Práctica P5_ADC_V2
Luis Castillo e Ian Fernandez

CONVERSIÓN DE DATOS ANALOGÍCOS A DIGITALES MEDIANTE EL ADC
Sincronización de los dispositivos (encuesta o interrupción)

- **Pulsador S4:** Se gestiona por encuesta
- **Pulsadores S3, S6:** Se gestionan por interrupción
- **Timer T9:** Se gestiona por encuesta
- **Timers T7, T5, T3:** Se gestionan por interrupción
- **ADC1:** Se gestionan por encuesta

Funciones que llama el programa principal

- **inic_oscilator** (en oscilator.c)
  - Inicializa y configura el módulo oscilator con una frecuencia de 40MHz
- **inic_leds** (GPIO.c)
  - Define e inicializa los puertos correspondientes a los leds
- **inic_pulsadores** (GPIO.c)
  - Define e inicializa los puertos correspondientes a los pulsadores
- **inic_LCD** (LCD.c)
  - inicializa el display LCD estableciendo los estados iniciales para data y pines de control e inicializa la secuencia de inicialización de la LCD
- **copiar_FLASH_RAM** (memoria.c)
  - Recibe una cadena de texto y asigna sus valores a la variable Ventana_LCD la cual se usa para mandar datos al display
- **line_1** (LCD.c)
  - Establece el cursor en la primera línea del display
- **puts_lcd** (LCD.c)
  - Recibe la variable Ventana_LCD y despliega su información en el display LCD
- **line_2** (LCD.c)
  - Establece el cursor en la segunda línea del display
- **inic_crono** (timers.c)
  - Inicializa las variables de cronómetro y cambia el valor del flag_inic_crono a 1
- **inic_Timer7** (timers.c)
  - Inicializa y configura el Timer 7
- **inic_Timer5** (timers.c)
  - Inicializa y configura el Timer 5
- **inic_Timer3** (timers.c)
  - Inicializa y configura el Timer 3
  - Se inicializa el registro de la cuenta a 0 con TMR3 y se le indica que ha de contar 40.000 ciclos con PR3
  - El preescaler TCKPS = 0 se establece a 1:1 porque no se excede del máximo 65.536 ciclos
  - El reloj interno y el modo gate se dejan en 0 TCS y TGATE
  - Se activan las interrupciones mediante T3IE y se pone a 0 el flag de interrupción T3IF
  - Por último se pone en marcha el timer con TON = 1
- **inic_CN** (CN.c)
  - Inicializa el módulo CN
- **Inic_UART2** (UART_RS232.c)

- Inicializa el módulo UART2 para recibir y transmitir información

- **Inic_ADC1** (ADC1.c)
  - Inicializa el ADC1 para convertir señales analógicas en digitales
  - SSRC está establecido en 7 que es el auto converter, para sincronizarlo con el timer 3, habría que ponerlo a 2
  - ASAM = 1 establece el muestreo automático
  - AD1CON2, AD1CON3 y AD1CON4 inicializan los registros de control a 0
  - SMAC y ADCS son los campos que indican cada cuantos ciclos de reloj se realiza el muestreo
  - El registro CHOSA selecciona la entrada analógica.
  - Con los registros AD1PCFGH y AD1PCFGL se ponen todas las AN como digitales exceptuando las que vayamos a usar (El potenciómetro, la temperatura, Px, Py y la palanca)
  - Como siempre, tenemos el interrupt enable AD1IE y el flag de interrupción AD1IF, ambos a 0.
  - Por último con ADON se habilita el ADC
- **crono** (timers.c)
  - Realiza el conteo de las variables del cronómetro, en cada cambio de valor de una de estas variables se realiza la conversión de tiempo de número entero a ASCII con la función conversion_tiempo de modo que la variable Ventana_LCD actualiza su contenido.
- **comprobar_inic_crono** (timers.c)
  - Evalúa el cambio del flag_inic_crono y realiza la conversión del tiempo de las variables del cronómetro, esto se realiza para que estas operaciones no se hagan directamente en la rutina de atención
- **calcularMediaMuestras** (ADC1.c)
  - Cuando el ADC1 ha recogido 8 muestras por cada entrada analógica, realiza una media de esas 8 muestras para cada una de las entradas. Cuando realiza esto, vuelve a habilitar el ADC que en la llamada a esta función se había deshabilitado y pone a 0 el *flag_muestras* que indica cuando se han realizado todas las muestras de cada entrada.

Otras funciones importantes

- **lcd_cmd** (en LCD.c)
  - Se encarga de enviar los comandos necesarios para el funcionamiento de la pantalla LCD.
- **lcd_data** (en LCD.c)
  - Subrutina encargada de imprimir un carácter en la pantalla LCD.
- **Delay_ms** (en timers.c)
  - Función que espera el tiempo pasado por parámetro en milisegundos.
- **Delay_us** (en timers.c)
  - Función que espera el tiempo pasado por parámetro en microsegundos.
- **conversion_tiempo** (en utilidades.c)
  - Función que, dados dos dígitos, devuelve estos en forma de caracteres.
- **conversion_adc** (en utilidades.c)
  - Función que, dados cuatro dígitos, devuelve estos en forma de caracteres. Es una ampliación de la función *conversión_tiempo* para poder representar los valores obtenidos por el ADC

- **recoger_valorADC1** (en ADC1.c)
  - Esta función se encarga de tomar las muestras de cada entrada analógica y guardarla en un vector para posteriormente mostrar el valor medio de las conversiones realizadas y no el ADC no toma muestras constantemente. Se evalúa el registro CHOSA y se almacena el valor recogido por la entrada analógica correspondiente, despues se cede el ADC para la siguiente entrada. Cuando se han recogido 8 muestras de todas las entradas, se activa el *flag_muestras* para realizar la media y se inhabilita el ADC.

Rutinas de atención

- **Rutina del módulo CN** (en CN.c): _CNInterrupt()
  - Comprueba el pulsador S3 y si está pulsado para o pone en marcha el timer 7.
  - Comprueba el pulsador S6 y si está pulsado reinicia el cronómetro y para el timer 7.
- **Rutina del timer T7** (en timers.c): _T7Interrupt()
  - Simplemente actualiza la variable mili cada 10ms e inhabilita el flag de interrupción.
- **Rutina del timer T5** (en timers.c): _T5Interrupt()
  - Se implementa la máquina de estados para que envíe datos y comandos al LCD cada 2,5ms.
- **Rutina de la UART2** (en UART2_RS232.c): _U2RXInterrupt()
  - Evalúa el registro recepción de la UART2, si el carácter recibido es “p” o “P” para el cronómetro, si es “i” o “I” lo inicializa y si es “c” o “C” lo pone en marcha.
- **Rutina de la UART2** (en UART2_RS232.c): _U2TXInterrupt()
  - Contiene una máquina de estados que enviará los datos y comandos necesarios a una velocidad 4800 baudios mediante la UART2 (10bits) y se verán reflejados en el terminal del Tera Term.
- **Rutina del timer T3** (en timers.c): _T3Interrupt()
  - Interrumpe cada milisegundo llamando a la función *recoger_valorADC1* e inhabilita el flag de interrupción.

FUNCIONAMIENTO GENERAL

El funcionamiento para el timer, LCD y UART2 son los mismos que para las prácticas anteriores.

En esta práctica hemos hecho uso del ADC1 junto con el timer T3 para conseguir convertir entradas analógicas en digitales, así como la del potenciómetro, sensor de temperatura, Px, Py y palanca.

El ADC1 se inicializa para que interrumpa cada vez que el timer 3 cuenta 1ms y lo que hará este, será recoger una muestra de la entrada analógica indicada. Cuando haya recogido 8 muestras de cada una de las entradas, realiza una media de las muestras recogidas y lo muestra por la pantalla LCD. De esta forma, se consigue reducir la cantidad de recursos que usa la placa reduciendo la cantidad de muestras que realiza el ADC.

En esta práctica se ha conseguido realizar en su totalidad con el muestreo del joystick y el registro de las muestras del adc mediante interrupción del timer con un periodo de 1 milisegundos. (En las capturas realizadas se puede ver como se reducen la cantidad de muestras realizadas)