CC BY NC SA

uc3m

Departamento Tecnología Electrónica

dte UC3M

Tema 6: Conversión A/D y D/A

Sistemas Digitales Basados en Microprocesadores

Universidad Carlos III de Madrid

Dpto. Tecnología Electrónica

Nota: Las figuras utilizadas para ilustrar las características y funcionalidades del microcontrolador del curso se han obtenido de la documentación técnica disponible en https://www.st.com/en/microcontrollers-microprocessors/stm32l151-152.html

Índice

CC BY NC SA

CC BY NC SA

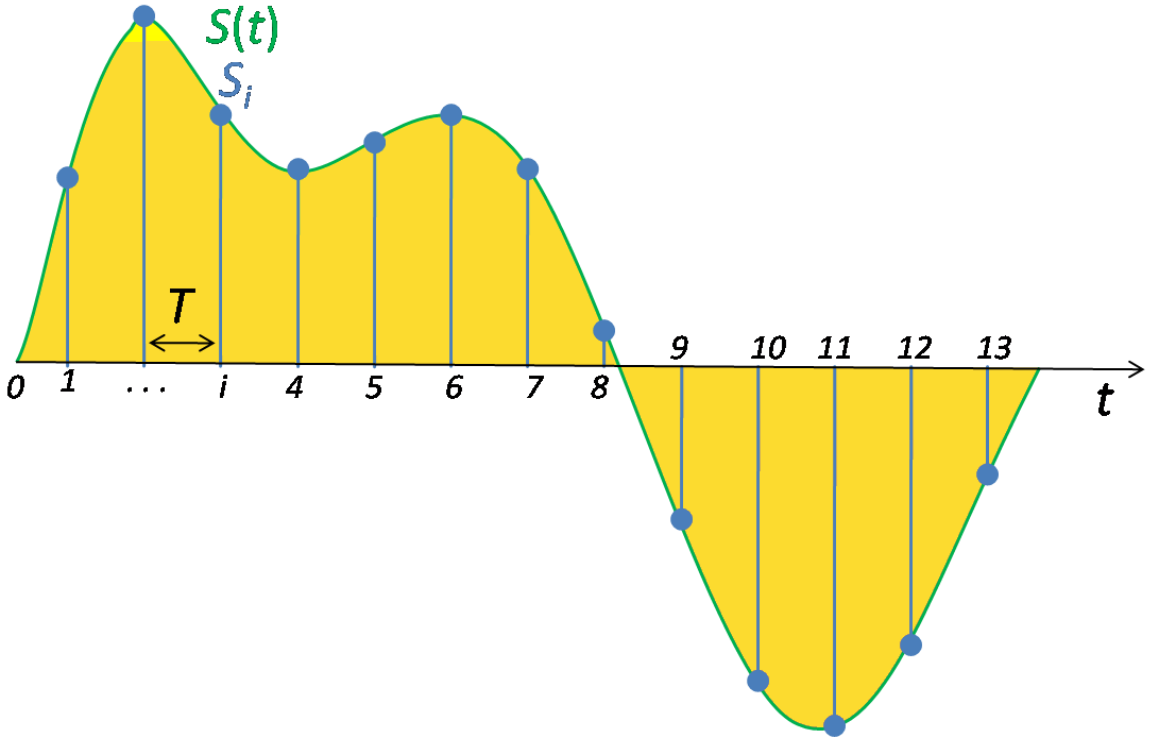
Conceptos Previos

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

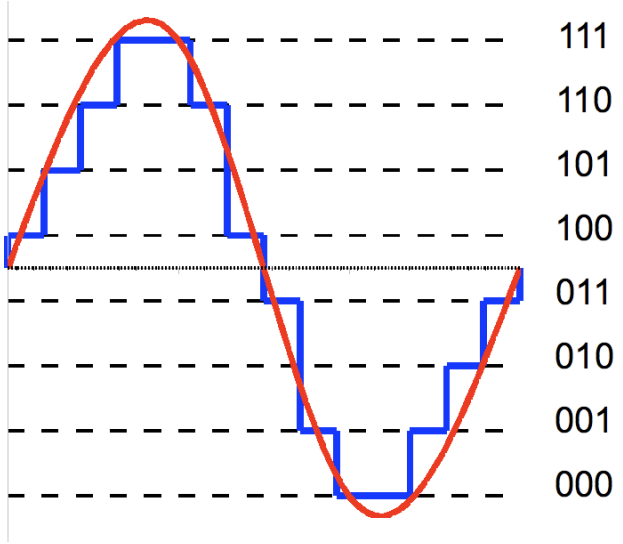
- **Muestreo:** Discretiza el eje de tiempos o espacio
  - Para no perder información hay que respetar el teorema de Nyquist ($f_{muestreo} > 2*f_{maxima}$)



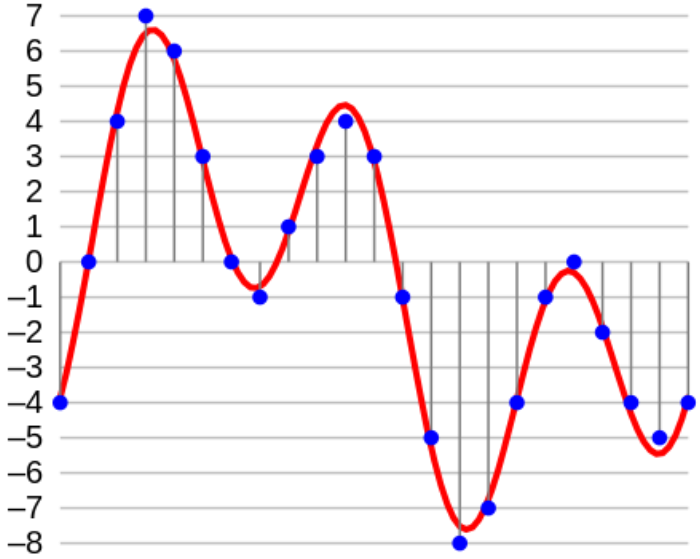
https://upload.wikimedia.org/wikipedia/commons/5/50/Signal_Sampling.png

Conceptos Previos

- **Cuantificación:** Discretiza 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



https://upload.wikimedia.org/wikipedia/commons/b/b7/3-bit_resolution_analog_comparison.png



https://upload.wikimedia.org/wikipedia/commons/thumb/2/21/4-bit-linear-PCM.svg/500px-4-bit-linear-PCM.svg.png

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)

CC BY NC SA

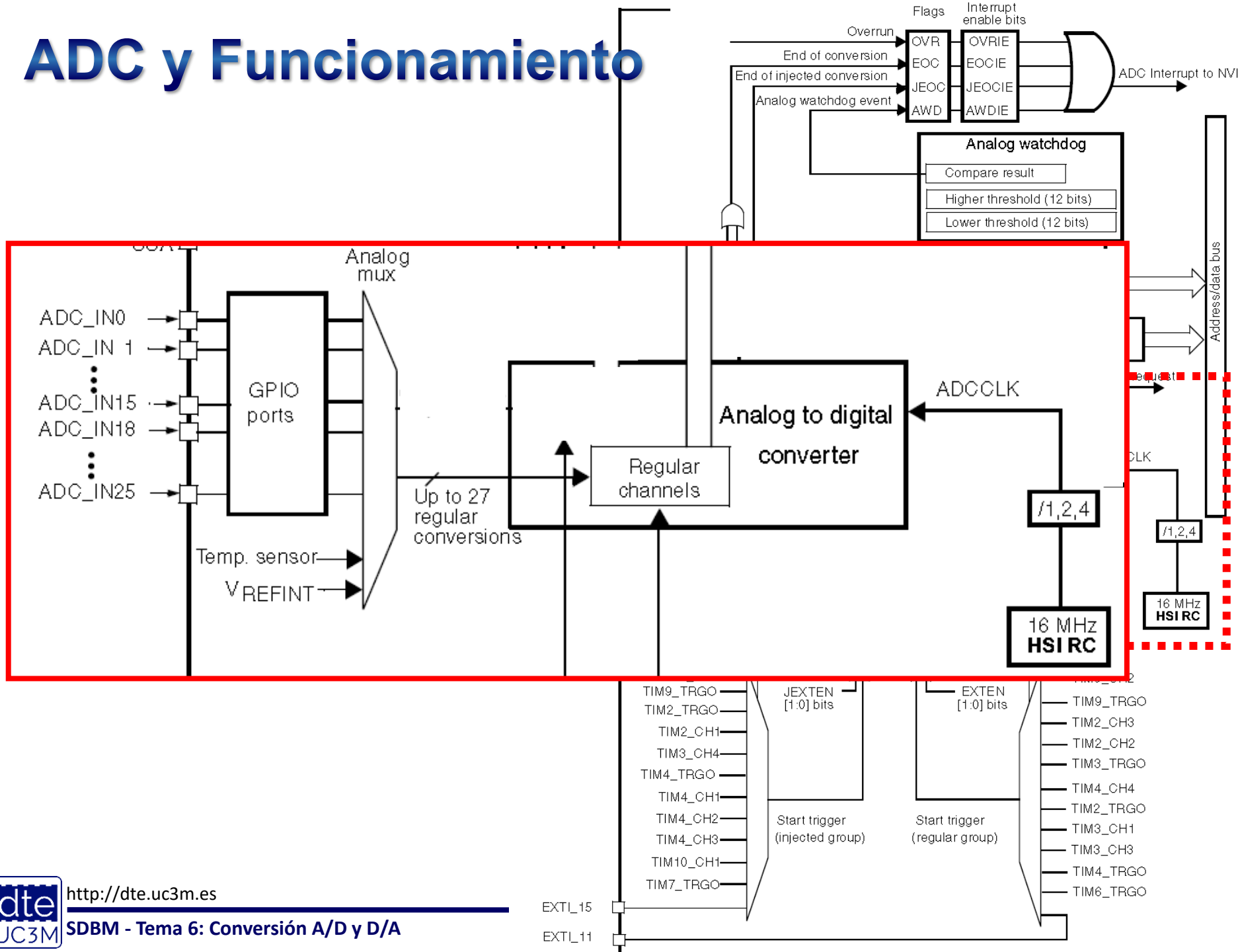
Conversor A/D

ADC y Funcionamiento

cc BY NC SA

- El STM32L152 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

cc BY NC SA

- **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 convirtiendo varios canales de forma secuencial. Para esto hay que escribir 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 en este curso, pon '0'.***
    - RES[1:0]: Resolución.
      - 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.
      - 0 – deshabilitado; 1 – habilitado
    - JEOCIE, AWDIE – todos los bits a '0'
    - EOCIE: Habilitación de interrupción por fin de conversión. Si no se quiere usar, pon '0'
    - 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. Normalmente alinearemos 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.
    - 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. Normalmente lo pondremos a 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

- 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 elemento se indica con su número (del 0 al 25) codificado en 5 bits, indicando el número de canal
    - 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

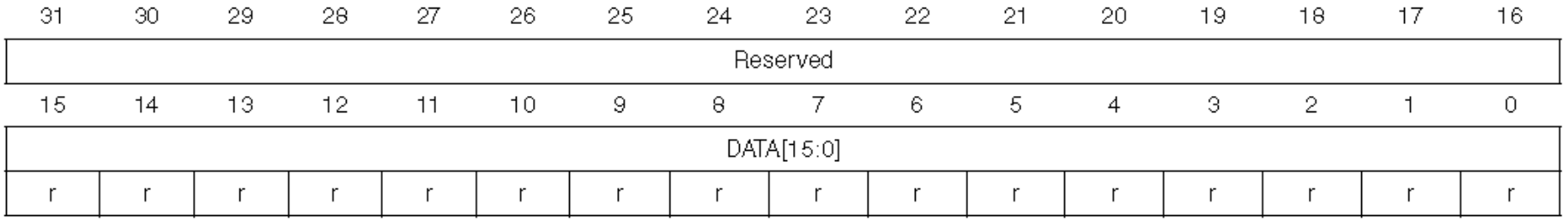
SQR5

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

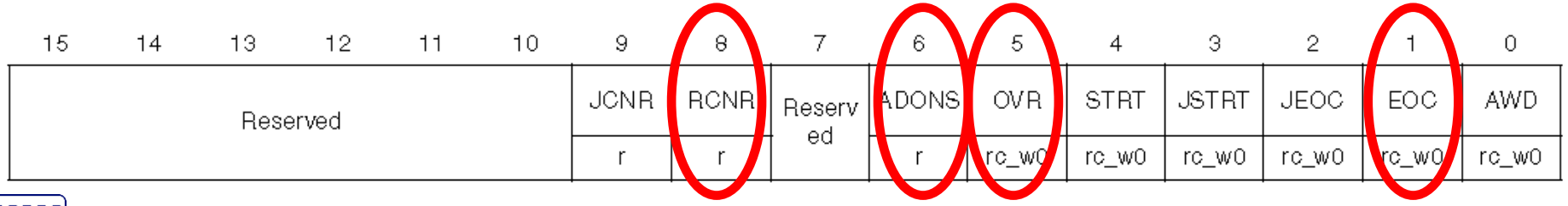
ADC: Registros de Datos

- 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
  - Tenga en cuenta que el dato NO ES ni un voltaje, ni temperatura, ni nada correspondiente a unidades reales, sino el número correspondiente al escalón para el que la tensión de entrada es equivalente, teniendo en cuenta los valores Vref+ y Vref-, el número de bits de conversión
    - Por ejemplo, si Vref-=0V y Vref+=3V, y convertimos a 10 bits tendríamos:
      - Si Vin=3V, el resultado es 1023 en decimal (es decir, 0x3FF)
      - Si Vin= 1,5V, el resultado es 511 en decimal (es decir, 0x1FF)



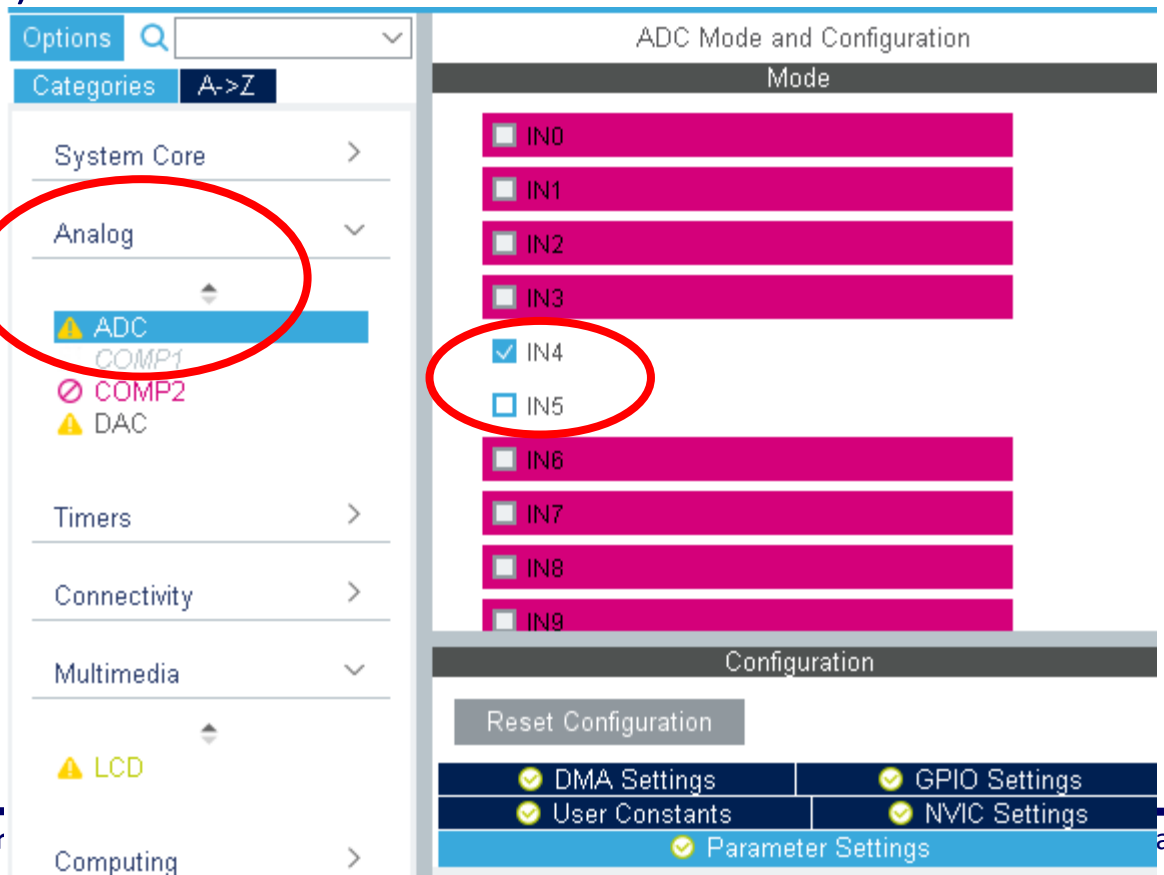
ADC: Registros de Estado

- 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 en este curso.***
    - 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 sobrescrito su valor con el resultado de la conversión actual.
    - EOC – End of Conversion
      - Indica con un 1 que se ha finalizado la conversión en curso y el resultado está en `ADC->DR`



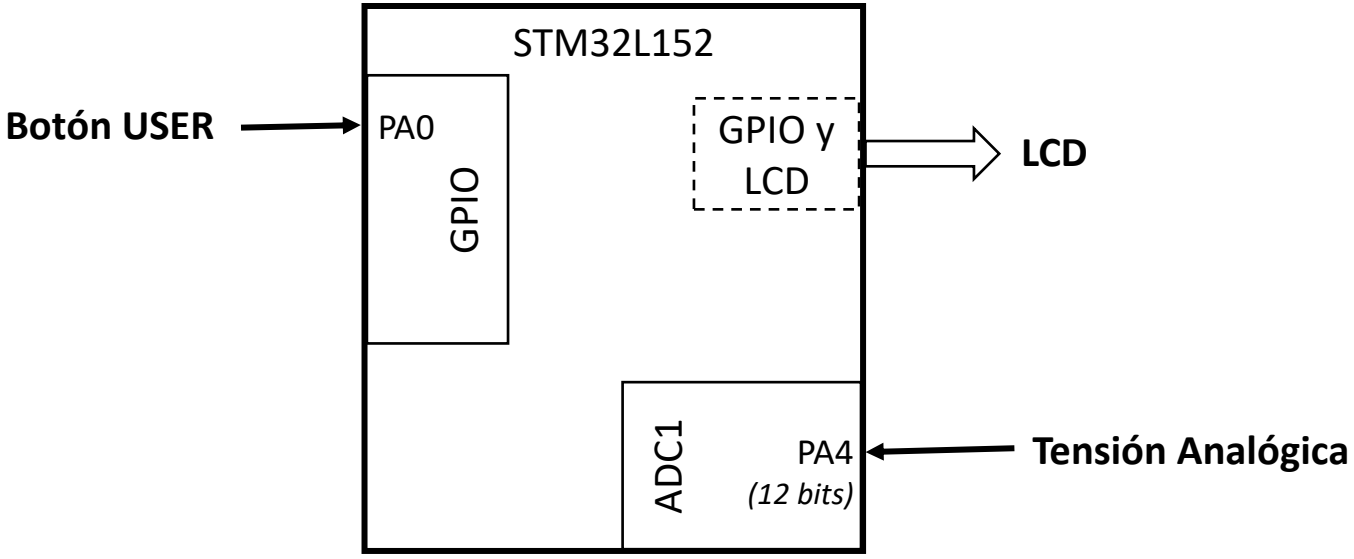
Peculiaridades de la Discovery y la perspectiva CubeMX

- Por defecto el ADC está desconectado para ahorrar energía
- Si se quiere usar el ADC, en CubeMX, en la izquierda, abre Analog, y allí abre ADC
  - En la ventana intermedia, activa el canal que quieres usar (por ejemplo IN4 para PA4)



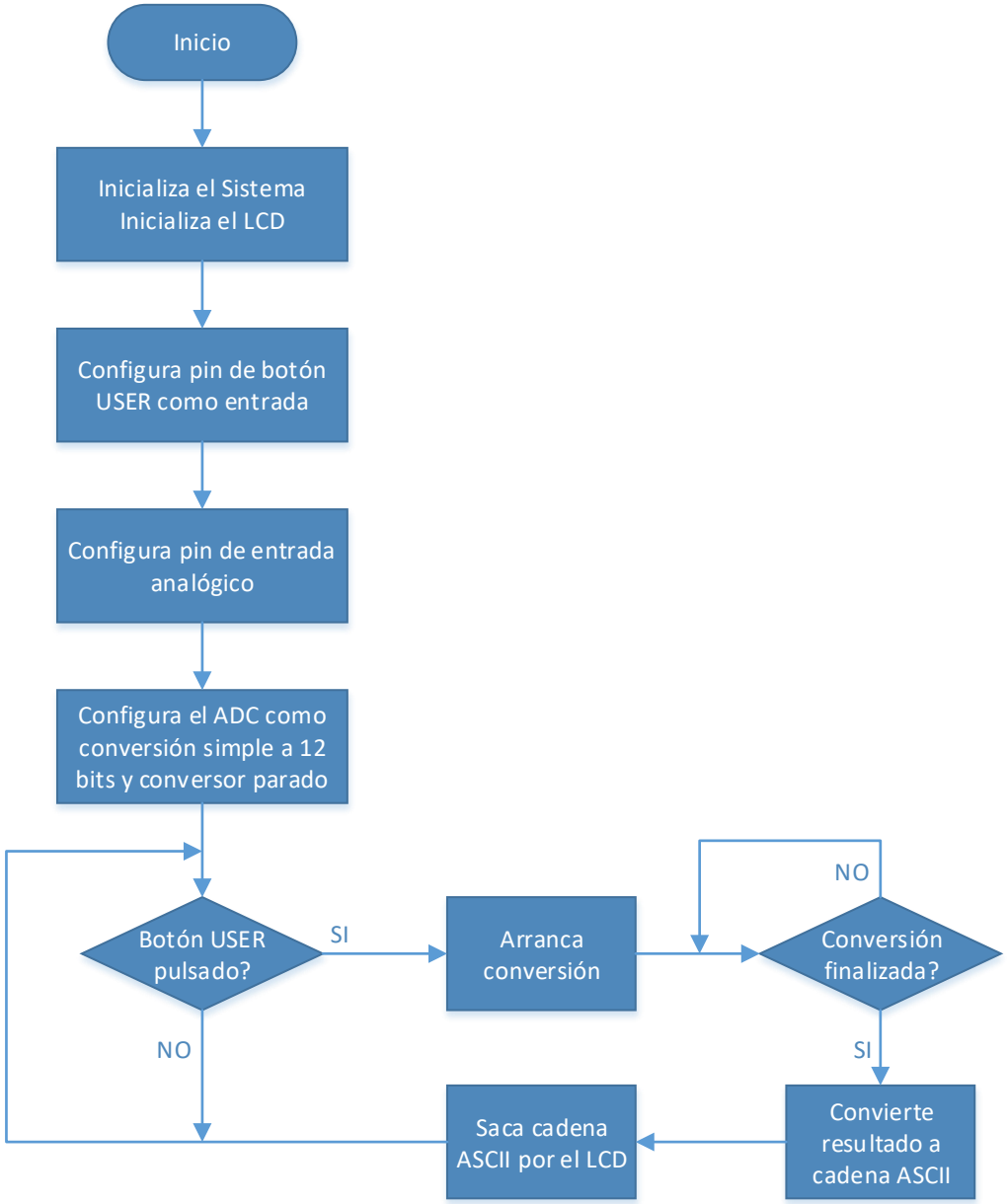
Ejemplo de Conversión Simple

- El siguiente ejemplo convierte cada vez que se pulsa el botón USER y saca el valor de conversión (de 12 bits, es decir, de 0-4095) 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 Conversión Simple

- Y el diagrama de flujo:
  - No se hace referencia directa al microcontrolador o a datos específicos del periférico
  - Las conexiones entre módulos deben estar realizados con flechas unidireccionales (es decir, con puntas de flecha indicando dirección)



Ejemplo de Conversión Simple

- Inicialización:

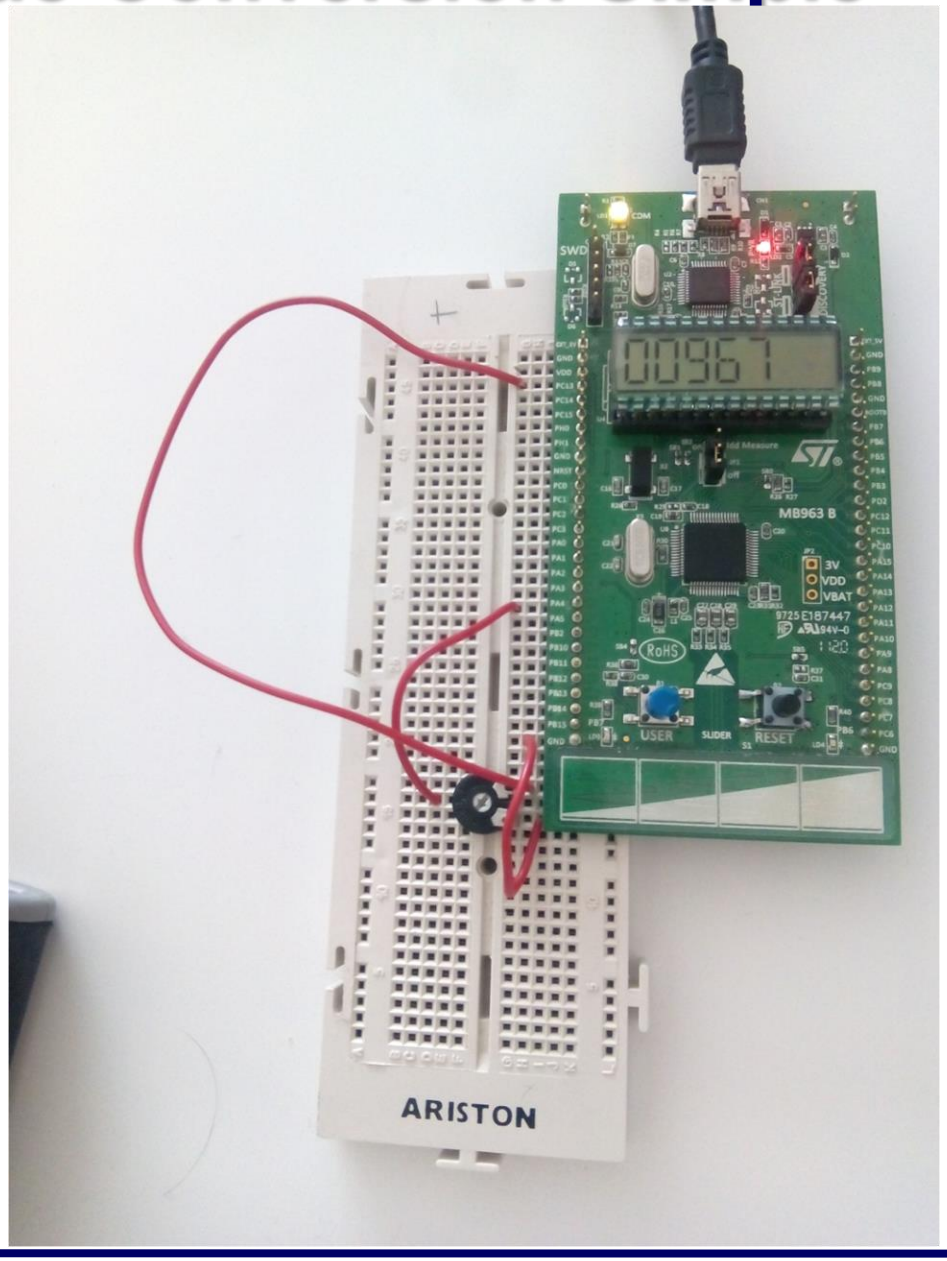
```
/* USER CODE BEGIN 2 */

BSP_LCD_GLASS_Init();
BSP_LCD_GLASS_BarLevelConfig(0);
BSP_LCD_GLASS_Clear();
// USER button configuration (PA0)
// PA0 as digital input(00)
GPIOA->MODER &= ~(1 << (0*2 +1));
GPIOA->MODER &= ~(1 << (0*2));
// ADC configuration
GPIOA->MODER |= 0x00000300;     // PA4 as analog
ADC1->CR2 &= ~(0x00000001);     // ADON = 0 (ADC powered off)
ADC1->CR1 = 0x00000000;         // OVRIE = 0 (Overrun IRQ disabled)
                                // RES = 00 (resolution = 12 bits)
                                // SCAN = 0 (scan mode disabled)
                                // EOCIE = 0 (EOC IRQ disabled)
ADC1->CR2 = 0x00000400;         // EOCS = 1 (EOC to be activated after each conversion)
                                // DELS = 000 (no delay)
                                // CONT = 0 (single conversion)
ADC1->SQR1 = 0x00000000;        // 1 channel in the sequence
ADC1->SQR5 = 0x00000004;        // Channel is AIN4
ADC1->CR2 |= 0x00000001;        // ADON = 1 (ADC powered on)

/* USER CODE END 2 */
```

Ejemplo de Conversión Simple

- Funcionalidad Continua:

```
/* USER CODE BEGIN WHILE */
while (1) {
      if ((GPIOA->IDR&0x00000001)!=0) { // If PA0 = 1 (USER pressed)
                                        // start conversión, otherwise finish
         while ((GPIOA->IDR&0x00000001)!=0) { // If PA0 = 1 (USER pressed),
                                              // wait to avoid rebounds
            espera(70000);
            }
         // Start conversion
         while ((ADC1->SR&0x0040)==0); // While ADONS = 0, i.e., ADC is not ready
                                       // to convert, I wait
         ADC1->CR2 |= 0x40000000; // When ADONS = 1, I start conversion
                                  // (SWSTART = 1)
         // Wait till conversion is finished
         while ((ADC1->SR&0x0002)==0);  // If EOC = 0, i.e., the conversion is not
                                        // finished, I wait
         valor = ADC1->DR;  // When EOC = 1, I take the result and store it in
                            // variable called valor

         // Convert result to string
         Bin2Ascii(valor,&texto[0]);
         // Show result in LCD
         BSP_LCD_GLASS_Clear();
         BSP_LCD_GLASS_DisplayString((uint8_t *)texto);
         }
      /* USER CODE END WHILE */
      }
```

Ejemplo de Conversión Simple



CC BY NC SA

Ejemplo de Conversión Continua

- El siguiente ejemplo convierte de forma continua y saca el valor de conversión (de 12 bits, es decir, de 0-4095) por el LCD
  - Inicialización:

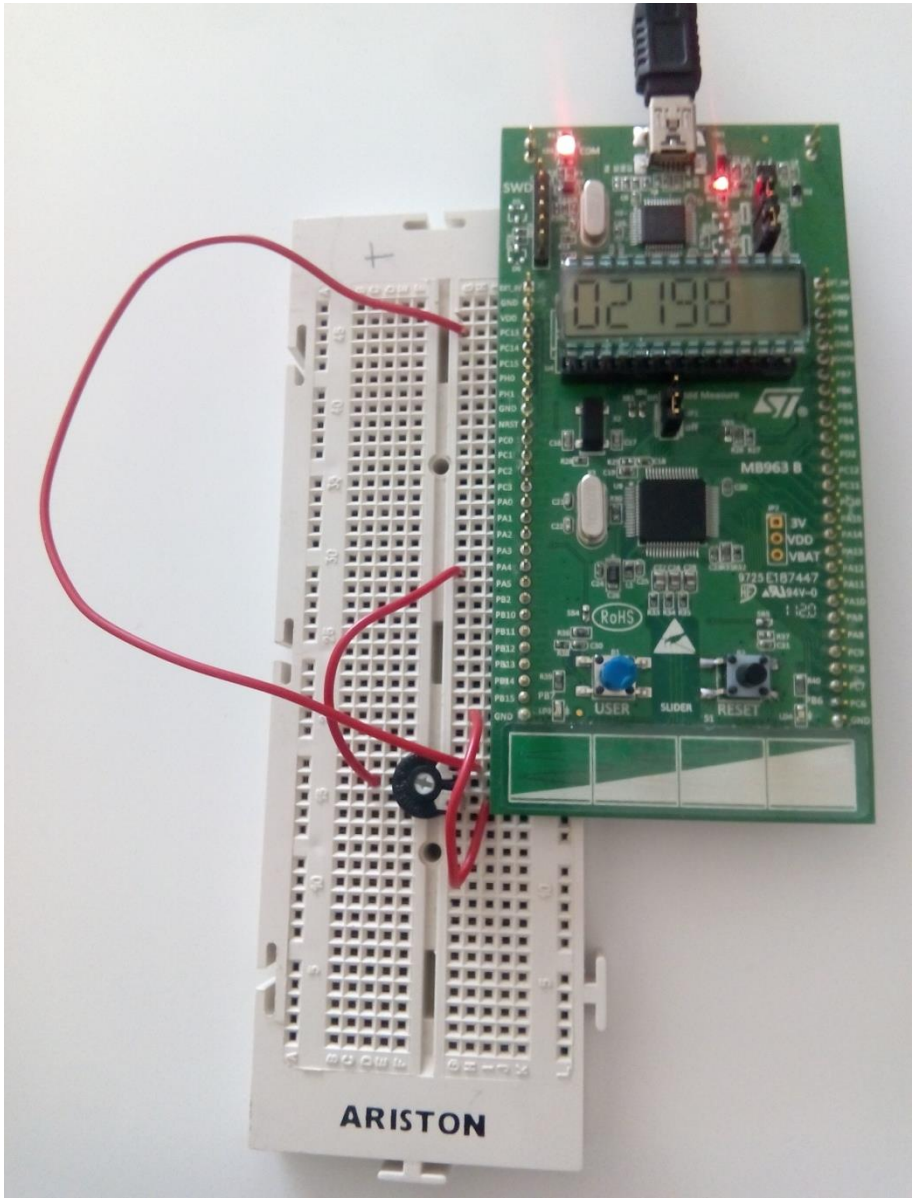
```
/* USER CODE BEGIN 2 */

BSP_LCD_GLASS_Init();
BSP_LCD_GLASS_BarLevelConfig(0);
BSP_LCD_GLASS_Clear();
// ADC Configuration
GPIOA->MODER |= 0x00000300;     // PA4 as analog
ADC1->CR2 &= ~(0x00000001);     // ADON = 0 (ADC powered off)
ADC1->CR1 = 0x00000000;         // OVRIE = 0 (overrun IRQ disabled)
                                // RES = 00 (resolution = 12 bits)
                                // SCAN = 0 (scan mode disabled)
                                // EOCIE = 0 (EOC IRQ disabled)
ADC1->CR2 = 0x00000412;         // EOCS = 1 (EOC is activated after each conversion)
                                // DELS = 001 (delay till data is read)
                                // CONT = 1 (continuous conversion)
ADC1->SQR1 = 0x00000000;        // 1 channel in the sequence
ADC1->SQR5 = 0x00000004;        // The selected channel is AIN4
ADC1->CR2 |= 0x00000001;        // ADON = 1 (ADC powered on)
while ((ADC1->SR&0x0040)==0);   // If ADCONS = 0, I wait till converter is ready
ADC1->CR2 |= 0x40000000;        // When ADCONS = 1, I start conversion (SWSTART = 1)

/* USER CODE END 2 */
```

Ejemplo de Conversión Continua

- Funcionalidad Continua:

```
/* USER CODE BEGIN WHILE */
while (1) {
     valor = ADC1->DR; // I take the result and copy it
                       // in a variable called valor
     // Convert result to a string
     Bin2Ascii(valor,&texto[0]);

     // Show result in the LCD
     BSP_LCD_GLASS_Clear();
     BSP_LCD_GLASS_DisplayString((uint8_t *)texto);

     /* USER CODE END WHILE */
  }
```

Ejemplo de Conversión Continua



CC BY NC SA

CC BY NC SA

Conversión D/A

DAC y Funcionamiento

cc BY NC SA

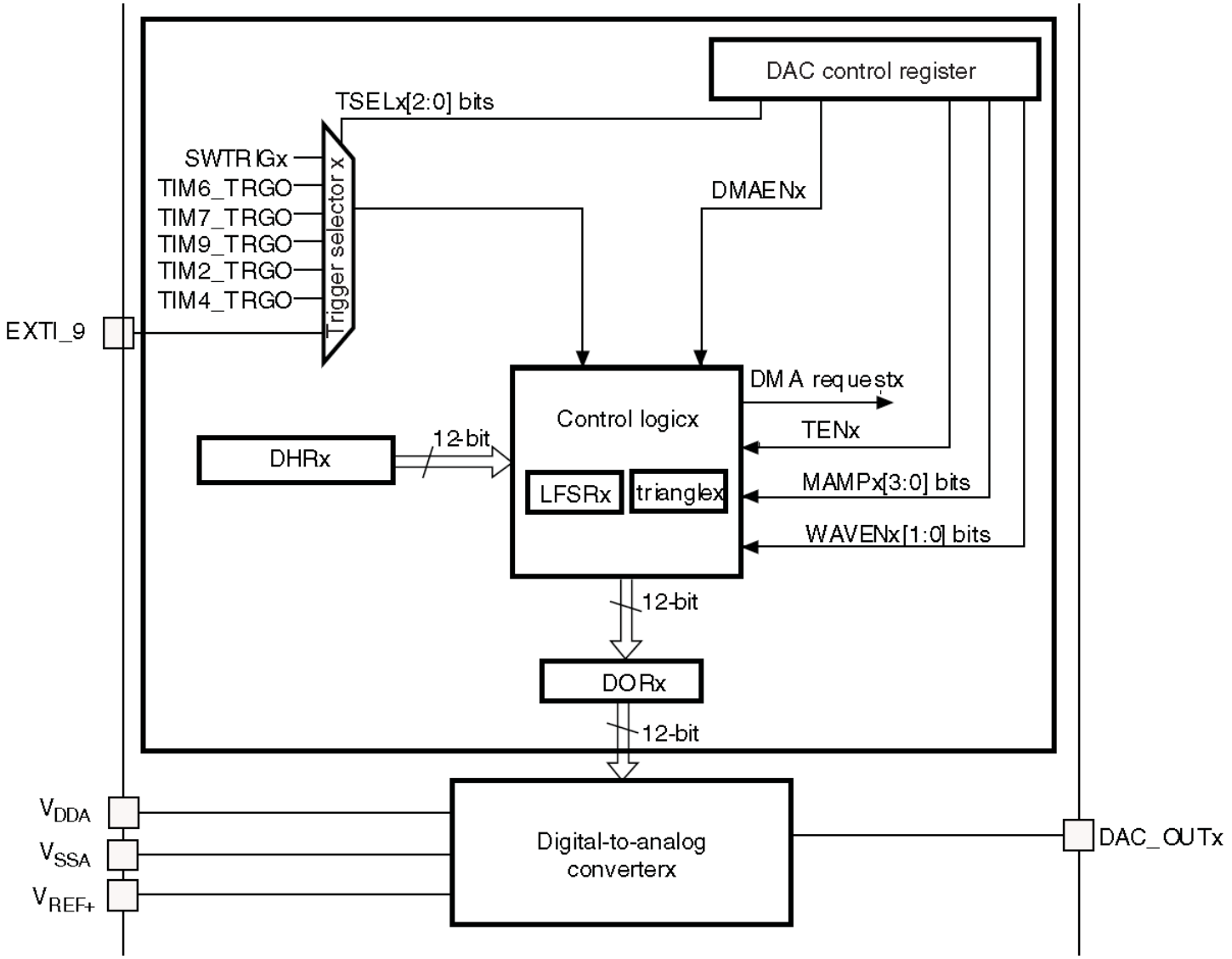
- El conversor DAC realiza la función opuesta al ADC, por lo que pondrá una tensión analógica en la salida, equivalente al valor digital procesado en el programa.
- Se puede utilizar para crear cualquier tipo de señal analógica, siempre que su evolución sea mucho más lenta que el tiempo de conversión del DAC

DAC y Funcionamiento

cc BY NC SA

- El STM32L152 tiene dos DACs de 12 bits, cada uno con su propia salida
- Sus características principales son:
  - Resolución configurable a 12 u 8 bits
  - Posibilidad de gestionar datos digitales alineados a la derecha o a la izquierda
  - Las tensiones a obtener dependen de los valores de dos pines de entrada (Vref+ y masa)
    - En nuestro caso será entre 3,3V y 0V
- El DAC está conectado al APB1
- Para convertir, lo único que hay que hacer es activar el conversor, y poner un valor digital en el registro DHR escogido
  - El de 12-bits alineado a la derecha
  - El de 12-bits alineado a la izquierda
  - El de 8-bits alineado a la derecha

DAC y Funcionamiento



DAC: Registros de Control

- DAC→CR – Control Register:
  - Registro de 32 bits con los siguientes bits de configuración:
    - DMAUDRIE2, DMAEN2, MAMP2[3:0], WAVE2[1:0], TSEL2[2:0], TEN2 – todos a '0'
    - BOFF2 – Deshabilitación del buffer de salida de canal 2 del DAC
      - Con un '0' está habilitado, con un '1' se deshabilita
    - EN2 – Habilitación del canal 2
      - Con un '0' se deshabilita, con un '1' se habilita
    - DMAUDRIE1, DMAEN1, MAMP1[3:0], WAVE1[1:0], TSEL1[2:0], TEN1 – todos a '0'
    - BOFF1 – Deshabilitación del buffer de salida de canal 1 del DAC
      - Con un '0' está habilitado, con un '1' se deshabilita
    - EN1 – Habilitación del canal 1
      - Con un '0' se deshabilita, con un '1' se habilita

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE2[1:0]		TSEL2[2:0]			TEN2	BOFF2	EN2
		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		DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE1[1:0]		TSEL1[2:0]			TEN1	BOFF1	EN1
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

DAC: Registros de Datos

cc BY NC SA

- DAC→DHR12R1 – DAC Data Register con 12 bits y alineado a la derecha (canal 1):
  - Se escribe en los bits [11:0] el dato a convertir
- DAC→DHR12L1 – DAC Data Register con 12 bits alineado a la izquierda (canal 1):
  - Se escribe en los bits [15:4] el dato a convertir
- DAC→DHR8R1 – DAC Data Register con 8 bits alineado a la derecha (canal 1):
  - Se escribe en los bits [7:0] el dato a convertir
- Y de la misma forma existen, para el canal 2:
  - DAC→DHR12R2
  - DAC→DHR12L2
  - DAC→DHR8R2

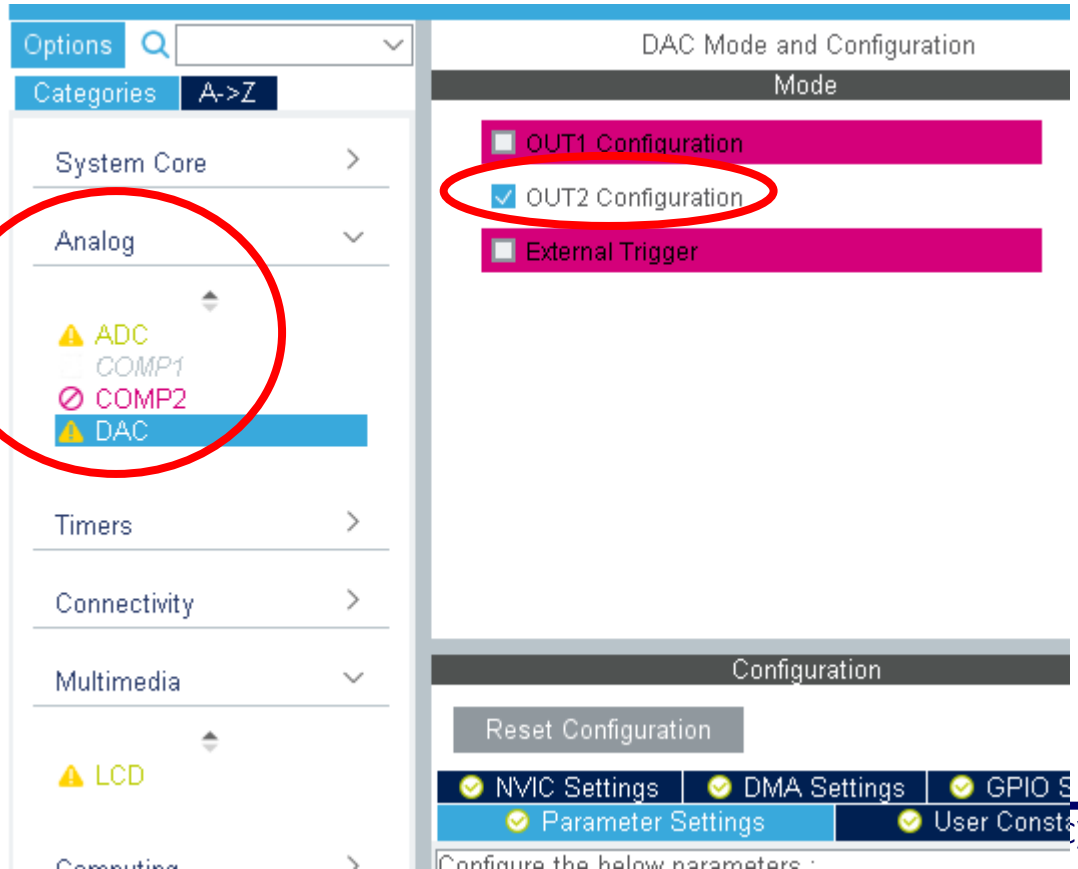
DAC: Registros de Estado

- DAC→SR – Status Registre
  - Registro de 32 bits con sólo 2 bits disponibles, pero ninguno útil para el curso.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DMAUDR2 rc_w1	Reserved												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DMAUDR1 rc_w1	Reserved												

Peculiaridades de la Discovery y la perspectiva CubeMX

- Por defecto el DAC está desconectado para ahorrar energía
- Si se quiere usar el DAC, en CubeMX, en la izquierda, abre Analog, y allí abre DAC
  - En la ventana intermedia, selecciona el canal que se quiere utilizar (por ejemplo, OUT2)



Ejemplo de Uso de Generación de Onda

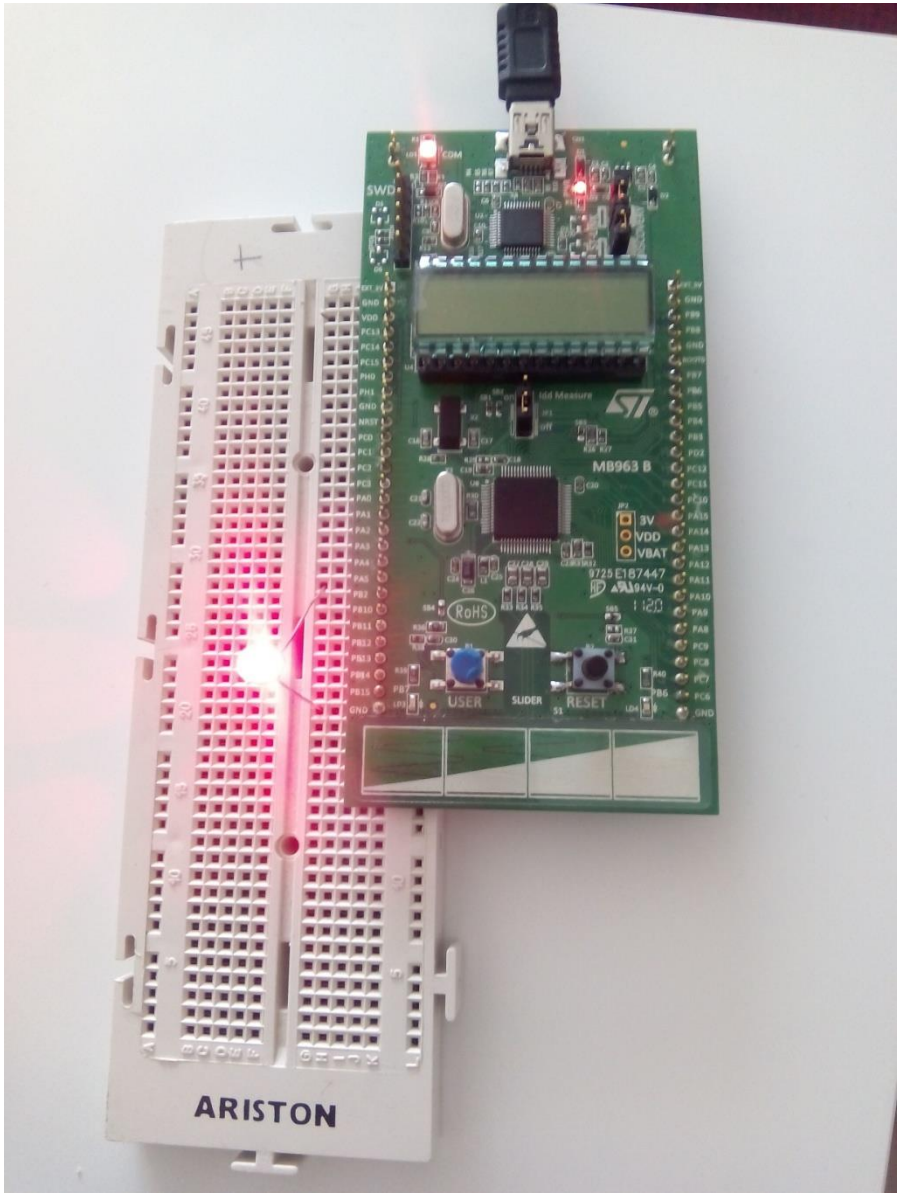
- El siguiente ejemplo genera una onda de variación no sinusoidal, usando valores de 8 bits, por el canal 2 (conectado al PA5)
  - Inicialización:

```
/* USER CODE BEGIN 2 */

unsigned short i = 0;
unsigned short onda[16] = {0, 2, 4, 8, 16, 32, 64, 128, 255, 128, 64, 32, 16, 8, 4, 2};

// DAC Configuration
GPIOA->MODER |= 0x00000C00;          // PA5 as analog signal
DAC->CR = 0x00010000;                // Configuration and enabling of DAC number 2

/* USER CODE END 2 */
```

  - Funcionalidad Continua:

```
/* USER CODE BEGIN WHILE */
while (1) {
   for (i=0; i<16; i++) {            // Get, one by one, the 16 values (one each second)
      DAC->DHR8R2 = onda[i];
      espera(5000000);
      }
/* USER CODE END WHILE */
}
```

Ejemplo de Uso de Generación de Onda



BY NC SA