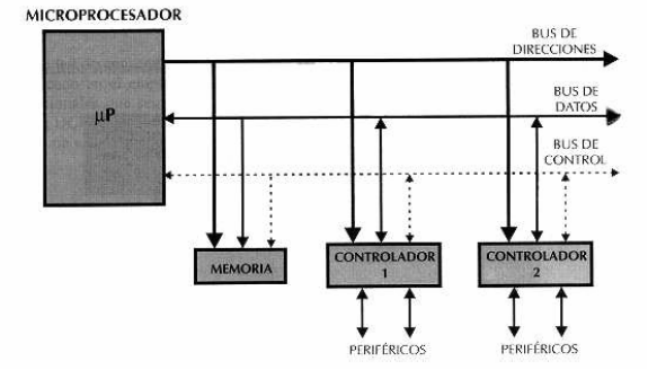
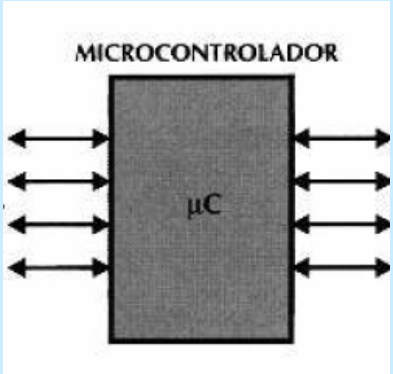
Microcontroladores PIC

Ofertamos la posibilidad de hacer el proyecto de la asignatura usando PICs en vez de Arduinos (AVR).

Microcontrolador = ordenador en un chip, incluyendo procesador, memoria, algunos periféricos.

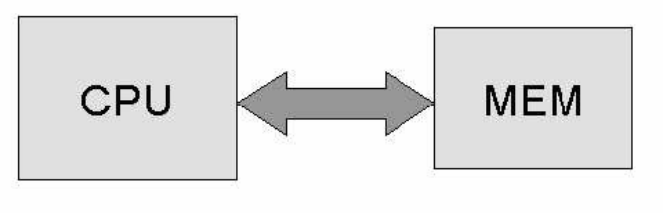
Al contrario que un microprocesador es “autosuficiente”



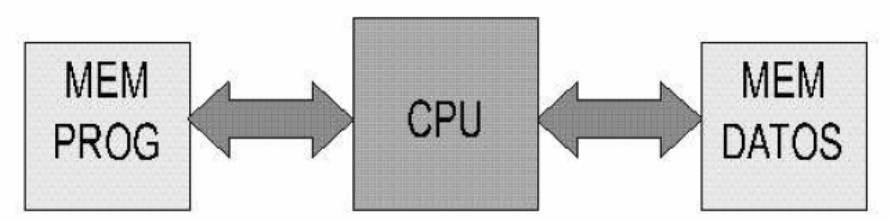


Microcontroladores PIC

Arquitectura Harvard: diferente memoria (y por lo tanto buses) para memoria de programa y memoria de datos.
Familia PIC18: 8 bits datos, 16 bits programa



Arquitectura Von Neumann



Arquitectura Harvard

No nos afecta demasiado trabajando en alto nivel (lenguaje C), pero en un microcontrolador el alto nivel nunca está demasiado lejos del hardware. Interesante tener una idea de cómo andan las cosas por abajo.

Familias microcontroladores PIC

PIC10, PIC12 → 8 bits, pocas patas, pocos recursos (módulos/perifericos)

PIC16 → gama media de 8 bits. Bastantes periféricos

PIC18→ gama alta de 8 bits

PIC24 →16 bits

dsPIC → 16 bits, especializados en tratamiento de señal (DSP)

PIC18: hasta 40 MHz (10 Mhz en ciclos de instrucción)
3/4 timers, hasta 20 fuentes de interrupción.
múltiples periféricos incorporados:

- 3 / 4 puertos I/O (8bits)
- Salidas Pulse Width Modulation (PWM)
- Conversor Analógico/Digital 10 bits
- Comunicaciones serie (UART, SPI, I2C)
- USB en algunos modelos.

PIC18F2540/4520

MEMORIA:

- 32K memoria flash de programa reprogramable (16K instrucciones)
- 1536 bytes de datos (RAM)
- 256 bytes EEPROM no volatil (como un periférico)

20 fuentes de interrupción con dos niveles de prioridad.

13 canales ADC con 10 bits de resolución.

Comunicaciones serie UART (RS232,R485), y síncrona SPI, I2C

Dos canales PWM, dos comparadores

WatchDog Timer → hasta 130 segundos

Multiplicador hardware 8x8 en un solo ciclo

Autoprogramable (permite reprogramarlo a través del puerto serie)

Special Function Registers (SFR)

Dentro de la memoria de datos, tenemos los SFR (Special Function Registers) a través de los cuales controlamos la mayor parte de las funciones del micro y sus periféricos.

Cada periférico tiene asociado 2 o 3 SFR a través de los cuales se controla.

Ejemplos:

- Puertos B de entrada/salida: TRISB, PORTB, LATB
  - TRISB determina si es entrada (1) o salida (1)
  - En PORTB leemos valores del puerto.
  - Usamos LATB para asignar valores a un puerto.

- Comunicaciones USART:
  - TXREG, TXSTA: datos a transmitir, status/configuración de TX
  - RCREG, RCSTA, datos recibidos, status/configuración de RX
  - BRGH → establece la velocidad del puerto (baudios)

  Se usan directamente con esos nombre dentro del compilador

Bits de configuración

También, dentro de la memoria del PiC hay un par de bytes que comprenden los llamados bits de configuración.

Se definen dentro de nuestro programa C (o al crear el proyecto) y configuran aspectos básicos de la configuración del PIC al arrancar.

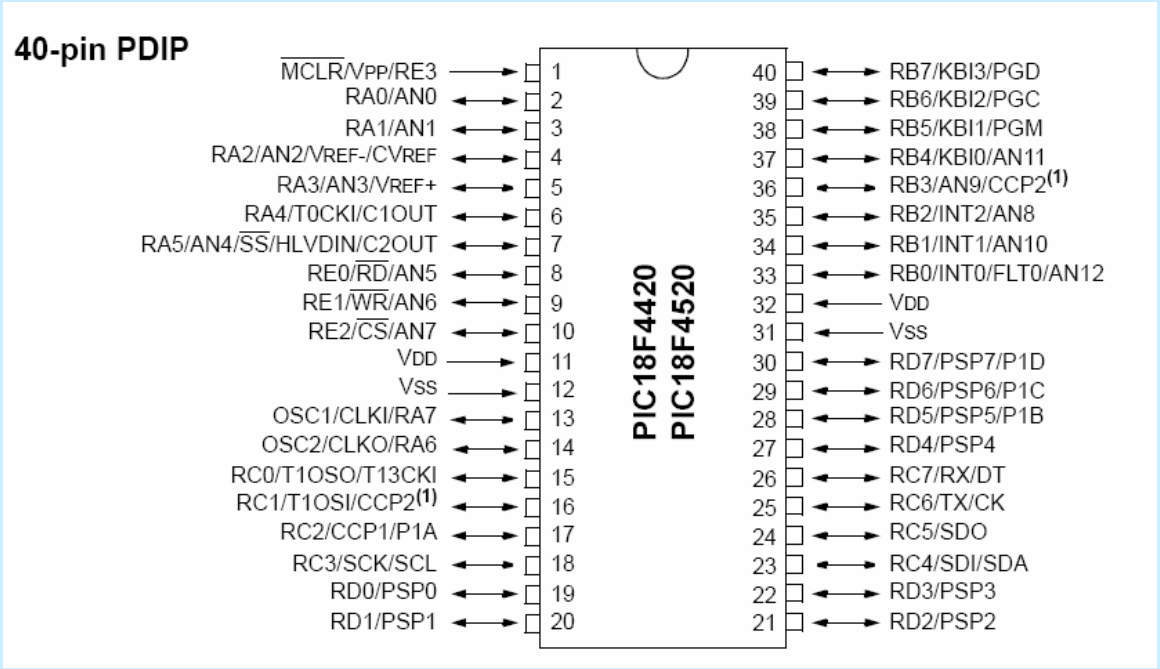
Ejemplos: habilitación/deshabilitación del watchdog timer
Decidir si tras un reset el puerto B se dedica a entradas analógicas o a un puerto normal de entrada/salida digital.

Si no los ponemos correctamente pueden ser fuente de errores frustrantes, sobre todo al principio:

- Si el puerto B arranca como entrada analógica no responderá a nuestros comandos para ponerlo a 0/1

- Si el “perro guardián” está activo pero nosotros no nos encargamos de “darle el hueso” el PIC se va a resetear cada poco tiempo y nuestro programa no va a funcionar.

Disposición de los 40 pines en el 18F4520



Uso básico de un pin: entrada/salida digital

La mayoría de los pines pueden tener varias funciones (excluyentes)

PUERTOS ENTRADA / SALIDA

4 o 5 puertos de entrada/salida RA, RB , RC, RD , RE

RB, RC, RD están completos → 8 pines
RA, RE → faltan algunos, dependiendo de nuestra configuración.

Registros asociados PORT, LAT, TRIS

TRISA = 0 / 1 → determina si el pin es de salida/entrada

PORT para leer el valor de un puerto, LAT para escribirlo 1 → high (5V)
0 → low (0V)

De hecho PORT puede usarse para ambos menesteres, pero se aconseja diferenciar entre lecturas/escrituras

En bloque o por separado LATC=255 (todos los pines de portC =1)
LATCbits.RC8 =1 (pin 8 de PORTC = 1)

Un pin puede dar (o recibir) un máximo de unos 25 mA.

TIMERS

- Un timer es un contador que se incrementa cada ciclo máquina (aunque algunos pueden configurarse para contar una entrada externa).

- En los PIC un ciclo máquina son 4 oscilaciones del reloj, luego con un cristal de 4 Mhz, los timers se incrementarán cada microsegundo.

- Tenemos hasta 4 timers, aunque algunos de ellos pueden estar "usados" si ciertos módulos están funcionando (p.e. PWM usa timer2)

- Modos de 8 y 16 bits: podemos definir el contador como 8 o 16 bits.

- Esto es importante porque al rebosar el contador 0xFF → 0x00 se activa la correspondiente interrupción del timer en cuestión.

- Si dicha interrupción está habilitada, saltamos a la rutina de procesar las interrupciones. Esto nos permite ejecutar una tarea periódica sin estar pendientes del contador.

INTERRUPCIONES

20 fuentes de interrupción:

4 interrupciones por rebosamiento de timers.

4 interrupciones causadas por cambios de un pin (RB0, RB1, RB2, RB4-7)

Interrupciones asociadas a periféricos:

TX → el registro de transmisión está vacío y listo para enviar un byte.
RX → acabamos de recibir un byte.

AD → ha terminado una conversión AD. Podemos acceder al resultado.

y muchas más.

2 niveles de prioridad: interrupciones altas pueden interrumpir a las bajas.

Es fundamental usar las interrupciones durante la programación.

Conversor Analógico-Digital (ADC)

Hasta 13 canales (no simultáneos) con una resolución de 10 bits.

Convierte voltaje externo (0-5V) en un número 0-1023

Lectura de sensores analógicos

Salida PWM

Permite simular salida “analógica” con un puerto digital: onda cuadrada de frecuencia alta (p.e 10 KHz) en la que podemos controlar el tiempo en “on “.

Disponemos de 2 canales de PWM con una misma frecuencia (programable) pero con diferente % en “on” (hasta 10 bits de resolución).

Muy útiles para controlar motores (junto con un H-bridge para controlar dirección y un driver de potencia)

Comunicaciones serie:

Asíncronas: módulo UART
Aplicación más común: comunicación RS232 con el ordenador.
Voltajes distintos, necesitaremos un conversor de voltajes.

PIC: 0 lógico = 0V 1 lógico = 5V
PC 0 lógico = [3V/15V] 1 lógico [-3V/-15V]

Comunicaciones síncronas: Módulo Master Synchronous Serial Port (MSSP)

Puede trabajar en uno de dos modos (excluyentes):

Serial Peripheral Interface (SPI)

Inter Integrated Circuit (I2C)

En ambos casos el módulo puede configurarse como master o slave.

Permiten comunicaciones con numerosos periféricos.

Enlaces sobre estos temas

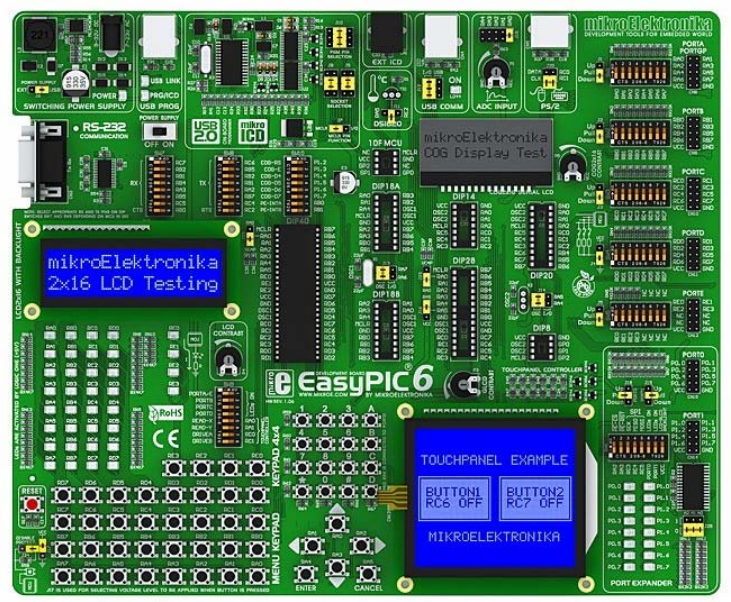
Tengo escritos algunos tutoriales sobre programación de PICs en C.
Cubren aspectos básicos sobre:
- Temporizadores, Interrupciones
- Uso del ADC, PWM
- Comunicaciones serie

http://picfernalia.blogspot.com.es/

Numerosos otros recursos en la web:

- http://embedded-lab.com/
- http://www.todopic.com.ar/
- http://www.mikroe.com/products/view/285/book-pic-microcontrollers-programming-in-c/
- http://picprojects.org.uk/projects/picprojects.htm
- http://www.coolcircuit.com/gadgets/category/pic-projects/

Placa de desarrollo: Easy PIC 6



Conexión a través de USB (instalar los drivers y software del programador)

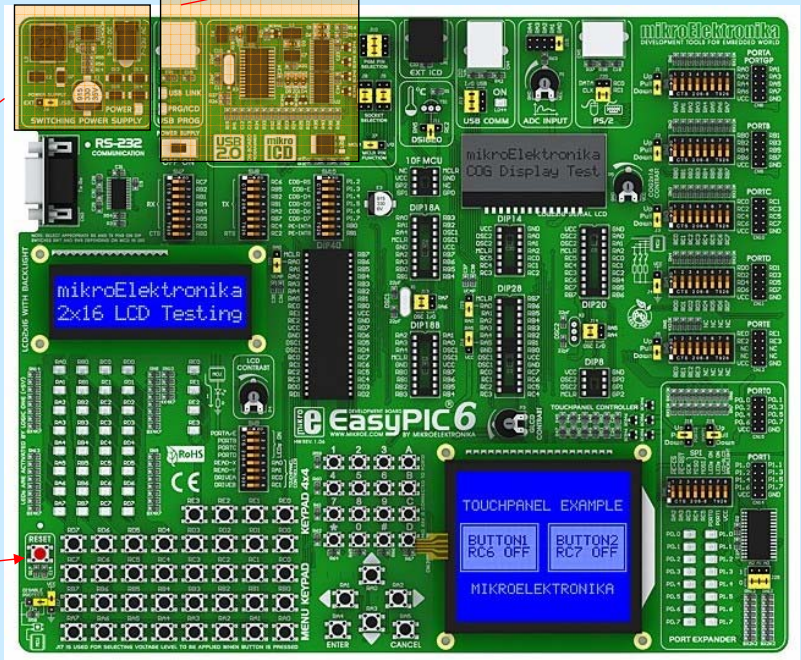
http://www.mikroe.com/downloads/get/1202/mikroprog_for_pic_drivers_v200.zip

http://www.mikroe.com/downloads/get/1201/mikroprog_suite_for_pic_v226.zip

Programador, alimentación (5V)

Conexión USB para programar

Alimentación (hasta 500 mA)



Alimentación externa (si es necesaria)

Se controla con un switch.

Botón de reset.

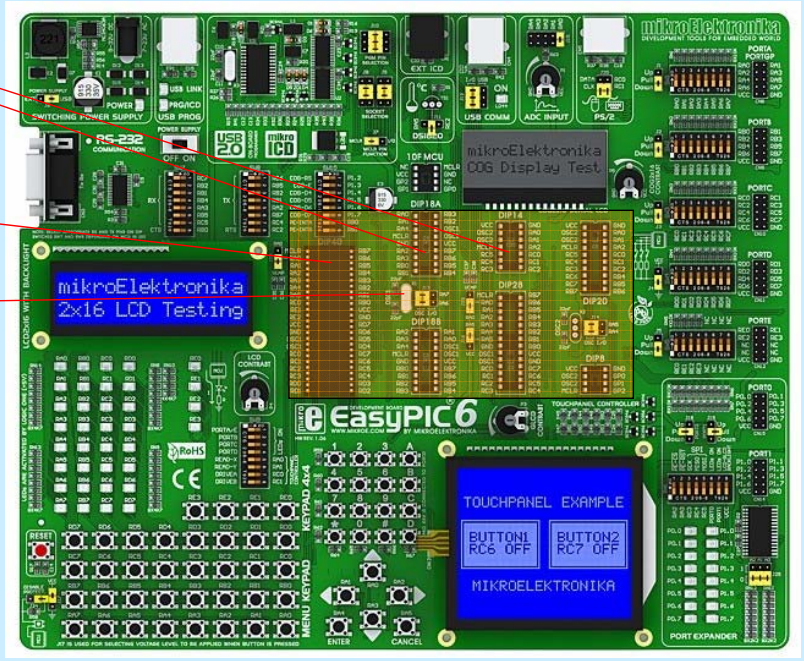
Resetea el micro.

Microcontrolador

Zócalos para distintos modelos de PICs: 8,14,18,20 28 y 40 pines

Solo uno de ellos puede estar ocupado.

Cristal incluido



Entrada / Salida

LEDs conectados a todos los pines. Permiten ver su estado: 1 = encendido

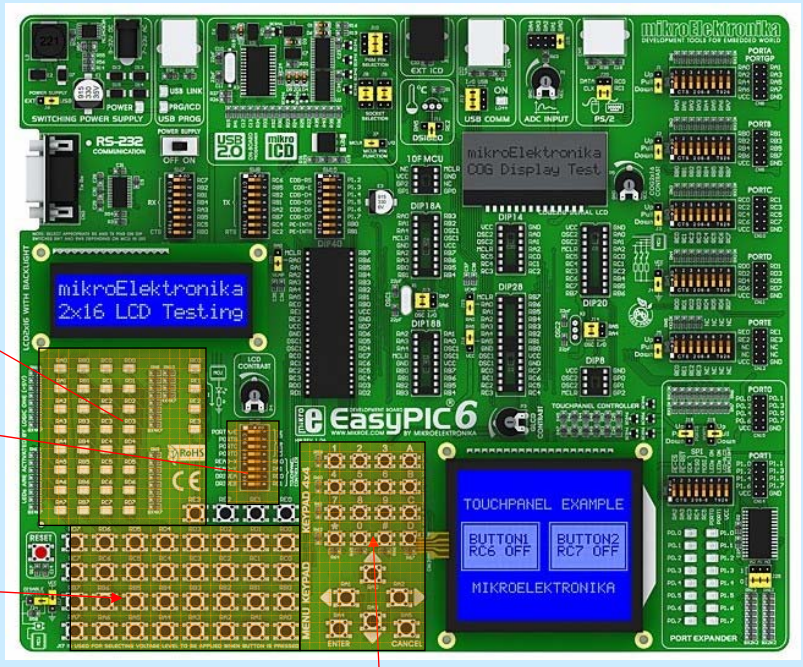
0 = apagado

Un microswitch permite decidir si los LEDS están conectados o no.

Pulsadores conectados a todos los pines.

Pueden configurarse para estar a 0 o 1 por defecto y cambiar al pulsar.

Para usarlos debemos declarar el pin de entrada.



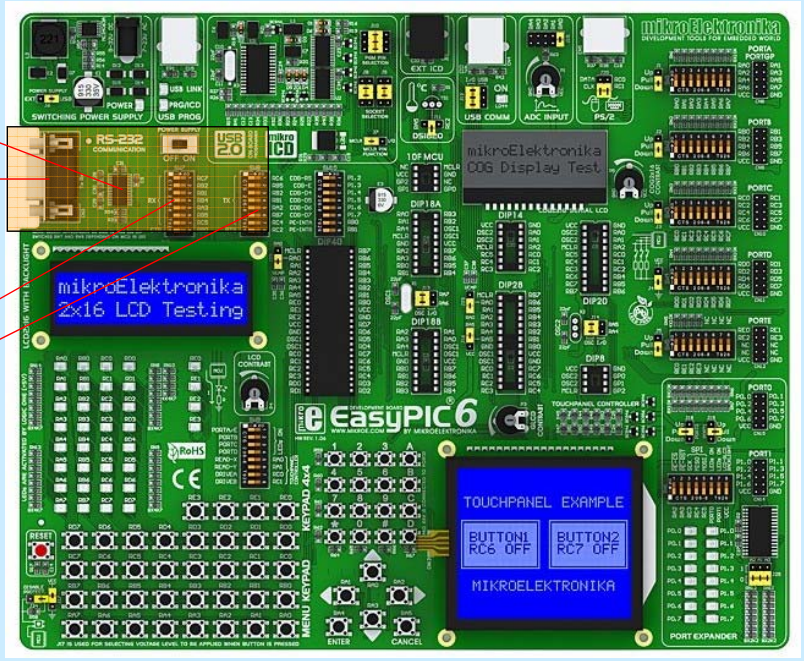
Teclados keypad 4x4
Teclado de flechas.

Puerto Serie RS232

Conversor de niveles (0-5V) a (-15/+15)V

Conector DB9 directamente conectable a RS232 del ordenador

Selección del pin del PIC usado como RX/TX



Pantalla LCD

Pantalla LCD.

Protocolo controlador Hitachi

Conectada a RB0-RB3 (datos) + RB4 / RB5 (RS, Enable)

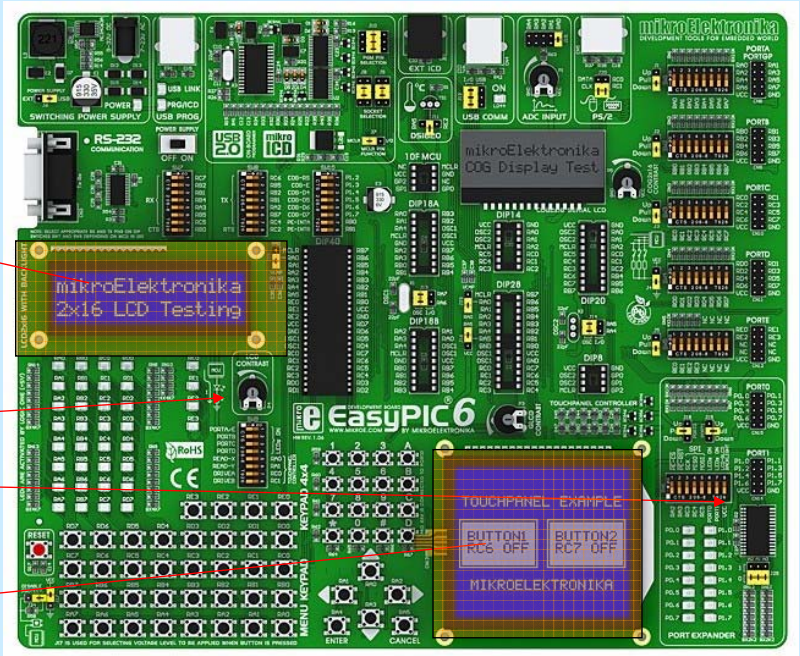
Modo de 4 bits de datos

Contrastre

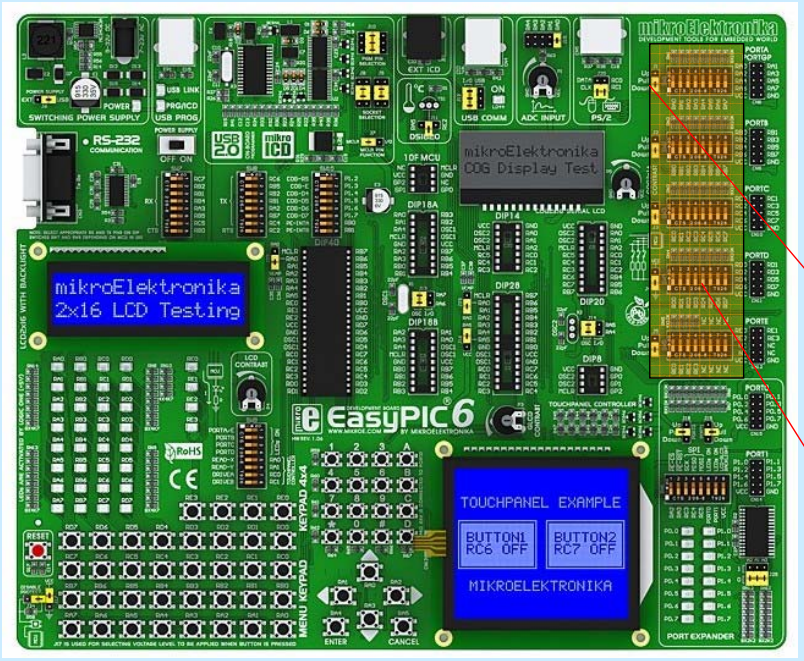
Backlight

Pantalla gráfica LCD

Usa puertos RB,RD



Pull-ups / Pull-down



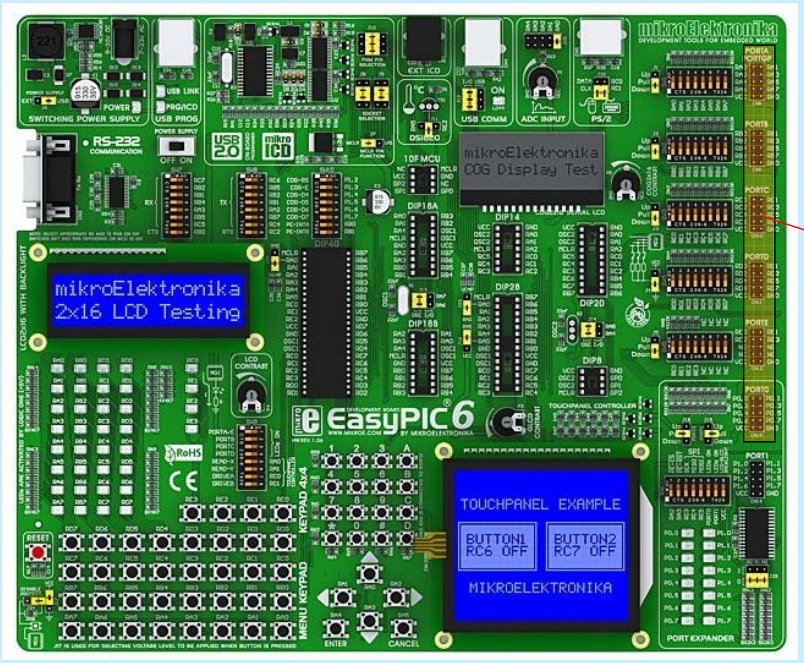
A veces es interesante que un pin por defecto esté a un nivel alto (5V) o bajo (GND)

Los pull-ups (resistencia desde el pin a 5V) mantienen el pin a nivel alto si no se le fuerza a 0.

Un jumper permite elegir entre pull-up (atado a 5V) o pull-down (atado a 0 V) para todo un puerto

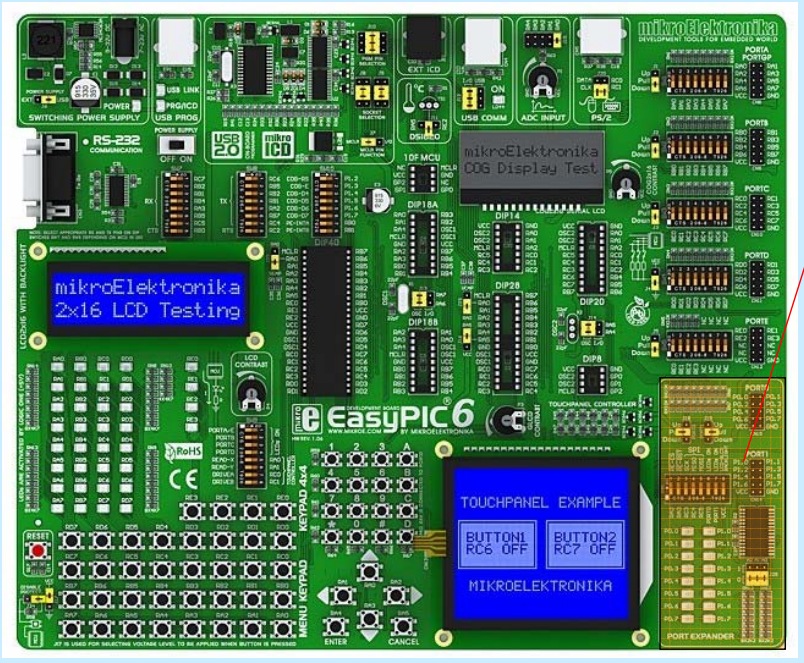
Un microswitch permite seleccionar los pines dentro del puerto.

Acceso a cualquier pin de cualquier puerto



Para todos los pines de todos los puertos tenemos un contacto por si queremos conectarlo a cualquier otra cosa fuera.

Extensor de puertos



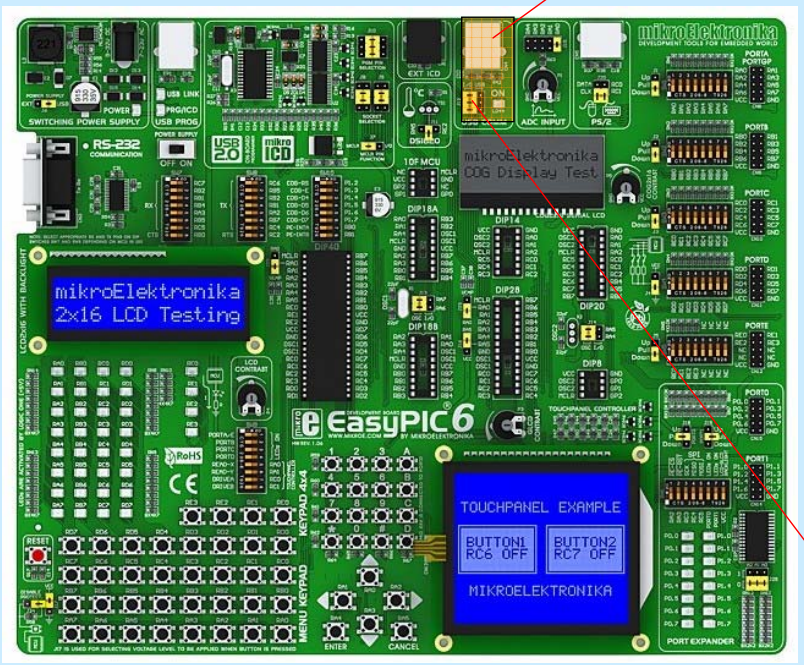
Por si acaso precisáramos más pines tenemos disponible un extensor de puertos.

Nos comunicamos con el a través de una comunicación serie y nos permite disponer de dos puertos extra.

No lo usaremos en nuestros proyectos.

Conexión USB

Comunicación USB con otro dispositivo. Solo para aquellos PIC con capacidad USB.



No confundir con la conexión USB del programador.

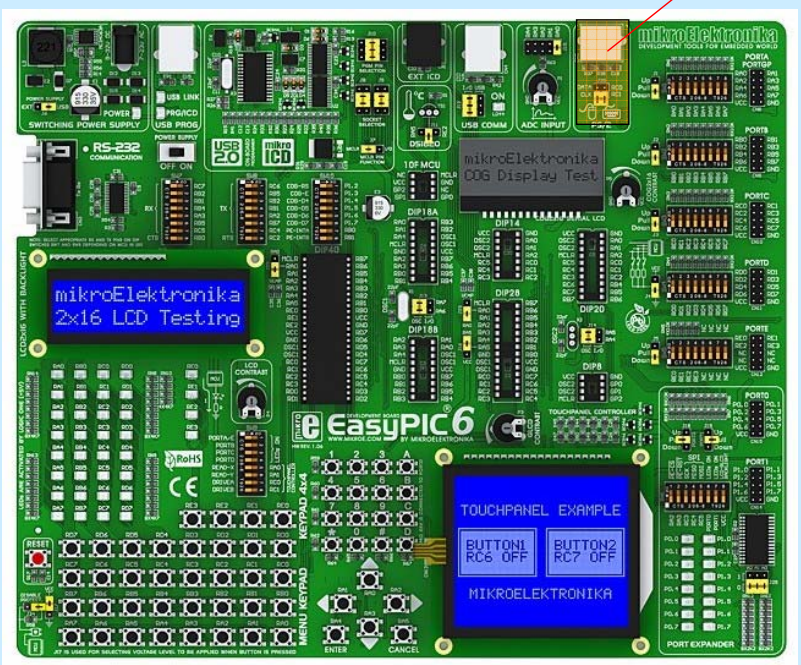
La conexión USB del programador solo está activa al programarse el PIC.

Esta comunicación USB se realiza dentro del programa del PIC, como alternativa a otros medios de comunicación.

Los jumpers permiten seleccionar que pines del PIC se encargan del USB.

Conexión PS2

Permite la conexión directa (hardware) con un dispositivo PS2 (ratón/teclado)



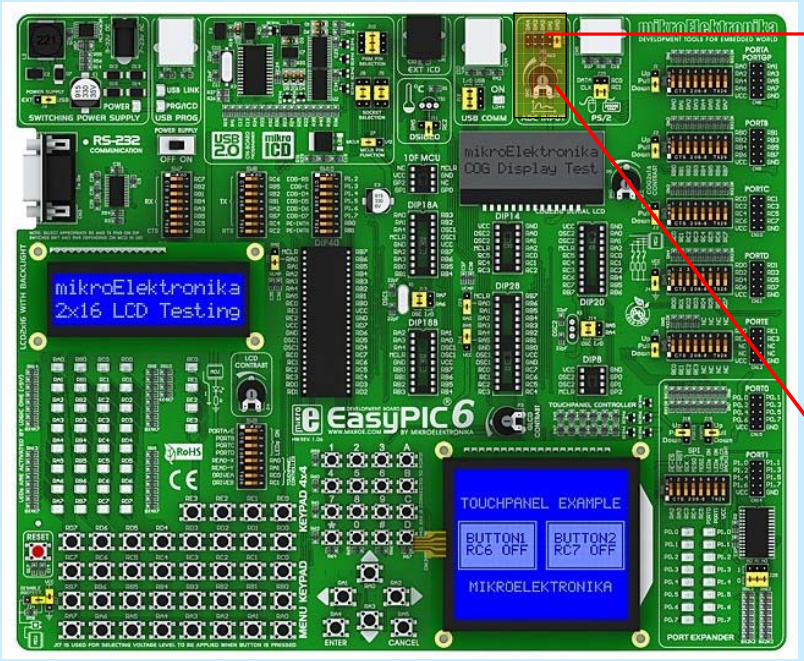
No hay ningún tipo de soporte hardware para el protocolo PS2 dentro del PIC.

Todo hay que hacerlo por software.

Los jumpers permiten seleccionar que pines del PIC se conectan a los diferentes hilos del ratón/teclado USB.

Test del ADC

Permite hacer un test rápido de un programa que maneje un ADC sin necesitar un sensor o voltaje externo.



Los jumpers determinan a que pin conectamos la salida del potenciometro.

Sirve para hacer una prueba rápida de nuestro código ADC antes de conectarlo a la verdadera fuente externa de datos

Dispone de un potenciometro con el que variar un voltaje entre 0 y 5 V.

Entorno de desarrollo: C18 de Microchip

Disponible en la página de Microchip:

http://www.microchip.com/

Versión gratuita para estudiantes, completa pero sin algunas optimizaciones.

ttp://www.mikroe.com/downloads/get/29/mikroc_pro_pic_2011_v561.zip

Requiere instalar IDE MPLAB + compilador C18

http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en010014

Seguimos teniendo que usar el programador de la placa (MikroElektronica).

No tiene limitaciones de tamaño de programa.

Primer programa en C18

Crear proyecto con el wizard: especificar dispositivo

```
#include <p18F4520.h>
#include <delays.h>

#pragma config OSC=HS
#pragma config WDT = OFF
#pragma config PBADEN = OFF,


void main()
{
 TRISC=0; PORTC=0x55;
 while(1)
  {
   PORTC=~PORTC;
   Delay10KTCYx(100);
  }
}
```

includes necesarios

Bits de configuración

Declaro puerto C salida

Valor = 0x55 (1 y 0 alternos)

Bucle infinito

Invierto PORTC

Espero 100x10000 ciclos

Proyectos:

Interfaz con un ratón PS2 y LCD gráfico.

Temas tratados:

- Protocolo PS2 ratón + Protocolo del LCD gráfico (similar a LCD texto).
- Interrupciones (para detectar comunicaciones en el protocolo PS2)
- Temporizadores para refresco pantalla
- Uso de un "buffer" de memoria gráfica en el PIC.
- Más opciones: añadir una memoria flash y hacer un sniffer de teclado que guarde las pulsaciones del teclado. A través de una conexión serie, implementar un juego de comandos para borrar datos, volcarlos al ordenador, etc.

Hardware: sin complicaciones adicionales.

conectores en placa para LCD gráfico + conexión PS2

Proyectos:

Control de dos servos con un Nunchuk

Temas tratados:

- Comunicaciones I2C con nunchuk
- Presentación datos (acelerometros/joystick) en LCD.
- Temporizadores para el manejo de servo motores
- Montar servos en plataforma azimuth-elevación
- Más opciones: - Fijar numchuck a la plataforma y usar sus acelerometros hacer una plataforma "estabilizada“

  - Montar sensor ultrasonidos sobre plataforma y enviar datos de distancias a través del puerto serie.

Hardware: conversores voltaje 5V - 3.3V

alimentación separada para servos

Proyectos: Datalogger con una tarjeta SD.

Temas tratados:

- Comunicaciones SPI.
- Protocolo tarjetas SD
- Conversor ADC

Más posibilidades:

- Modos de consumo reducido (despertar cada cierto tiempo, tomar una medida, volver a dormir)
- Reproductor de ficheros de audio que lea fichero WAV de tarjeta SD y use algún tipo de DAC para oírlo por un altavoz.

Hardware: conexión a tarjeta SD

conversor voltaje 5V-3.3V

driver de potencia, para el caso del altavoz

Proyectos:

Levitador magnetico

Temas tratados:

- Uso de PWM para controlar una bobina.
- Uso del ADC para leer un sensor de flujo magnético.
- Interrupciones, temporizadores.
- Control PID para hacer levitar un imán

Hardware: Driver de potencia, H-bridge.
Alimentación separada para bobina
Soporte + construcción de una bobina.