Universidad Galileo
Microprocesadores

Septiembre de 2014
Eduardo Corpeño

Práctica de Laboratorio #6
Convertidor Analógico-Digital

Objetivo

Que el estudiante aprenda el procedimiento necesario para utilizar el Convertidor Analógico-Digital de un microcontrolador LPC1115, utilizando el IDE Keil μVision.

Teoría

El microcontrolador LPC1115/303 cuenta con un *Convertidor Analógico-Digital* de aproximaciones sucesivas de 10 bits, con 8 canales de entrada analógica multiplexados. En la tarjeta LPCXpresso LPC1115, el voltaje de referencia es de 3.3V, y dada su frecuencia de operación, una conversión típicamente toma 2.44μs.

En esta práctica usted desarrollará una aplicación para el LPC1115 en lenguaje ensamblador, la cual hace uso del ADC. Para su comodidad, hemos incluido la información pertinente a continuación.

ADC

Para emplear este módulo operacional, es necesario utilizar los siguientes registros:

Escritura:

- Input/Output Configuration (IOCON_PIOx_y). Para configurar el pin GPIO.
- AHB Clock Control (SYSAHBCLKCTRL). Para suministrar la señal de reloj al ADC.
- Power-Down Configuration (PDRUNCFG). Para encender el ADC.
- A/D Control Register (AD0CR). Aquí se configura el ADC.
- A/D Interrupt Enable Register (AD0INTEN). Para configurar interrupciones.

Lectura:

- A/D Global Data Register (AD0GDR). La conversión más reciente está aquí.
- A/D Channel x Data Register (AD0DRx). Conversiones de cada canal (x:0-7)
- A/D Status Register (AD0STAT). Guarda información de las conversiones realizadas.

Lenguaje Ensamblador

El CPU Cortex-M0 implementa la arquitectura ARMv6-M, la cual utiliza el set de instrucciones Thumb de 16 bits, con tecnología Thumb-2.
El código que usted verá en esta práctica contiene una cantidad muy reducida de instrucciones, cuyo uso usted debe consultar en la guía que prefiera. Nuestra recomendación siempre es la especificación de ARM (infocenter.arm.com) y el manual de su microcontrolador.

GPIO

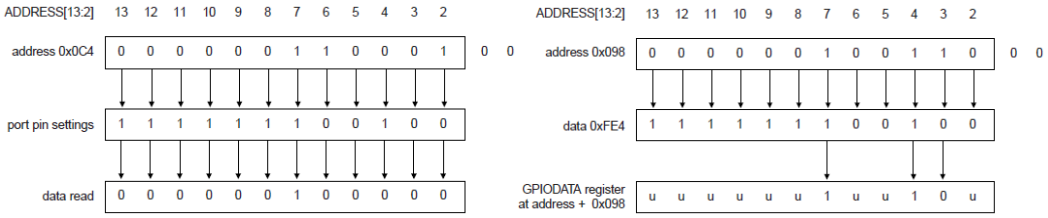
El microcontrolador LPC1115/303 cuenta con 4 puertos paralelos bidireccionales de propósito general: Los puertos 0, 1, y 2 (12 pines cada uno), y el puerto 3 (6 pines).

Cada puerto tiene un registro de datos asociado (GPIO2DATA por ejemplo). Si se escribe a este registro, se escribe a los pines configurados como salidas, y los bits configurados como entradas tienen el estado lógico de los pines asociados.

Cada puerto tiene también un registro de dirección de datos (Data Direction Register o **`GPIOnDIR`**), que se utiliza para configurar los bits del puerto como entradas o salidas individualmente (**`GPIO2DIR`** por ejemplo).

Cada pin de cada puerto tiene un registro de configuración llamado **`IOCON`** (por ejemplo **`IOCON_PIO2_11`**) que controla, entre otras cosas, la función del pin, habilitación de *Pull Resistors*, y otros modos de operación.

Los cuatro registros de datos soportan escritura y lectura enmascarada por la dirección de memoria. Es decir: La dirección de memoria que utilice
¡Sí, La dirección de memoria se está utilizando para hacer contrabando de información!
A continuación tenemos la lógica de operación de esta característica:



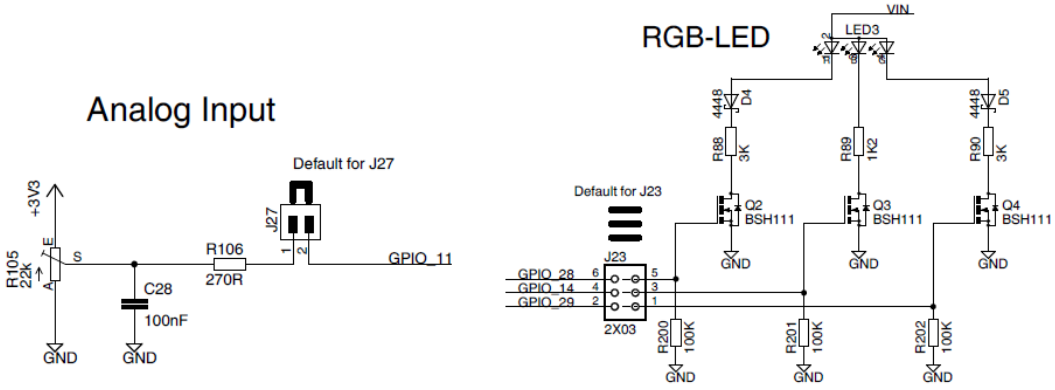
Para conocer más, puede buscar "Masked access" en el *ARM Infocenter*.

En la Tarjeta LPCXpresso Base Board

La mayoría de los pines del microcontrolador están conectados a un periférico. Algunos de estos periféricos son controlados digitalmente sin protocolos especiales. En esta práctica, usted utilizará los siguientes:

- Potenciómetro rotacional.
- LED tricolor (RGB) diseñado para generar múltiples colores con PWM.

A continuación tenemos parte del diagrama esquemático de la tarjeta LPCXpresso Base Board, en donde aparecen estos periféricos.



Materiales y Equipo **(Proporcionados por el Laboratorio)**

- Tarjeta LPCXpresso Base Board con una tarjeta LPCXpresso LPC1115.
- PC con Windows 7 y Keil µVision 4.
- 2 Cables USB 2.0 a USB Mini-B (no *Micro-B*, sino *Mini-B*).

Procedimiento

Siga los siguientes pasos prestando atención a las indicaciones de su instructor. Si tiene alguna duda o no está seguro de cómo proceder, pregunte a su instructor.

1. Asegúrese de conocer el hardware con el que trabajará. En particular verifique lo siguiente:

    a. Mantenga la tarjeta alejada de piezas metálicas como joyas y llaves.
    b. Mantenga la tarjeta alejada de piezas metálicas como joyas y llaves.
    c. Note que en adelante nos referiremos a dos tarjetas:
        i. La tarjeta LPCXpresso Base Board, que contiene periféricos.
        ii. La tarjeta LPCXpresso LPC1115, que contiene:
            1. El hardware LPC-Link, para descargar programas.
            2. El microcontrolador LPC1115 de NXP.

Tarjeta LPCXpresso LPC1115, de Embedded Artists

    d. La tarjeta LPCXpresso Base Board es alimentada por el conector USB Mini-B llamado X1 (Cerca del conector Ethernet RJ45), sin interruptor de encendido. No la conecte a la PC todavía.
    e. La tarjeta LPCXpresso LPC1115 es programada por su conector USB Mini-B llamado J3 (El único). No la conecte a la PC todavía.
    f. Para conectar los extremos USB Mini-B de los cables, asegúrese de sujetar *el conector hembra* y ***NO*** la tarjeta. Si no entendió esto, pregunte a su instructor.
    g. Si en algún momento necesita apagar las tarjetas temporalmente, desconecte los cables **de sus extremos de la PC** (USB standard). No manipule los extremos de los cables que se conectan a las tarjetas más que para iniciar y finalizar su práctica. Cuando lo haga, recuerde hacerlo sujetando el conector y ***NO*** la tarjeta.

2. Descomprima la carpeta **`Lab6.zip`** donde prefiera.

3. Cree un proyecto en Keil para desarrollar una aplicación para el microcontrolador LPC1115 en lenguaje ensamblador, en un directorio vacío creado para ese proyecto. Póngale un nombre simple, sin espacios ni caracteres especiales, ni longitudes grandes. No olvide utilizar la extensión **`.uvproj`**
   Utilice **`startup_LPC11xx.s`**, y si lo desea, el archivo **`main.s`** proporcionado.

4. Este es el paso más importante de esta práctica. Se le recomienda llevarlo a cabo antes de presentarse al laboratorio. No le recomendamos dejar esto para el momento de su práctica, ya que si tiene dudas, su instructor probablemente estará atendiendo a sus compañeros y no le podrá ayudar tan pronto como usted lo necesite.

   Escriba código en lenguaje ensamblador para configurar el ADC, luego debe hacer un ciclo infinito para hacer una conversión y utilizar el dato convertido para controlar el LED RGB.
   Los colores del LED deben encenderse y apagarse de acuerdo al recorrido angular del potenciómetro, de la siguiente manera:

Recorrido	Rojo	Verde	Azul
0 a 1/3	Encendido	Apagado	Apagado
1/3 a 2/3	Apagado	Encendido	Apagado
2/3 a 1	Apagado	Apagado	Encendido

   Recomendaciones

   - Puede reutilizar partes del código de su práctica anterior para implementar los pasos en lenguaje ensamblador (escribir 1 ó 0 a bits específicos, etc.).
   - Su programa debe hacer uso del LED RGB, por lo que se le recomienda tener a la mano el código que utilizó anteriormente para este fin.
   - Utilice el simulador para formarse una idea del comportamiento de su programa.
   - Utilice el manual del microcontrolador para obtener las direcciones de memoria de los registros. Las necesitará para escribir su código.
   - Utilice el diagrama esquemático de la tarjeta LPCXpresso Base Board para identificar los pines GPIO a los que están conectados el LED RGB y el potenciómetro.
   - Para su conveniencia, hemos incluido un anexo que muestra lo que necesita implementar, en lenguaje C.

5. Una vez haya depurado su código, ejecute su programa y muestre a su instructor lo siguiente:
   a. El código que escribió.
   b. El hardware funcionando. Al mover el potenciómetro, el LED RGB debe pasar por sus tres colores.

6. Ahora modifique su código para no utilizar el dato convertido por el ADC. Muestre a su instructor esta aplicación corriendo. El mover el potenciómetro de extremo a extremo no debe ocasionar ningún cambio en el LED RGB. Esto lo puede lograr comentando las líneas en donde escribe a los registros de datos de los puertos paralelos.

7. Si ha llegado exitosamente hasta este paso, usted está listo para su segundo examen parcial. Si no, alístese.

Anexo
Código en C para configurar el ADC y leer una conversión

```
/*-------------------------------------------------------------------------
  Example for initialising ADC (channel 0 AD0)
  *-------------------------------------------------------------------------*/
void ADCInitialization( uint32_t ADC_Clk ){
  LPC_SYSCON->PDRUNCFG &= ~(MASK(4));       //Power to ADC
  LPC_SYSCON->SYSAHBCLKCTRL |= (MASK(13)); //Clock to ADC

  LPC_IOCON->R_PIO0_11 &= ~0x8F;  //Clear IO setting
  LPC_IOCON->R_PIO0_11 |= 0x0A;  //Analog mode and pull down the pin

  LPC_ADC->CR = (0x01<<0) | (0<<8); //Select ADC0 with CLKDIV=0x0, no division of clock
  LPC_ADC->CR &= ( ~MASK(16) ) &   //Software controlled mode
    ( ~MASK(17) ) &               //10 bits, 11 clock cycle
    ( ~MASK(24) ) ;              //Do not start
  return;
}

/*-------------------------------------------------------------------------
  Example for reading ADC value
  *-------------------------------------------------------------------------*/
uint32_t readADC(){
  uint32_t regVal, ADC_Data;
  LPC_ADC->CR |= (1 << 24) ;          //Start the conversion
  while ( !(LPC_ADC->GDR&MASK(31))); //Wait for the conversion to finish
  regVal = LPC_ADC->DR[0];          //Read conversion
  LPC_ADC->CR &= 0xF8FFFFFF;       //Stop conversion
  ADC_Data = ( regVal >> 6 ) & 0x3FF;
  return ( ADC_Data );
}
```