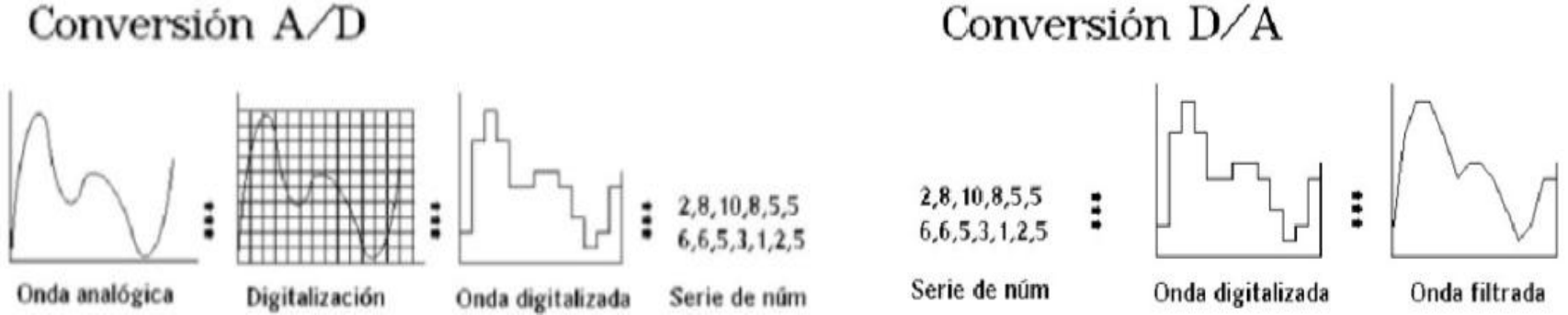
Entrada y Salida

Señales Analógicas y Digitales

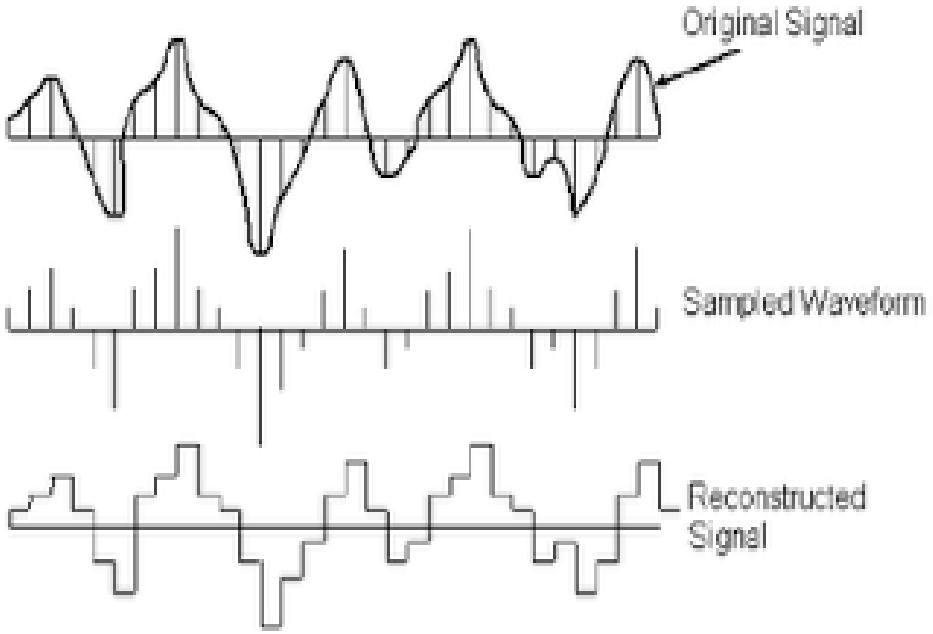
Conceptos Previos

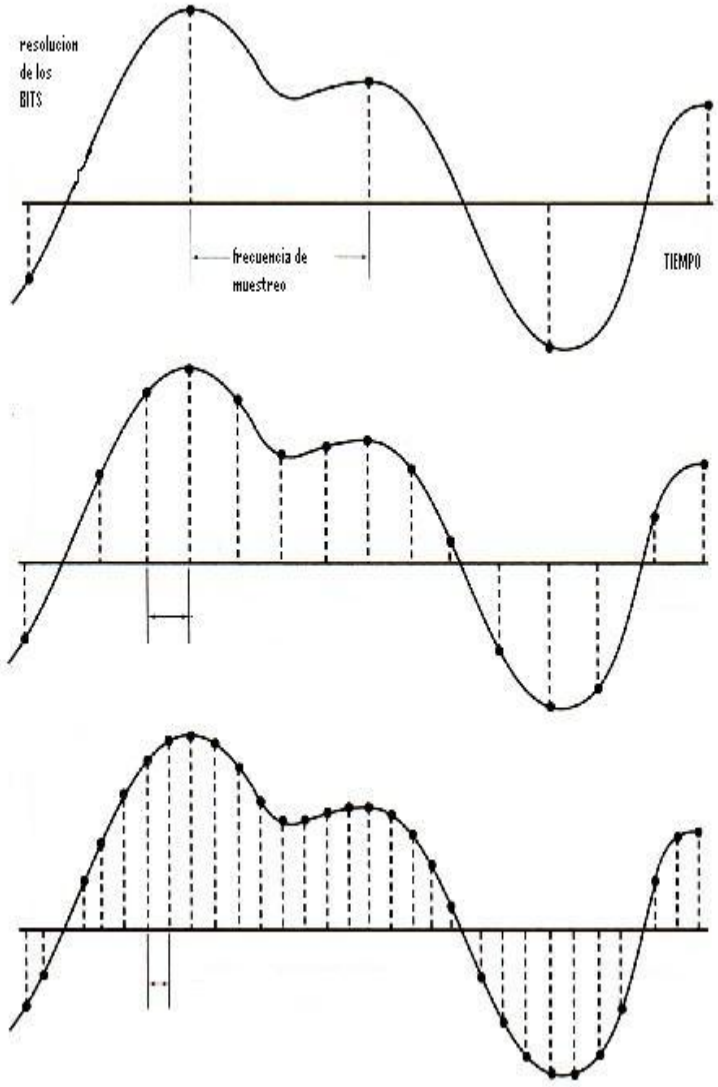
- Los conversores ADC y DAC sirven para interactuar con el mundo exterior
  - En el exterior la información es analógica
  - Se convierte a digital
  - Se procesa en digital
  - Se convierte a analógica
  - Se entrega al mundo exterior como información analógica (luz, sonido, imagen, etc.)
- El proceso de conversión analógica a digital se basa en:
  - Discretizar el eje de tiempos o espacio (muestrear)
  - Discretizar el eje de amplitud (cuantificar)
  - Codificar
- El proceso de conversión digital a analógica radica en hacer lo inverso



Conceptos Previos

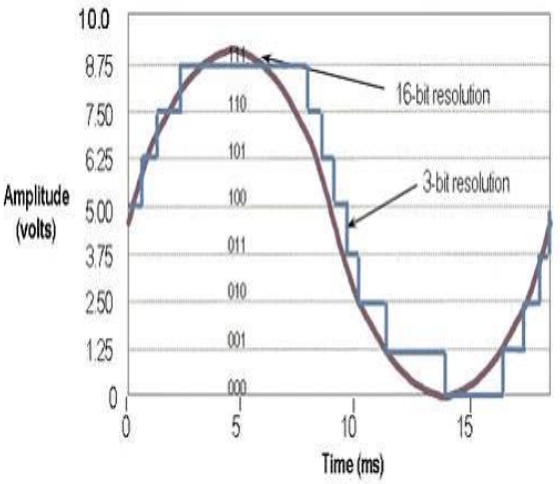
- **Muestreo:** Discretizo el eje de tiempos o espacio
  - Para no perder información hay que respetar el teorema de Nyquist ($f_{muestreo} > 2*f_{superior}$)

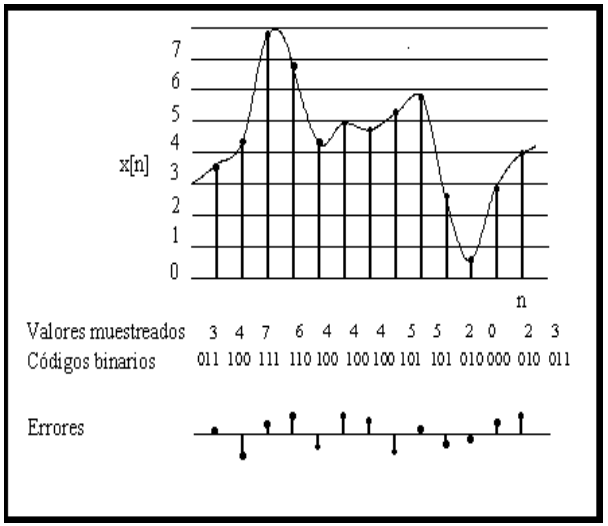




Conceptos Previos

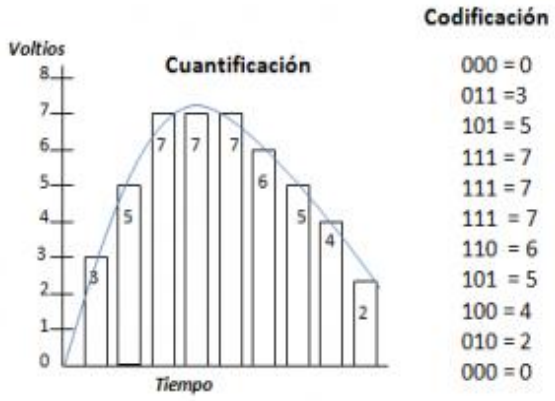
- **Cuantificación:** Discretizar el eje de amplitud
  - El número de niveles depende del tamaño de palabra (si la palabra es de 8 bits -> 256 niveles de amplitud, etc…)
  - Los niveles pueden estar equiespaciados (escala lineal), o no (escalas logarítmicas, exponenciales, etc.) dependiendo de la aplicación
  - Se introduce el ***error de cuantificación***: diferencia entre el valor real de la señal y el valor cuantificado
    - Esto hace que valores distintos de amplitud que se convierten en un mismo valor digital





Conceptos Previos

- **Codificación:**
  - Se trata de la asignación de valores binarios a cada uno de los niveles de cuantificación
  - Se puede realizar en varios pasos
    - Sacar el código de una muestra
    - Sacar el código de un conjunto de muestras, mediante la relación entre ellas
  - Tradicionalmente un ADC codifica inicialmente en modo binario, y luego el procesador decide codificar de alguna forma más óptima (atendiendo a la aplicación)



Tema 5: Conversión Analógica / Digital

Sistemas Digitales basados en Microprocesador

Grado en Ingeniería Telemática

© Raúl Sánchez Reíllo

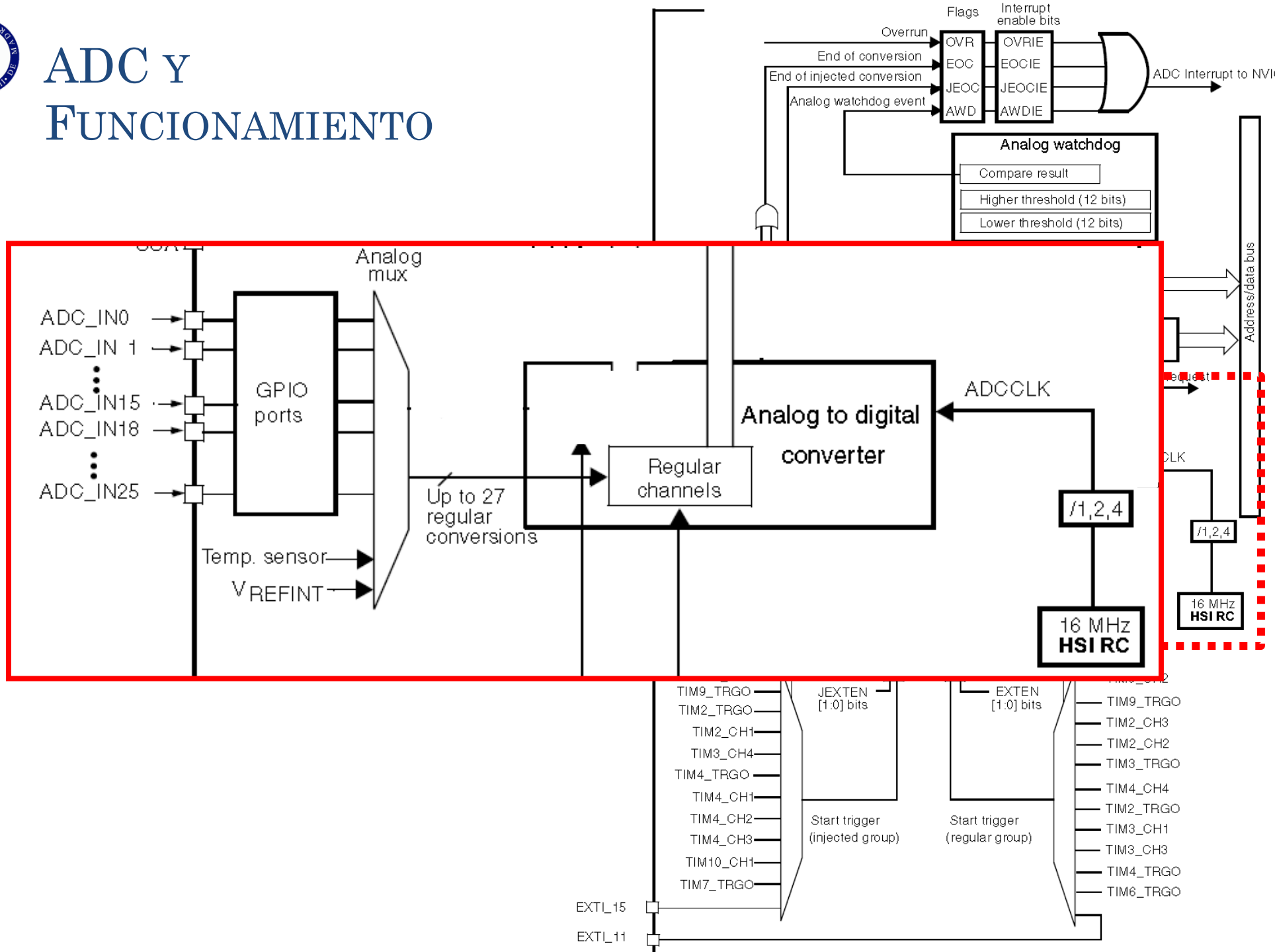
ÍNDICE

- Conversor A/D y Funcionamiento
- ADC: Registros de Control
- ADC: Registros de Datos
- ADC: Registros de Estado
- Ejemplo de Conversión Simple
- Ejemplo de Conversión Continua
- Ejercicios

ADC y Funcionamiento

- El STM32L152RB tiene un único ADC de 12 bits con entrada multiplexada entre 24 posibles fuentes externas y 2 internas.
- Sus características principales son:
  - Resolución configurable a 12, 10, 8 o 6 bits
  - Capaz de generar avisos al finalizar las conversiones
  - Modo de conversión simple o continua
  - Conversión programable para escanear varios canales de forma cíclica
  - Reloj del conversor procedente directamente del HSI (a 16MHz)
  - Posibilidad de introducir retardos entre conversiones
  - Las tensiones a convertir dependen de los valores de dos pines de entrada (Vref+ y Vref-)
    - En nuestro caso será entre 3,3V y 0V

ADC y Funcionamiento



ADC y Funcionamiento

- **Conversión simple:**
  1. Se configura el ADC
  2. Se enciende el ADC (`ADON=1`)
  3. Se activa el `SWSTART` para arrancar la conversión
  4. Se espera a que haya acabado la conversión (bit `EOC`)
  5. Se toma el dato del `ADC1->DR`
  6. Si se quiere repetir el proceso, se vuelve al punto 3
- **Conversión continua:**
  1. Se configura el ADC
  2. Se enciende el ADC (`ADON=1`)
  3. Se activa el `SWSTART` para arrancar la conversión
  4. Se va consultando el valor del `ADC1->DR` según sea conveniente, para obtener el valor actual
- **Conversión en modo scan**
  - Idéntico a cualquiera de los dos casos anteriores, pero escribiendo la secuencia de canales en los registros `ADC1->SQRx` durante la configuración del ADC

ADC: Registros de Control

- ADC→CR1 – Control Register 1:
  - Registro de 32 bits con los siguientes bits de configuración, que deben escribirse sólo cuando ADON=0:
    - OVRIE: Habilitación de interrupción por overrun. No lo usaremos.
    - RES[1:0]: Resolución. Siempre elegiremos 12 bits.
      - 00 – 12bits; 01 – 10bits; 10 – 8bits; 11 – 6bits
    - AWDEN, JAWDEN, PDI, PDD, DISCNUM, JDISCEN, DISCEN, JAUTO, AWDSGL – todos los bits a '0'
    - SCAN: Scan mode. Siempre lo configuraremos deshabilitado.
      - 0 – deshabilitado; 1 – habilitado
    - JEOCIE, AWDIE – todos los bits a '0'
    - EOCIE: Habilitación de interrupción por fin de conversión. No lo usaremos
    - AWDCH[4:0] – todos los bits a '0'

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					OVRIE	RES[1:0]		AWDEN	JAWDEN	Reserved				PDI	PDD
					rw	rw	rw	rw	rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISCNUM[2:0]			JDISCEN	DISCEN	JAUTO	AWDSGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

ADC: Registros de Control

- ADC→CR2 – Control Register 2:
  - Registro de 32 bits, con los siguientes bits de configuración , que deben escribirse sólo cuando ADON=0 (salvo indicado en contra):
    - SWSTART – Escribiendo un '1' aquí inicia una conversión (el propio hardware lo pone a '0' automáticamente)
      - Este bit sólo se puede activar con ADON=1 y RCNR=0
    - EXTEN, EXTSEL[3:0], JSWSTART, JEXTEN, JEXTSEL[3:0] – todos a '0'
    - ALIGN – Con un '0' alinea el dato a la derecha del registro de 16bits, y con un '1' lo alinea a la izquierda. Siempre lo pondremos a la derecha.
    - EOCS – Selección del modo de aviso del EOC
      - Con un '0' sólo se activa el EOC al finalizar una secuencia completa de conversión (modo scan). Con un '1' se activa con cada conversión. Siempre lo pondremos a "1".
    - DDS, DMA – todos los bits a '0'
    - DELS – Configuración del retardo entre conversiones:
      - 000 – Sin retardo; 001 – Hasta que se lea el dato anterior; 010 – 111 retardos de 7, 15, 31, 63, 127 y 255 ciclos de APB. Lo pondremos 000 (simple) o 001 (continua).
    - CONT – Con un '0' la conversión es simple; con un '1' la conversión es continua.
    - ADON – Con un '1' enciende el ADC, con un '0' lo apaga.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWSTART	EXTEN		EXTSEL[3:0]				Reserved	JSWSTART	JEXTEN		JEXTSEL[3:0]			
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ALIGN	EOCS	DDS	DMA	Res.	DELS			Reserved		CONT	ADON
				rw	rw	rw	rw		rw	rw	rw			rw	rw

ADC: Registros de Control

- GPIOx→SMPRx – Sample time register x (no lo vamos a usar)
  - Conjunto de 3 registros de 32 bits que indican la selección de tiempo de muestreo para cada uno de los 26 canales
    - SMPR1 – canales 20 a 25
    - SMPR2 – canales 10 a 19
    - SMPR3 – canales 0 a 9
  - Para cada canal, se usan 3 bits:
    - 000 – 4 ciclos; 001 – 9; 010 – 16; 011 – 24; 100 – 48; 101 – 96; 110 – 192; 111 – 384
  - Estos registros sólo se deben escribir con ADON=0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP5[0]		SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

ADC: Registros de Control

- ADC→SQRx – Sequence Register x:
  - Conjunto de 5 registros (el 1 es distinto al resto) que configuran el número de canales a convertir y el orden de los mismos
    - Se puede repetir canal en la secuencia
    - El máximo son 27 pasos en la secuencia
  - En los bits 24-20 del SQR1 hay que definir el número de elementos de la secuencia
    - 00000 – 1 elemento; 00001 – 2 elementos; …; 11010 – 27 elementos
    - Cada secuencia se indica con su número (del 0 al 25) codificado en 5 bits
    - Si sólo se usa un elemento, hay que escribir el SQR1 = 0 y en los 5 bits más bajos del SQR5 el canal utilizado.

SQR1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							L[4:0]					Reserved			
							rw	rw	rw	rw	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserv-ed	SQ27[4:0]					SQ26[4:0]					SQ25[4:0]				
	rw	rw	rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

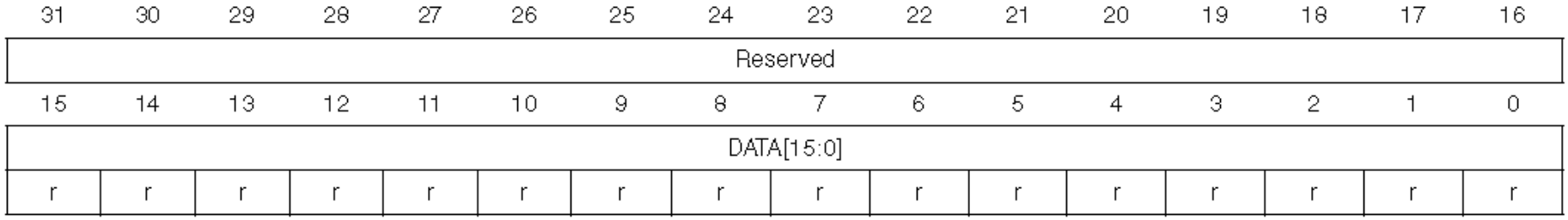
SQR5

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SQ6[4:0]					SQ5[4:0]					SQ4[4:1]			
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ4_0	SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

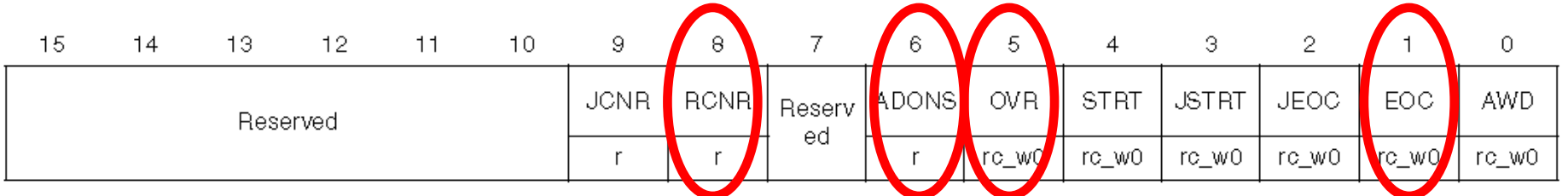
ADC: Registros de Datos

- ADC→DR – ADC Data Register:
  - Registro de 32 bits, con solo 16 útiles (los menos significativos), donde se deposita el dato alineado a la izquierda o a la derecha, según se haya seleccionado en ALIGN



ADC: Registros de Estado

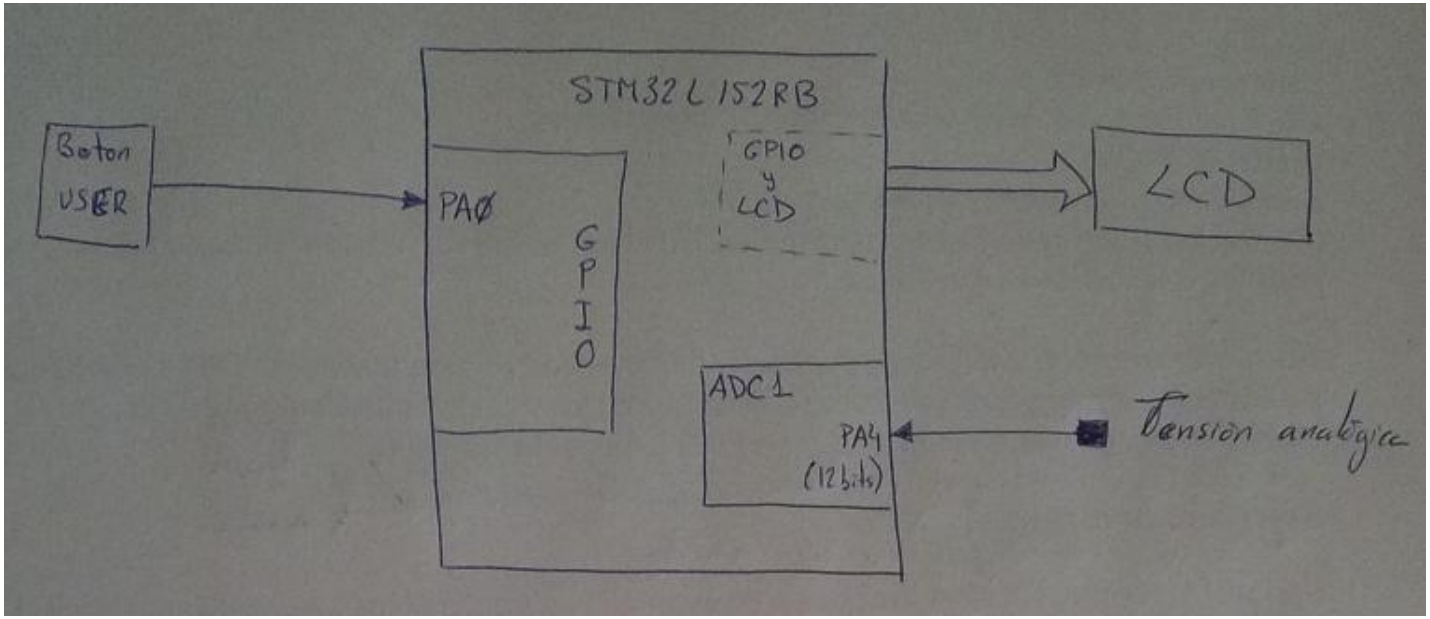
- ADC→SR – Status Register
  - Registro de 32 bits con sólo 10 disponibles y sólo 4 útiles para el curso:
    - RNCR – Regular channel not ready
      - Indica con un 1 que el canal de conversión NO está disponible, por lo que no se debe activar el `SWSTART`. No lo usaremos.
    - ADONS – ADC ON status
      - Indica con un 1 que el ADC está listo para convertir.
    - OVR – Overrun
      - Indica con un 1 que no se ha leído previamente el resultado de la conversión anterior, y se ha sobreescrito su valor con el resultado de la conversión actual. No lo usaremos.
    - EOC – End of Conversion
      - Indica con un 1 que se ha finalizado la conversión en curso y el resultado está en `ADC->DR`. Lo usaremos en conversiones simples.



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						JCNR	RCNR	Reserved	ADONS	OVR	STRT	JSTRT	JEOC	EOC	AWD
						r	r		r	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

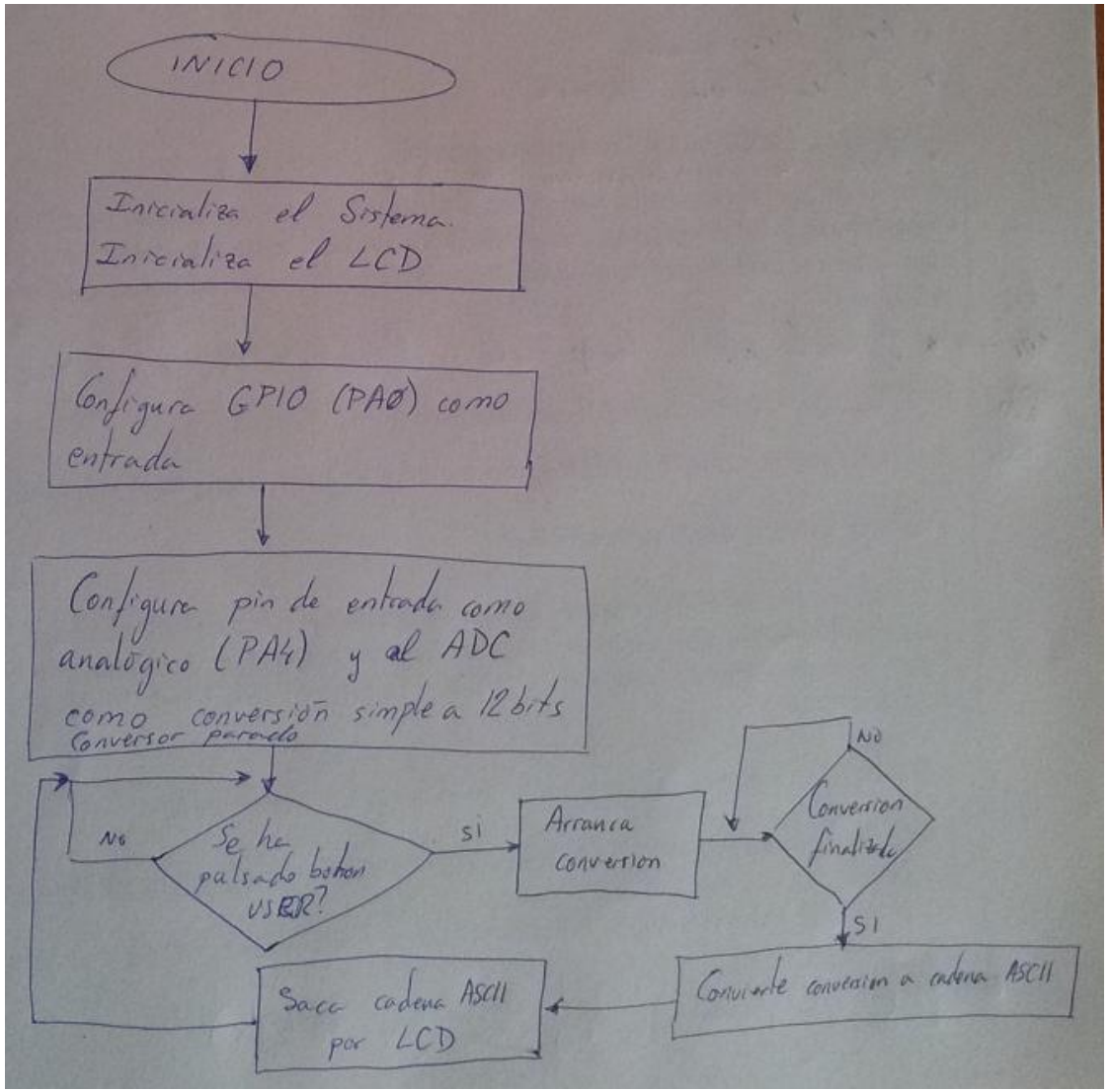
Ejemplo de Uso de Conversión Simple

- El siguiente ejemplo convierte cada vez que se pulsa el botón USER y saca el valor de conversión (de 12 bits, es decir, de 0-4096) por el LCD
- El diagrama de bloques sería:
  - Tenga en cuenta que el diagrama de bloques debe servir para hacer una representación gráfica que resuma el enunciado del problema:
    - Las flechas tienen sentido de interacción
    - Incluso aparecen los pines en los que están conectados cada uno de los dispositivos externos.
    - Están especificados los periféricos del microcontrolador a utilizar.
    - Se pueden añadir más informaciones que resuman el enunciado (por ejemplo, los 12 bits)



Ejemplo de Uso de Conversión Simple

- Y el diagrama de flujo:
  - No se hace referencia directa al microcontrolador o a datos específicos del periférico
  - Se puede llegar a poner alguna información adicional (que se podría quitar sin eliminar el sentido al diagrama de flujo), como por ejemplo el pin usado.
  - Las conexiones entre módulos deben estar realizados con flechas unidireccionales (es decir, con puntas de flecha indicando dirección)



Ejemplo de Uso de Conversión Simple (1)

- El siguiente ejemplo convierte cada vez que se pulsa el botón USER y saca el valor de conversión (de 12 bits, es decir, de 0-4096) por el LCD

```
#include "stm32l1xx.h"
#include "Biblioteca_SDM.h"
#include "Utiles_SDM.h"

int main(void){

   unsigned short valor = 0;
   unsigned char texto[6];
   Init_SDM();
   Init_LCD();

   // Configuración del botón USER (PA0)
   // PA0 como entrada (00)
   GPIOA->MODER &= ~(1 << (0*2 +1));
   GPIOA->MODER &= ~(1 << (0*2));

   // PA0 sin pull-up, pull-down (00)
   GPIOA->PUPDR &= ~(11 << (0*2));

   // Configuración del ADC
   GPIOA->MODER |= 0x00000300;        // PA4 como analógico
   ADC1->CR2 &= ~(0x00000001);        // ADON = 0 (ADC apagado)
   ADC1->CR1 = 0x00000000;            // OVRIE = 0 (deshabilitada la habilitación por interrupción)
                                      // RES = 00 (resolución = 12 bits)
                                      // SCAN = 0 (modo scan deshabilitado)
                                      // EOCIE = 0 (deshabilitada la interrupción por EOC)
   ADC1->CR2 = 0x00000400;            // EOCS = 1 (activado el bit EOC al acabar cada conversión)
                                      // DELS = 000 (sin retardo en la conversión)
                                      // CONT = 0 (conversión simple)
   ADC1->SMPR1 = 0;                   // Sin sampling time (4 cycles)
   ADC1->SMPR2 = 0;
   ADC1->SMPR3 = 0;
   ADC1->SQR1 = 0x00000000;           // 1 elemento solo en la secuencia
   ADC1->SQR5 = 0x00000004;           // El elemento es el canal AIN4
   ADC1->CR2 |= 0x00000001;           // ADON = 1 (ADC activado)
```

Ejemplo de Uso de Conversión Simple (2)

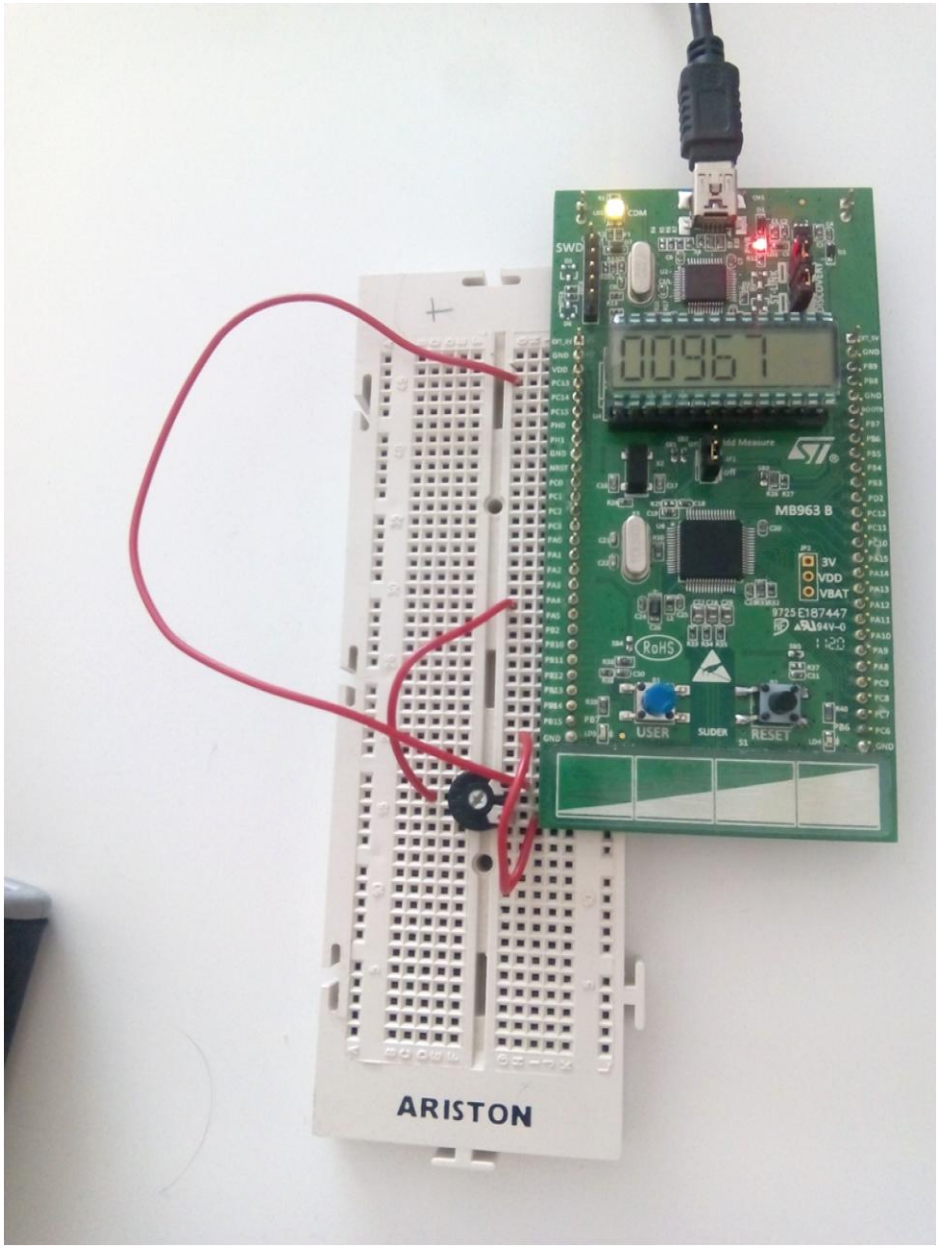
- El siguiente ejemplo convierte cada vez que se pulsa el botón USER y saca el valor de conversión (de 12 bits, es decir, de 0-4096) por el LCD

```
while (1) {
      if ((GPIOA->IDR&0x00000001)!=0) // Si PA0 = 1 (pulsador pulsado)
                                      // hago conversión, si no termino
      {
         while ((GPIOA->IDR&0x00000001)!=0)  // Si PA0 = 1 (pulsador pulsado),
                                             // espero para evitar rebotes
            {
            espera(70000);
            }
         // Arranca conversión
         while ((ADC1->SR&0x0040)==0); // Mientras ADONS = 0, o sea, el ADC
                                       // no está listo para convertir, espero
         ADC1->CR2 |= 0x40000000; // Cuando ADONS = 1 y arranco la conversión
                                  // (SWSTART = 1)
         // Espera a conversión finalizada
         while ((ADC1->SR&0x0002)==0);  // Si EOC = 0, o sea, no he acabado
                                        // con la conversión, espero
         valor = ADC1->DR;  // Cuando EOC = 1, cojo el valor convertido
                            // y lo guardo en la variable valor

// Convierte la conversión a texto
         Bin2Ascii(valor,&texto[0]);

         // Saca conversión por LCD
         LCD_Texto(texto);
         }
      }
   }
```

PRUEBA DEL EJEMPLO EXPLICADO



Ejemplo de Uso de Conversión Continua (1)

- El siguiente ejemplo convierte de forma continua y saca el valor de conversión (de 12 bits, es decir, de 0-4096) por el LCD

```
#include "stm32l1xx.h"
#include "..\Biblioteca_SDM.h"
#include "..\Utiles_SDM.h"

int main(void){

            unsigned short valor = 0;
   unsigned char texto[6];
   Init_SDM();
   Init_LCD();

   // Configuración ADC
   GPIOA->MODER |= 0x00000300;     // PA4 como analógico
   ADC1->CR2 &= ~(0x00000001);     // ADON = 0 (ADC apagado)
   ADC1->CR1 = 0x00000000;         // OVRIE = 0 (deshabilitada la habilitación por interrupción)
                                   // RES = 00 (resolución = 12 bits)
                                   // SCAN = 0 (modo scan deshabilitado)
                                   // EOCIE = 0 (deshabilitada la interupción por EOC)
                                   // OVRIE = 0 (deshabilitada la habilitación por interrupción)
   ADC1->CR2 = 0x00000412;         // EOCS = 1 (activado el bit EOC al acabar cada conversión)
                                   // DELS = 001 (retardo de la conversión hasta que se lea el dato anterior)
                                   // CONT = 1 (conversión continua)
   ADC1->SMPR1 = 0;                // Sin sampling time (4 cycles)
   ADC1->SMPR2 = 0;
   ADC1->SMPR3 = 0;
   ADC1->SQR1 = 0x00000000;        // 1 elemento solo en la secuencia
   ADC1->SQR5 = 0x00000004;        // 1 elemento es el canal AIN4
   ADC1->CR2 |= 0x00000001;        // ADON = 1 (ADC activado)

   while ((ADC1->SR&0x0040)==0);  // Si ADCONS = 0, o sea no estoy para convertir, espero

   ADC1->CR2 |= 0x40000000;        // Cuando ADCONS = 1, arranco la conversión (SWSTART = 1)
```

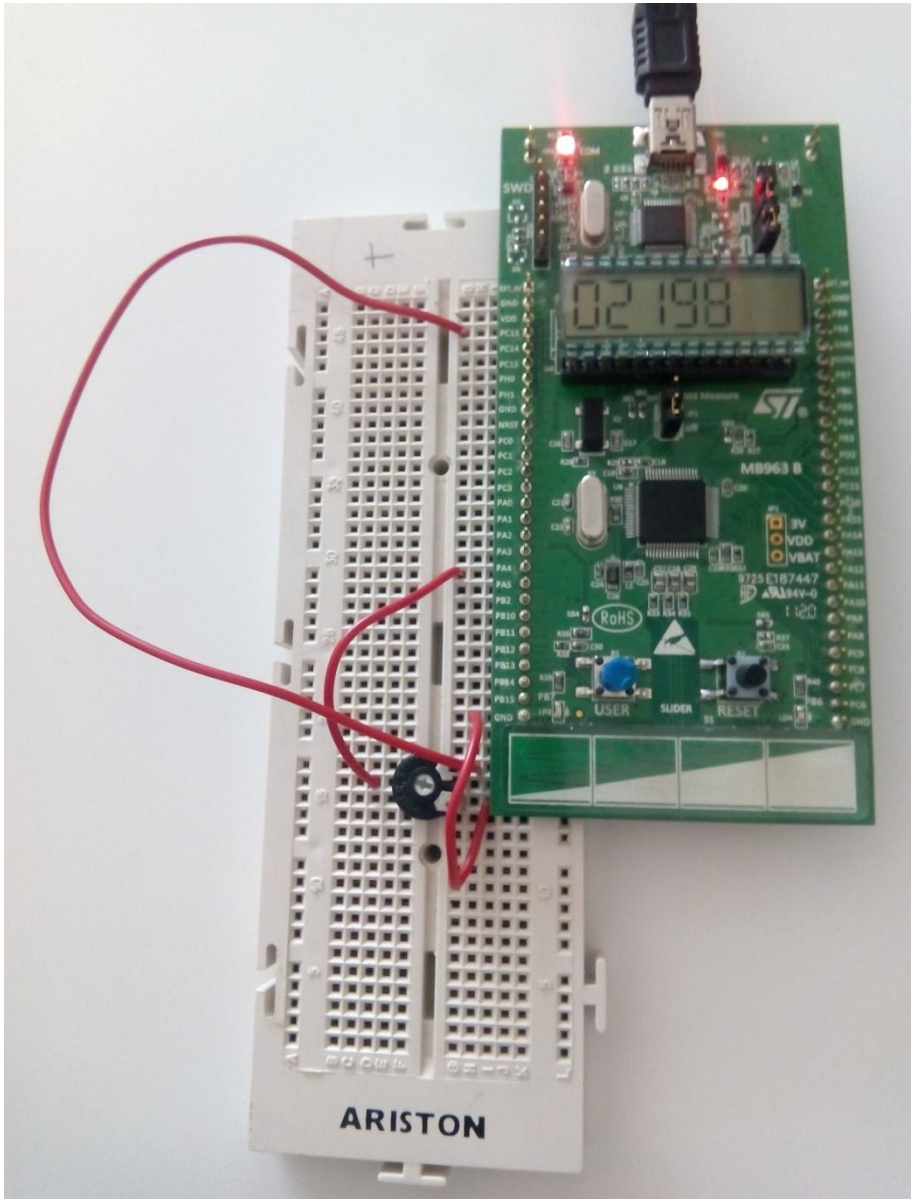
Ejemplo de Uso de Conversión Continua (2)

- El siguiente ejemplo convierte de forma continua y saca el valor de conversión (de 12 bits, es decir, de 0-4096) por el LCD

```
while (1) {
     valor = ADC1->DR; // Cojo el valor convertido y lo
                       // guardo en la variable valor
     // Convierte conversión a texto
     Bin2Ascii(valor,&texto[0]);

     // Saca conversión por LCD
     LCD_Texto(texto);
     }
  }
```

PRUEBA DEL EJEMPLO EXPLICADO



EJERCICIOS

- 1) Análisis de los ejemplos: Realice el diagrama de flujo del segundo ejemplo, cree un proyecto para cada uno y escriba el código fuente comentando cada línea y/o grupo funcional. Complete los ejemplos añadiendo la función que hace falta para convertir de binarioa ASCII y sacar los valores por el LCD y finalmente ejecútelos y pruébelos con el depurador.
- 2) Cambie uno de los ejemplos para que convierta en 8 bits de resolución en lugar de 12.
  - ¿Qué diferencias aprecia al ejecutarlos?.
- 3) Con la resolución que desee, modifique el programa, para que, en lugar de aparecer en el LCD el valor de la conversión realizada, aparezca el valor del voltaje convertido.
- 4) Haga un programa que convierta continuamente de dos canales, y que según se pulse el botón, se muestre uno u otro canal (pero continuando la conversión con el otro también).