de esos módulos. Ahora….

1. Observa el programa fuente generado, **el main.c**. Puedes ver que se han incluido las funciones de configuración producidas por el MCC:

    a. **#include "mcc_generated_files/mcc.h"**
    b. **SYSTEM_Initialize();**

2. También puedes observar que el bucle principal está vacío:

```
while (1)
    {
        /* No hay código para ejecutar. La intermitencia se hace por hardware
        sin intervención de software alguno */
    }
```

Esto implica que una vez el controlador ejecuta las funciones de configuración, ya no tiene nada que hacer. Lo podemos detener y dejar en cualquiera de los múltiples modos de bajo consumo (IDLE). La intermitencia se mantiene.

También puede ejecutar cualquier tarea y desentenderse totalmente de la intermitencia. Esta se sigue manteniendo.

En resumidas cuentas en este ejemplo, salvo en la configuración inicial, la intermitencia se realiza por hardware con la intervención independiente y combinada de los módulos TMR0 y CLC1. ¿Qué te parece?

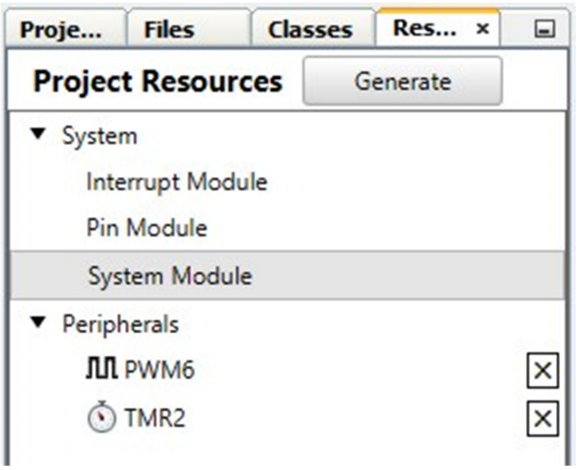
Por supuesto que puedes cambiar a tu antojo todas las configuraciones y analizar el resultado que producen. Recuerda que si cambias algo, debes hacer que el MCC genere el nuevo código **y compilar** todo el conjunto para obtener el nuevo fichero *.HEX que debes grabar en ArduPLIC

8-8 A8 – PWM_Blink

Otro ejemplo de demostración de la potencia del PIC16F18855 de ArduPIC y de la herramienta MCC de MPLAB-X IDE. En esta ocasión usamos el módulo TMR2 y el módulo PWM6 para producir una intermitencia de duración variable sobre el led ámbar (D10) conectado con la señal RA5 del PIC. Carga el proyecto y activa el generador de código de configuración MCC.

8-8-1 Project Resources

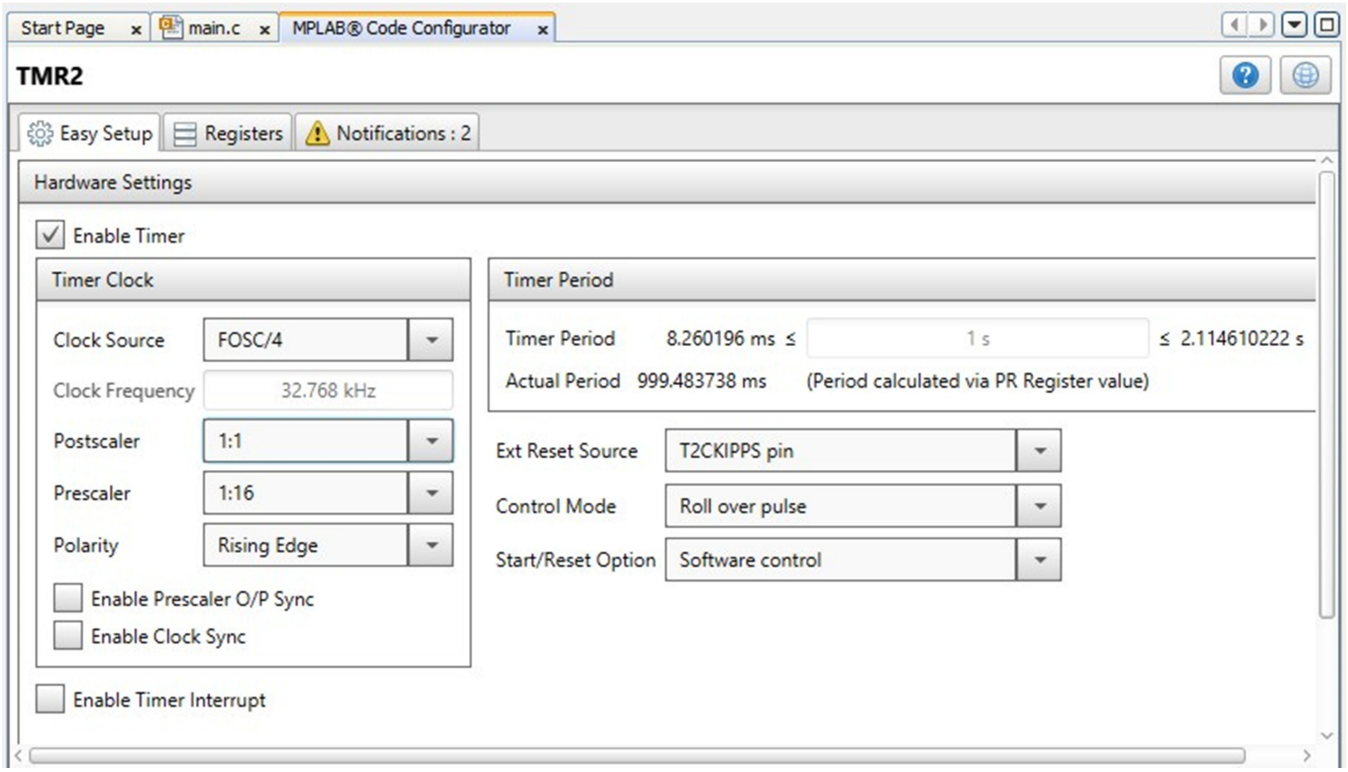
Colocamos en esta ventana los dos periféricos que van a formar parte de este proyecto: el módulo TMR2 y el módulo PWM6. Basta con hacer doble click sobre cada uno de ellos desde la ventana **Device Resources**.



Además de estos módulos, ya sabes que por defecto también están los módulos de interrupción, de pines y del sistema. Los vamos a dejar como están.

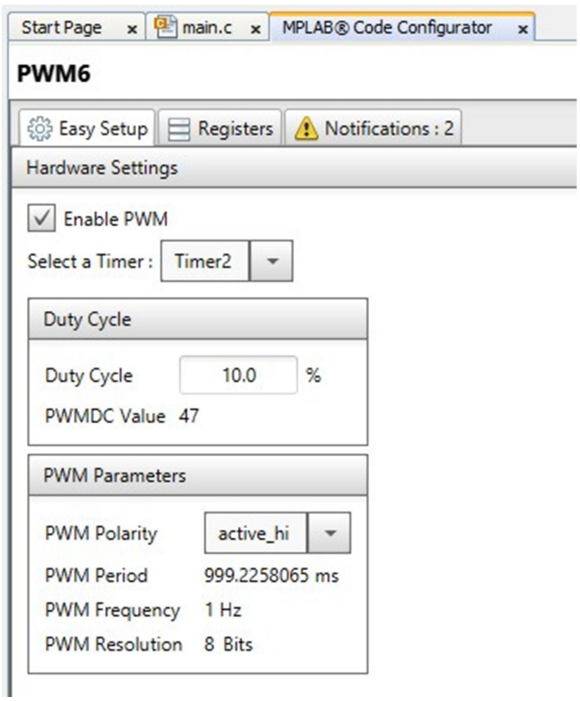
8-8-2 Configuración del TMR2

Selecciona las distintas configuraciones mostradas en la figura: Fuente de reloj, postcaler, prescaler, polaridad, periodo, etc… Como de costumbre debes recurrir al Data Sheet que facilita Microchip para estudiar todas las posibilidades que tiene este módulo. En este ejemplo hemos configurado el TMR2 para producir un periodo de 1 segundo.



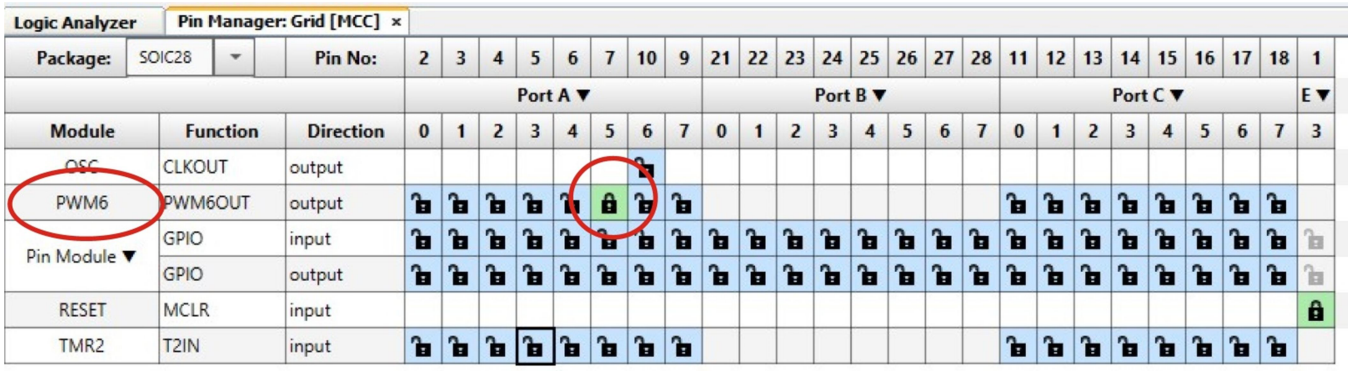
8-8-3 Configuración del módulo PWM6

Aquí vamos a seleccionar el temporizador usado como base de tiempos, el TMR2 en este ejemplo. Podemos elegir la polaridad de la señal PWM, si el ciclo útil es activo por nivel "1" o por nivel "0". Por último y lo más importante, podemos determinar el porcentaje del ciclo útil respecto al periodo. Por ejemplo, si el periodo es de 1", el 10% del ciclo útil significa que la señal PWM se mantiene activada durante 0,1" y desactivada durante los 0,9" restantes.



8-8-4 Configuración de las E/S

Finalmente vamos a configurar la patilla por la que queremos dar salida a la señal PWM. Lo hacemos mediante la pestaña **Pin Manager:Grid** como se muestra en la figura.



En este caso la salida de la señal que proporciona el módulo PWM6 se realiza por la patilla RA5 que precisamente está conectada con el led ámbar D10 de la tarjeta de periféricos BASIC I/O.

8-8-5 Compilación y comprobaciones

Configurados los periféricos que vamos a emplear en este proyecto, el módulo TMR2 y el módulo PWM6, pulsamos **Generate** de la ventana **Project Resources** para generar automáticamente el código que configura el funcionamiento de esos módulos.

Al igual que en el ejemplo anterior el MCC ha generado el código necesario para configurar al TMR2, el módulo PWM6 y las patillas de E/S. Sin embargo el bucle principal del programa está vació, no hay más código que ejecutar.

Nuevamente vemos que una vez configurado el hardware del PIC16F18855, éste puede funcionar sin intervención alguna del software. Prueba a cambiar el porcentaje del ciclo útil y también la duración del periodo.

Acuérdate de generar el código automáticamente y luego de compilar todo el conjunto. Esto lo debes hacer siempre que modifiques algo en el proyecto.

8-9 A9 – Regulación

En este ejemplo se emplea tres módulos o periféricos de los muchos que tiene el PIC16F18855 de la tarjeta ArduPIC: el módulo convertidor ADC, el temporizador TMR2 y el módulo PWM6. Se configuran mediante el MCC.

Se trata de generar una señal PWM cuyo ciclo útil varía en función de una entrada analógica, la A0 que está conectada con la señal RA0 (ANA0) y procede del potenciómetro V1 de la tarjeta BASIC I/O.

La salida de la señal PWM se realiza por la patilla RC2 del PIC que está conectada con el led rojo D11 de BASIC I/O.El cuerpo principal del programa es muy sencillo…

```
int ADC0;                          //Resultado de la conversión
void main(void)
{
   SYSTEM_Initialize();         //Inicialización del PIC según el MCC
   while (1)
   {
      ADC0=ADCC_GetSingleConversion(channel_ANA0);    //Lectura del canal ANA0
      PWM6_LoadDutyValue(ADC0);                    //Ajustar el ciclo útil de la señal PWM
   }
}
```

En resumidas cuentas. El brillo del led se puede ajustar mediante el potenciómetro

8-10 A0 – Sensores analógicos

En este último ejemplo vamos a leer el valor analógico procedente de los sensores de la tarjeta BASIC I/O: Potenciómetro 1 (A0), Potenciómetro 2 (A1), Sensor de luz (A2), Sensor de IR (A3) y Sensor de temperatura (A4).

Tras la conversión se realizan los cálculos para obtener las tensiones equivalentes en ambos potenciómetros y la temperatura en grados centígrados (ºC).

Los resultados se transmiten vía serie mediante el EUSART. Para poder verlos deberás disponer en tu PC de algún tipo de programa de comunicaciones como el Hyperterminal, Tera Term, DockLight o cualquier otro similar.

Mediante el pulsador D4 (RB2) se inicia/para la secuencia del programa y se emite una señal sonora por el altavoz D2 (RB0) de la tarjeta BASIC I/O.

El ejemplo emplea los módulos ADC y EUSART además de las clásicas patillas de E/S. Analiza sus configuraciones.