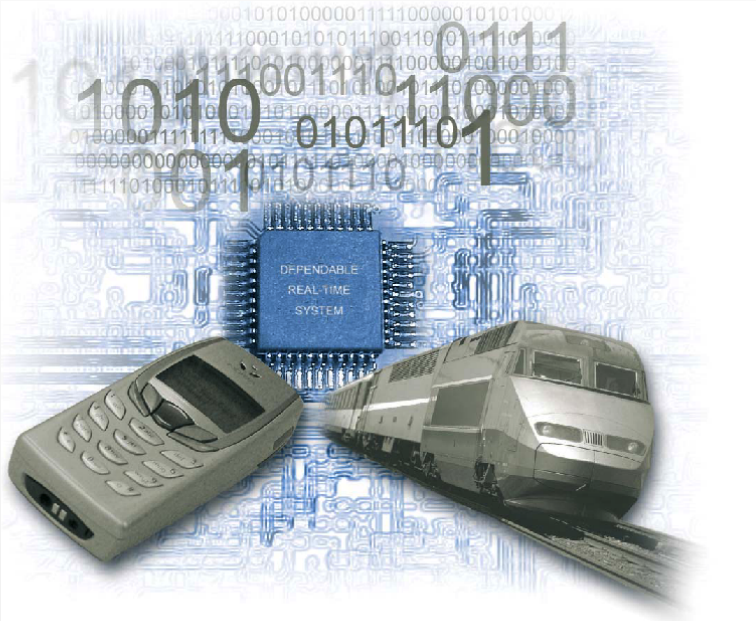
Conversión A/D D/A en LPC17xx



Departamento de Electrónica
Universidad de Alcalá

Índice

- Introducción
- Conversión A/D
- Conversión D/A

Introducción al la conversión A/D

- **¿Qué parámetros caracterizan un conversor A/D?**
  - Número de bits del conversor
  - Tipo de conversor
  - SPAN
  - Número de entradas multiplexadas.
  - Tiempo de conversión mínimo (en ciclos de reloj y en us)
  - Frecuencia de muestreo máxima
  - Errores de offset, ganancia, linealidad, …
  - Tipos de entradas (normal o diferencial)

Introducción al la conversión A/D

- **¿Cómo se controla un ADC?**
  - ¿Cómo se configura la frecuencia de reloj de control del ADC?
  - ¿Cómo se habilita el conversor?
  - ¿Cómo se da la orden de iniciar una conversión?
  - ¿Cómo se puede saber que la conversión ha finalizado?
  - ¿Dónde puedo acceder al resultado de la conversión?
  - ¿Cómo se controla el multiplexor de las entradas?
  - ¿Qué pines se pueden utilizar como entradas analógicas?
  - ¿Qué modos de entradas analógicas hay? ¿Cómo se configuran?

El conversor A/D del LPC1768

- **El LPC1768 dispone de un conversor A/D de 12 bits:**
  - Conversor de aproximaciones sucesivas
  - 8 entradas multiplexadas.
  - Frecuencia máxima de conversión de 200KHz
  - Tiempo mínimo de conversión 5us (65 ciclos de reloj del reloj de entrada que debe ser menor o igual a 13MHz)
  - SPAN de V_{REFN} a V_{REFP} (no debe superar V_{DDA})
  - Posibilidad de conversión en modo ráfaga (“Burst”)
  - Posibilidad de iniciar una conversión con la transición de un pin externo o con un módulo de comparación de un temporizador.
- **Configuración básica**
  - Se debe activar la alimentación del ADC con PCADC (PCONP[]).
  - Configurar PCLK_ADC en PCLKSEL0
  - Habilitar los pines de entrada analógica con PINSEL y PINMODE
  - Habilitar las interrupciones en el NVIC
  - Activar el DMA

Modos de funcionamiento

- **Modo manual (software controlled)**
  - El conversor debe estar activo: PDN(CR[21]) = 1
  - Configurado modo manual: BURST(CR[16]) = 0
  - Sólo puede estar seleccionado un canal de entrada en SEL(CR[7:0])
  - La conversión se activa con el recurso seleccionado en START(CR[26:24]) y EDGE(CR[27])
    - No conversión
    - Conversión manual
    - Por evento en P0.10, P1.27, MAT0.1, MAT0.3, MAT1.0, MAT1.1
- **Modo ráfaga (BURST)**
  - El conversor debe estar activo: PDN(CR[21]) = 1
  - Configurado modo ráfaga: BURST(CR[16]) = 1
  - Selección de los canales de entrada activos en SEL(CR[7:0])
  - Se desactiva el inicio de conversión manual START(CR[26:24]) = 0x00
- **Notas**
  - Mientras se está conviertiendo se ignoran nuevos inicios de conversión hasta que haya terminado.
  - Se debe seleccionar el pin de entrada analógico sin pull-up ni pull-down

Lectura del resultado de la conversión

- **Lectura de resultado en ADGDR (Global Data Register)**
  - El resultado de la última conversión en RESULT(ADGDR[15:4])
  - El canal donde se muestreó la última conversión en CHN(ADGDR[26:24])
  - Se indica si se ha robreescrito alguna conversión anterior en OVERRUN(ADGDR[30])
    - Se borra con la lectura de ADGDR
  - Se indica cuándo ha finalizado la conversión en DONE(ADGDR[31])
    - Se borra con la lectura de ADGDR
    - Puede producir interrpción si ADGINTEN(ADINTEN[8]) = 1
- **Lectura de resultado en ADDRn (Data Register of Channel n)**
  - El resultado de la última conversión del canal n en RESULT(ADDRn[15:4])
  - Se indica si se ha robreescrito alguna conversión anterior del canal n en OVERRUN(ADDRn[30])
    - Se borra con la lectura de ADDRn
  - Se indica cuándo ha finalizado la conversión del canal n en DONE(ADDRn[31])
    - Se borra con la lectura de ADDRn
    - Puede producir interrpción si ADGINTEN(ADINTEN[n]) = 1 y ADGINTEN(ADINTEN[8]) = 0

Lectura del resultado de la conversión

- **Lectura conjunta de los flags de DONE y OVERRUN en ADSTAT**
  - Se pueden leer de una vez todos los bits de DONE(ADSTAT[7:0])
  - Se pueden leer de una vez todos los bits de OVERRUN(ADSTAT[15:8])
  - El flag de interrupción del ADC (OR de todos los DONE) en ADINT(ADSTAT[16])

- **Registro de ajuste fino del conversor (ADTRIM)**
  - Contiene información de ajuste de offset interno del ADC y DAC

- **DMA**
  - Se pueden configurar transferencias directas a memoria del resultado del conversor.
  - La transferencia de DMA se inicia con el evento de interrupción (manteniendo la interrupción deshabilitada en el NVIC).
  - El conversor debe estar en modo ráfaga.
  - El tamaño de la transferencia de datos puede ser de 1, 4 u 8 palabras y debe coincidir con el tamaño de la ráfaga.

Ejemplo de código del A/D

ADC.C (cont)

```
/*-------------------------------------------------------------------------
  Function that initializes ADC
 *-------------------------------------------------------------------------*/
void ADC_Init (void) {

  LPC_SC->PCONP |= ((1 << 12) | (1 << 15));  /* enable power to ADC & IOCON   */

  LPC_PINCON->PINSEL1  &= ~( 3 << 18);
  LPC_PINCON->PINSEL1  |=  ( 1 << 18);       /* P0.25 is AD0.2                */
  LPC_PINCON->PINMODE1 &= ~( 3 << 18);
  LPC_PINCON->PINMODE1 |=  ( 2 << 18);       /* P0.25 no pull up/down         */

  LPC_ADC->ADCR        =  ( 1 <<  2) |       /* select AD0.2 pin              */
                          ( 4 <<  8) |       /* ADC clock is 25MHz/5          */
                          ( 1 << 21);        /* enable ADC                    */

#ifdef __ADC_IRQ
  LPC_ADC->ADINTEN     =  ( 1 <<  8);        /* global enable interrupt       */

  NVIC_EnableIRQ(ADC_IRQn);                  /* enable ADC Interrupt          */
#endif
}
```

Ejemplo de código del A/D

ADC.C (cont)

```
/*-----------------------------------------------------------------------------
  start AD Conversion
 *----------------------------------------------------------------------------*/
void ADC_StartCnv (void) {
  LPC_ADC->ADCR &= ~( 7 << 24);             /* stop conversion                */
  LPC_ADC->ADCR |=  ( 1 << 24);             /* start conversion               */
}


/*-----------------------------------------------------------------------------
  stop AD Conversion
 *----------------------------------------------------------------------------*/
void ADC_StopCnv (void) {

  LPC_ADC->ADCR &= ~( 7 << 24);             /* stop conversion                */
}
```

Ejemplo de código del A/D

ADC.C (cont)

```
/*-------------------------------------------------------------------------
  get converted AD value
 *-------------------------------------------------------------------------*/
uint16_t ADC_GetCnv (void) {

#ifndef __ADC_IRQ
  while (!(LPC_ADC->ADGDR & ( 1UL << 31)));  /* Wait for Conversion end       */
  AD_last = (LPC_ADC->ADGDR >> 4) & ADC_VALUE_MAX; /* Store converted value   */

  AD_done = 1;
#endif
                                          En ADC.h → #define ADC_VALUE_MAX      (0xFFF)
  return(AD_last);
}


/*-------------------------------------------------------------------------
  A/D IRQ: Executed when A/D Conversion is done
 *-------------------------------------------------------------------------*/
#ifdef __ADC_IRQ
void ADC_IRQHandler(void) {
  volatile uint32_t adstat;

  adstat = LPC_ADC->ADSTAT;                       /* Read ADC clears interrupt     */

  AD_last = (LPC_ADC->ADGDR >> 4) & ADC_VALUE_MAX; /* Store converted value   */

  AD_done = 1;
}
#endif
```

Ejemplo de código del A/D

Cabecera del ADC.C

```
#include "LPC17xx.H"                        /* LPC17xx definitions           */
#include "ADC.h"

uint16_t AD_last;                           /* Last converted value          */
uint8_t  AD_done = 0;                       /* AD conversion done flag       */
```

ADC.h

```
#ifndef __ADC_H
#define __ADC_H

#define ADC_VALUE_MAX      (0xFFF)

extern uint16_t AD_last;
extern uint8_t  AD_done;

extern void     ADC_Init    (void);
extern void     ADC_StartCnv(void);
extern void     ADC_StopCnv (void);
extern uint16_t ADC_GetCnv  (void);

#endif
```

Ejemplo de código del A/D

Ejemplo de uso sin interrupción

```
int main(void) {

   uint32_t ADC_Data;
   uint32_t ADC_Buf;
   uint8_t i;

   SystemInit();
   UART0_Init();
   ADC_Init();
   ADC_StartCnv();

   while (1) {
      ADC_Data = 0;
      for(i = 0;i < 8; i++) {
         ADC_Buf = ADC_GetCnv();
         ADC_StartCnv();
         ADC_Data += ADC_Buf;
      }

      ADC_Data = (ADC_Data / 8);                /*sampling 8 times*/
      ADC_Data = (ADC_Data * 3300)/4096;

      UART0_SendString("AD channel5 input: ");
      UART0_SendChar(ADC_Data);                 /*send data to com*/
      UART0_SendString("mV\n");

      Delay(1000);
    }
}
```

```
#include "LPC17xx.h"
#include "uart.h"
#include "adc.h"


void Delay (uint32_t Time) {
    uint32_t i;

    i = 0;
    while (Time--)
        for (i = 0; i < 5000; i++);
}
```

Parámetros analógicos

Carácterísticas eléctricas del ADC

Table 18. ADC characteristics (full resolution)

V_{DDA} = 2.7 V to 3.6 V; T_{amb} = −40 °C to +85 °C unless otherwise specified; ADC frequency 13 MHz; 12-bit resolution.

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IA}	analog input voltage			0	-	V_{DDA}	V
C_{ia}	analog input capacitance			-	-	15	pF
E_D	differential linearity error		[1][2]	-	-	±1	LSB
$E_{L(adj)}$	integral non-linearity		[3]	-	-	±3	LSB
E_O	offset error		[4][5]	-	-	±2	LSB
E_G	gain error		[6]	-	-	0.5	%
E_T	absolute error		[7]	-	-	4	LSB
R_{vsi}	voltage source interface resistance		[8]	-	-	7.5	kΩ
$f_{clk(ADC)}$	ADC clock frequency			-	-	13	MHz
$f_{c(ADC)}$	ADC conversion frequency		[9]	-	-	200	kHz

[1] The ADC is monotonic, there are no missing codes.

[2] The differential linearity error (E_D) is the difference between the actual step width and the ideal step width. See Figure 26.

[3] The integral non-linearity ($E_{L(adj)}$) is the peak difference between the center of the steps of the actual and the ideal transfer curve after appropriate adjustment of gain and offset errors. See Figure 26.

[4] The offset error (E_O) is the absolute difference between the straight line which fits the actual curve and the straight line which fits the ideal curve. See Figure 26.

[5] ADCOFFS value (bits 7:4) = 2 in the ADTRM register. See *LPC17xx user manual UM10360*.

[6] The gain error (E_G) is the relative difference in percent between the straight line fitting the actual transfer curve after removing offset error, and the straight line which fits the ideal transfer curve. See Figure 26.

[7] The absolute error (E_T) is the maximum difference between the center of the steps of the actual transfer curve of the non-calibrated ADC and the ideal transfer curve. See Figure 26.

[8] See Figure 27.

[9] The conversion frequency corresponds to the number of samples per second.

Parámetros analógicos

Carácterísticas eléctricas del ADC

Table 19. ADC characteristics (lower resolution)

T_{amb} = –40 °C to +85 °C unless otherwise specified; 12-bit ADC used as 10-bit resolution ADC.

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
E_D	differential linearity error		[1][2]	-	±1	-	LSB
$E_{L(adj)}$	integral non-linearity		[3]	-	±1.5	-	LSB
E_O	offset error		[4]	-	±2	-	LSB
E_G	gain error		[5]	-	±2	-	LSB
$f_{clk(ADC)}$	ADC clock frequency	$3.0\ V \le V_{DDA} \le 3.6\ V$		-	-	33	MHz
		$2.7\ V \le V_{DDA} < 3.0\ V$		-	-	25	MHz
$f_{c(ADC)}$	ADC conversion frequency	$3\ V \le V_{DDA} \le 3.6\ V$	[6]	-	-	500	kHz
		$2.7\ V \le V_{DDA} < 3.0\ V$	[6]	-	-	400	kHz

[1] The ADC is monotonic, there are no missing codes.

[2] The differential linearity error (E_D) is the difference between the actual step width and the ideal step width. See Figure 26.

[3] The integral non-linearity ($E_{L(adj)}$) is the peak difference between the center of the steps of the actual and the ideal transfer curve after appropriate adjustment of gain and offset errors. See Figure 26.

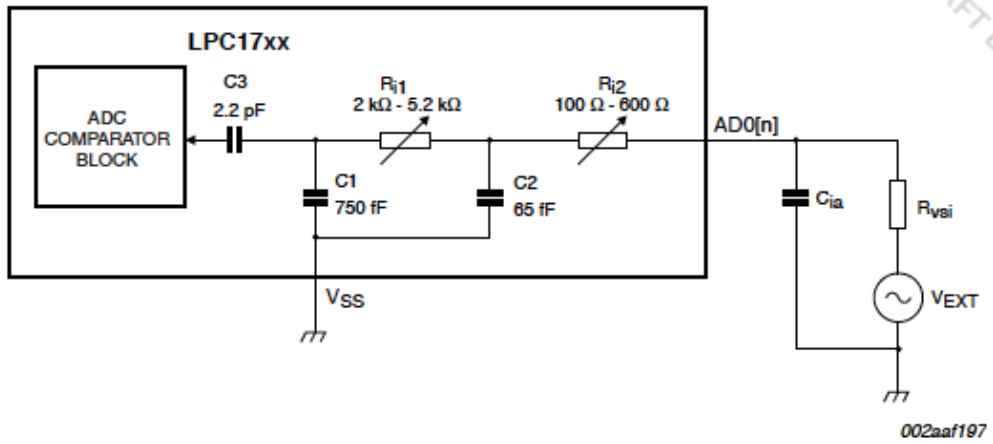
[4] The offset error (E_O) is the absolute difference between the straight line which fits the actual curve and the straight line which fits the ideal curve. See Figure 26.

[5] The gain error (E_G) is the relative difference in percent between the straight line fitting the actual transfer curve after removing offset error, and the straight line which fits the ideal transfer curve. See Figure 26.

[6] The conversion frequency corresponds to the number of samples per second.

Parámetros analógicos

Carácterísticas eléctricas del ADC



The values of resistor components R_{i1} and R_{i2} vary with temperature and input voltage and are process-dependent (see Table 20).

Parasitic resistance and capacitance from the pad are not included in this figure.

Fig 27. ADC interface to pins AD0[n]

Table 20. ADC interface components

Component	Range	Description
R_{i1}	2 kΩ to 5.2 kΩ	Switch-on resistance for channel selection switch. Varies with temperature, input voltage, and process.
R_{i2}	100 Ω to 600 Ω	Switch-on resistance for the comparator input switch. Varies with temperature, input voltage, and process.
C1	750 fF	Parasitic capacitance from the ADC block level.
C2	65 fF	Parasitic capacitance from the ADC block level.
C3	2.2 pF	Sampling capacitor.

Conversor D/A

Carácterísticas del DAC del LPC17xx

- Conversor DAC de 10 bits
- Basado en una cadena de resistencias.
- Salida de baja impedancia desacoplada (buffered output).
- Frecuencia máxima de muestreo de 1MHz
- Posible configuración de la velocidad.

- Se habilita con la asignación de la salida en PINSEL (P0.26), no con PCONP.
- Frecuencia de salida
  - Se escribe el dato en VALUE (DACR[15:6])
  - La salida después del tiempo de establecimiento toma el valor:
    - VALUE × ((V_{REFP} - V_{REFN})/1024) + V_{REFN}.
  - El tiempo de establecimiento se puede configurar con BIAS (DACR[16]).
    - 0 → 1us max. 700 uA max. Frecuencia máx 1MHz
    - 1 → 2,5 us max, 350 uA max. Frecuencia max 400KHz

Parámetros analógicos

- **Carácterísticas eléctricas del DAC**

Table 21. DAC electrical characteristics

V_{DDA} = 2.7 V to 3.6 V; T_{amb} = –40 °C to +85 °C unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
E_D	differential linearity error		-	±1	-	LSB
$E_{L(adj)}$	integral non-linearity		-	±1.5	-	LSB
E_O	offset error		-	0.6	-	%
E_G	gain error		-	0.6	-	%
C_L	load capacitance		-	200	-	pF
R_L	load resistance		1	-	-	kΩ

Referencias

- **Manual LPC17xx**
- **Ejemplos de programación de Keil.**