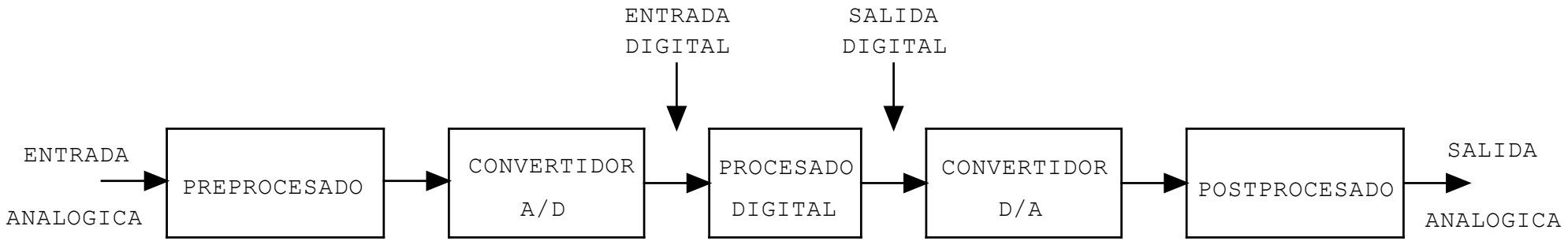
Convertidores D/A y A/D

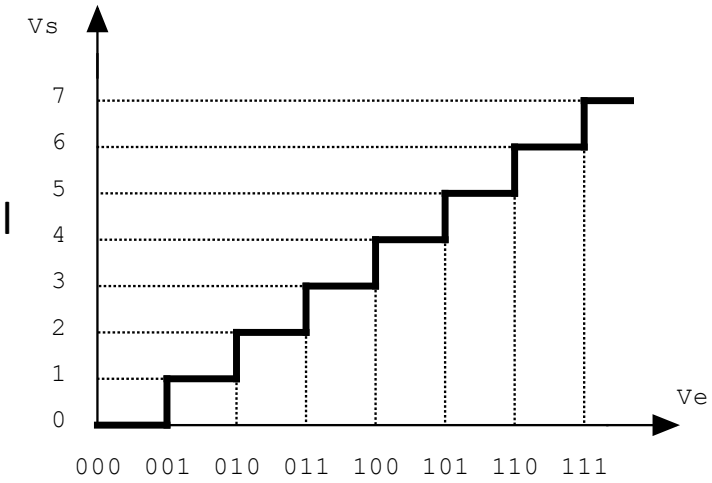
Índice

Introducción.

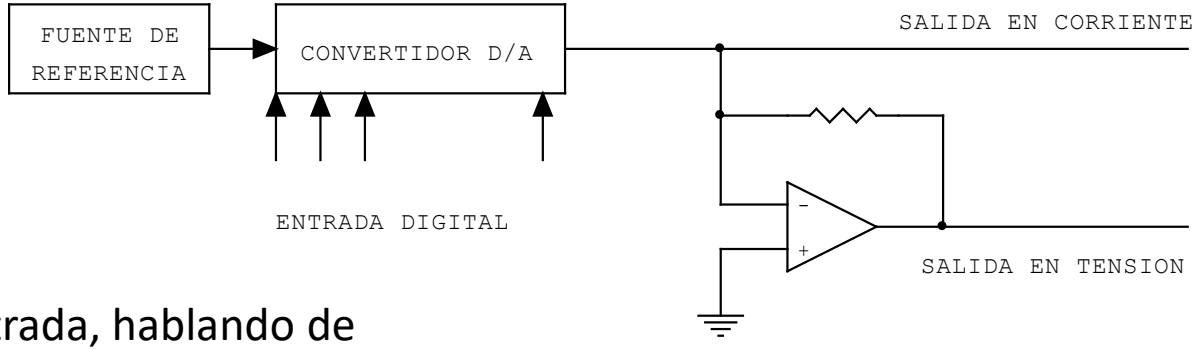
- Con los microprocesadores muchas de las tareas que se realizaban con técnicas analógicas ahora se hacen con técnicas digitales.
- Estas tareas digitales se pueden realizar de forma más sencilla, con mejores prestaciones y de forma más reproducible.
- Pero las magnitudes físicas, lo que medimos o sobre lo que queremos actuar, son reales y analógicas (temperatura, audio, etc).
- Aparece la necesidad de tener circuitos que conviertan la señales analógicas en digitales (Conversor A/D) y otros que vuelvan a convertir las señales digitales en analógicas (Conversor D/A).
- Diagrama de bloques del procesado digital de una señal real:



Convertidores D/A



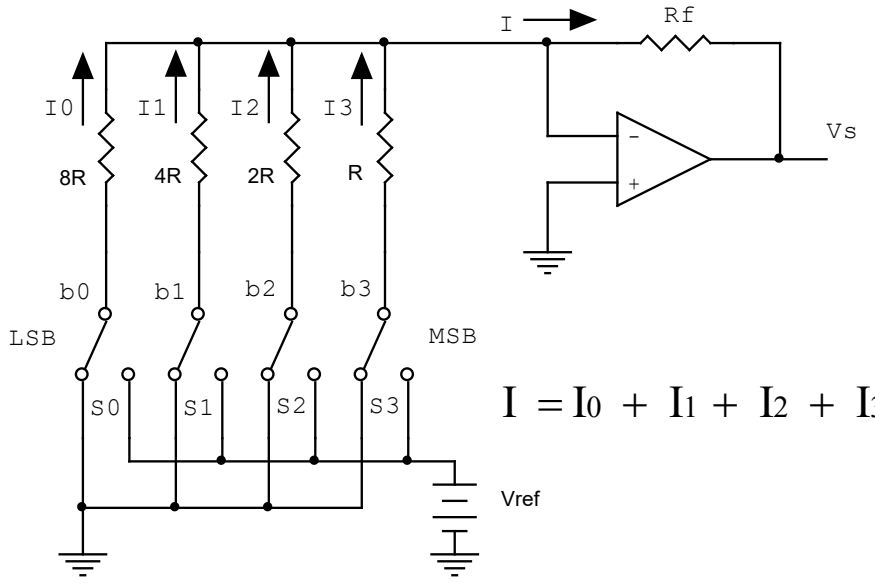
- Circuito que genera una señal analógica de valor proporcional al valor digital de su entrada (Vs=k*Valor digital).
- La salida puede ser en tensión o en corriente, la entrada digital suele ser en binario.
- Tipos:
  a) Conversión directa
     1. Red de resistencias ponderadas
     2. Red de resistencias R-2R
  b) Conversión indirecta
     1. Por generación de impulsos
     2. Por frecuencia variable o estocástico



- Otra clasificación se puede hacer en función de la entrada, hablando de convertidores serie y paralelo.
- También podemos clasificarlos en función de las referencias de tensión: convertidores unipolares (Vref y GND) y bipolares (+Vref y –Vref).

Convertidores D/A
D/A de resistencias ponderadas.

- Conversión directa.
- La señal digital actúa sobre los interruptores (S0 a S3, 4 bits) conectando las resistencias a GND (bit a 0) o a Vref (bit a 1).
- Corriente de salida I para los bits (b3,b2,b1,b0).
- Las referencias de tensión son generalmente negativas con estos convertidores.
- Inconvenientes (sobre todo para D/A de más de 8 bits):
  - La exactitud de las resistencias en todos los valores (por ejemplo un convertidor de 12 bits: la primera resistencia de 1kΩ, la última de 2’048MΩ).
  - La dispersión térmica.
  - La resistencia de los conmutadores no es cero, por lo que afecta al valor de la corriente de salida.



$$I = I_0 + I_1 + I_2 + I_3$$

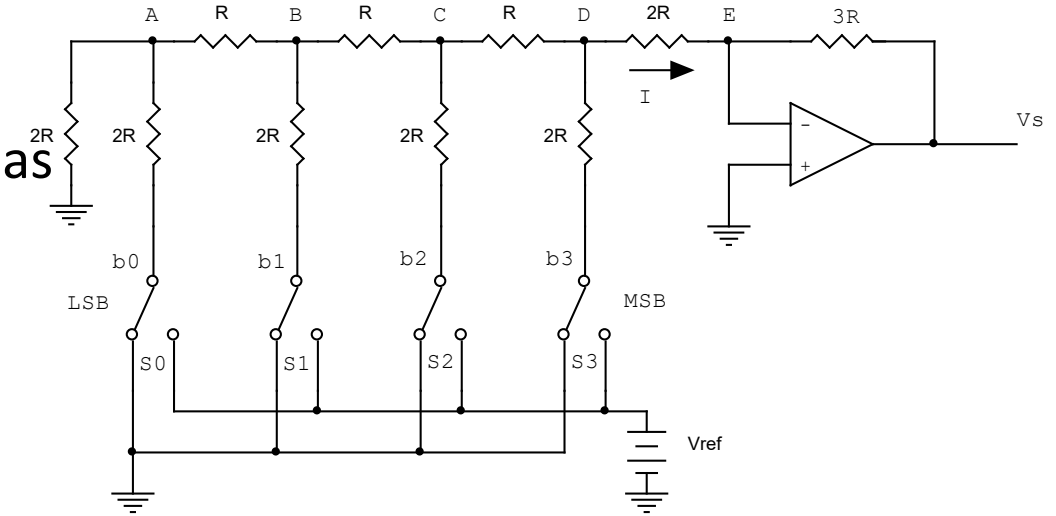
$$I_0 = \frac{V_{REF}}{8 \cdot R} = \frac{V_{REF}}{2^{n-1} \cdot R} \quad I_1 = \frac{V_{REF}}{4 \cdot R}, \; I_2 = \frac{V_{REF}}{2 \cdot R} \; \text{y} \; I_3 = \frac{V_{REF}}{R}$$

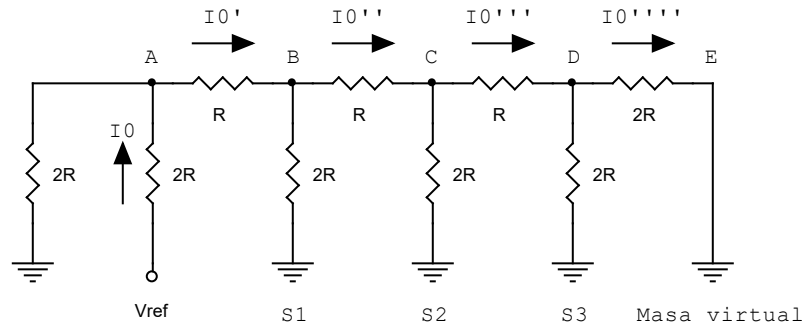
$$I = \frac{V_{REF}}{8R} + \frac{V_{REF}}{4R} + \frac{V_{REF}}{2R} + \frac{V_{REF}}{R} = \frac{V_{REF}}{R}\left(\frac{b0}{8} + \frac{b1}{4} + \frac{b2}{2} + b3\right)$$

$$Vs = -I \cdot Rf = -\frac{Rf}{R} V_{REF}\left(\frac{b0}{8} + \frac{b1}{4} + \frac{b2}{2} + b3\right)$$

Convertidores D/A
D/A con red de resistencias R-2R

- Conversión directa.
- Evita algunos problemas del conversor de resistencias ponderadas.
- Aplicamos superposición para el análisis.
- Suponemos que b0=1 (Vref) y b1=b2=b3=0 (GND).
- Como inconveniente de este sistema indicar que se necesita el doble de resistencias.





$$I0 = \frac{V_{REF}}{2R + (2R//2R)} = \frac{V_{REF}}{3R}$$

$$I0'''' = \frac{I0'''}{2} \qquad I0''' = \frac{I0''}{2} \qquad I0'''' = \frac{I0''}{4}$$

$$-V_S = 3R \cdot I0'''' = 3R \cdot \frac{I0}{16} = 3R \cdot \frac{V_{REF}}{3R} \cdot \frac{1}{16} = \frac{V_{REF}}{16} = V_{REF}\,\frac{1}{2^4}$$

$$-V_S = V_{REF}\left(\frac{b3}{2^1} + \frac{b2}{2^2} + \frac{b1}{2^3} + \frac{b0}{2^4}\right)$$

Convertidores D/A
Parámetros de los convertidores D/A.

- **Resolución**: es el incremento de la tensión analógica de salida para un cambio de un bit de la entrada digital.
  - Por ejemplo, tomemos una salida unipolar comprendida entre 0 y 10 voltios en un convertidor de 10 bits:

$$\text{resolución} = \frac{10 - 0}{2^{10}} = \frac{10}{1024} = 9'76\ \text{mV}$$

- **Precisión**: es la desviación, en el peor caso, de la tensión de salida respecto al valor ideal esperado.
  - Aquí se incluyen errores y otros parámetros, y no debemos confundirnos con la resolución (podríamos tener un convertidor de alta precisión: error menor de 0`01% y baja resolución: cuatro bits). El valor de la precisión se suele expresar como un porcentaje sobre el valor a fondo de escala (0'1% del FS) o como una fracción del LSB (<1/2 LSB)
- **Tiempo de conversión**: es el tiempo que tarda la salida en tomar un nuevo valor estable (dentro de un margen) tras un cambio en la entrada digital.
- **Estabilidad térmica**: es la inmunidad del convertidor D/A a los cambios de temperatura.
  - Se expresa por el número de µV que cambia la salida por cada grado centígrado que varía la temperatura (µV/ºC)

Convertidores D/A
Parámetros de los convertidores D/A.

- **Error de offset**: es la señal de salida que se obtiene en el convertidor D/A cuando la entrada digital es cero.
  - Estará presente en la salida independientemente de la entrada digital aplicada. Este error se puede ajustar con la ayuda de un potenciómetro exterior.
- **Error de ganancia (o escala)**: este error se debe a que la ganancia del convertidor difiere de la real (por arriba o por abajo).
  - Los escalones son mayores o menores de lo que deberían ser. También se puede corregir con un potenciómetro
- **Error de linealidad**: se trata del error que se produce cuando incrementos iguales en la combinación de la entrada digital producen incrementos desiguales en la señal analógica de salida.
  - En realidad deberíamos decir de “no linealidad”.
- **Error de monotonicidad**: podría definirse como un caso extremo del error de linealidad, ya que se dice que un convertidor DAC es monotónico siempre que al incrementarse positivamente la combinación binaria de la entrada se produce un aumento de la señal analógica de salida.
  - Este error aparece pues, cuando al incrementarse la combinación binaria de la entrada se produce una disminución de la señal analógica de salida. Es más acusado en los convertidores que emplean el método de las resistencia ponderadas o el de las resistencias R-2R.

Convertidores D/A
ESP32

- Convertidores D/A en ESP32
  - Tiene 2 convertidores de 8 bits
    - Canal 1 conectado al GPIO25 (y al altavoz) DAC_CHANNEL_1
    - Canal 2 conectado al GPIO26 DAC_CHANNEL_2
  - Permite conversión simultánea en los dos canales e independiente
  - Tiene un driver para permitir diferentes tensiones
  - Permite acceso vía DMA mediante el I2S usando el “built-in DAC mode”
  - Posee un generador de ondas cosenoidales (CW)

Convertidores D/A ESP32

Registro que elige la señal a convertir

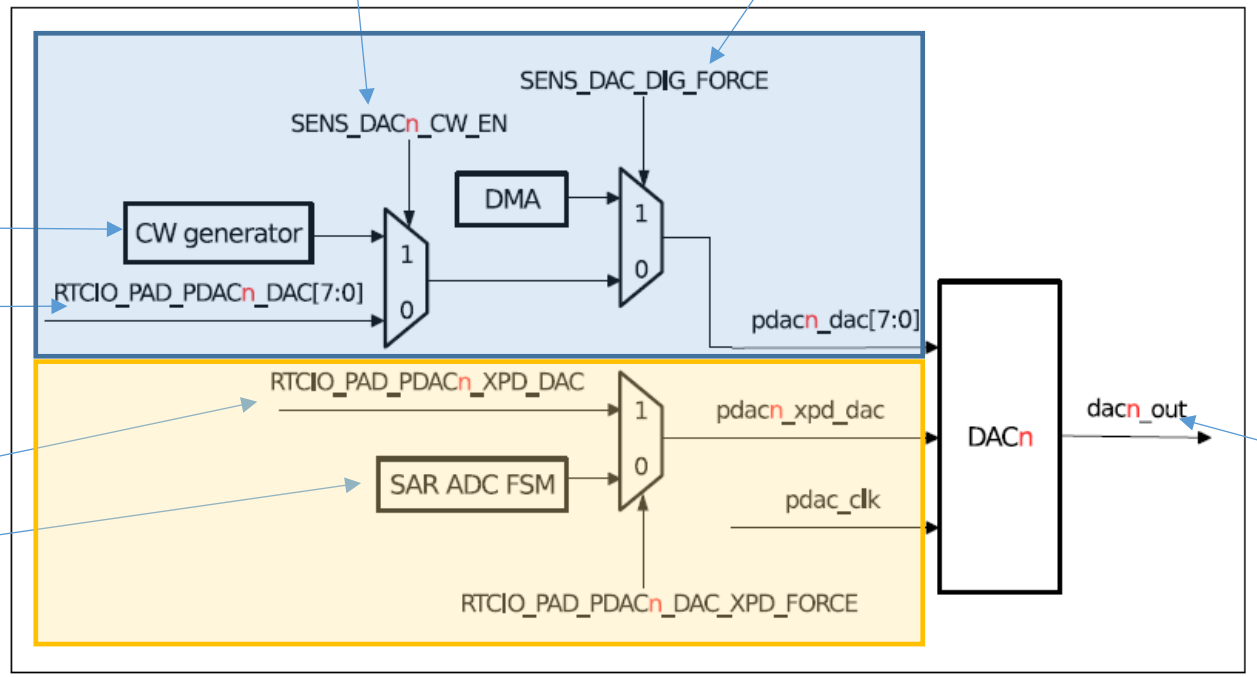
Registro que elige generador o entrada digital

Circuito selección de la señal a convertir

Generador ondas cosenoidales

Entrada digital

Control de la conversión:
- Por software
- Por el ADC



Salida analógica

Figure 148: Diagram of DAC Function

Fuente: esp32 technical refence manual (capítulo 29.5)

pdac_clk = 57kHz

Convertidores D/A
ESP32. Ejemplo con librería Arduino

- Función escritura en el DAC

```
void dacWrite(uint8_t pin, uint8_t value);
```

- Ejemplo generación de ondas sinusoidales

```
void setup() {
}

void loop() {
  for (int deg = 0; deg < 360; deg = deg + 1){
    dacWrite(25, int(128 + 127 * (sin(deg*PI/180))));
    dacWrite(26, int(128 + 127 * (sin(deg*PI*4/180))));
  }
  //dacWrite(26, 80);
}
```

Si la frecuencia de reloj por defecto es de 57kHz

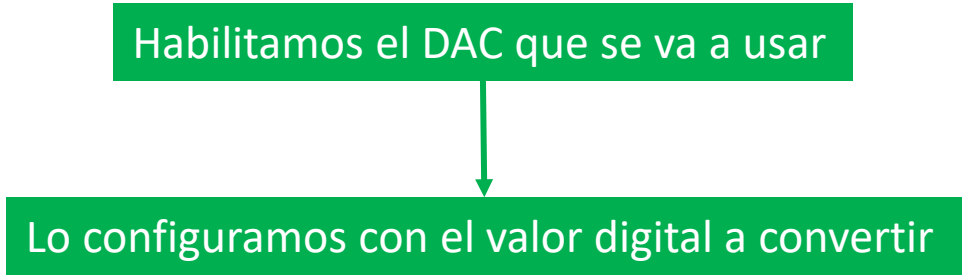
¿Cuál será la frecuencia de la onda generada en los DACs del ejemplo?

Arduino-ESP32 > Libraries > DAC >https://espressif-docs.readthedocs-hosted.com/projects/arduino-esp32/en/latest/api/dac.html

Convertidores D/A
ESP32. Ejemplo con esp-idf

- Configurar el canal 1 (GPIO25) para que ofrezca una tensión analógica aprox. 78% de VDD_A (siendo VDD_A=3.3V)

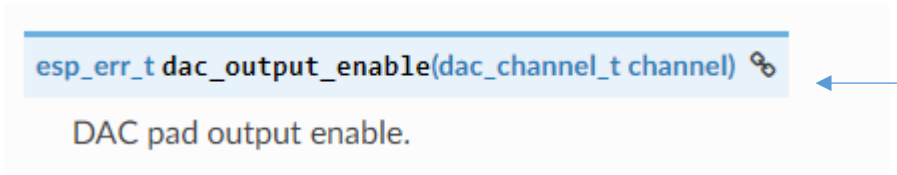
(VDD_A * 200 / 255) para VDD_A=3.3 salida a 2.59V

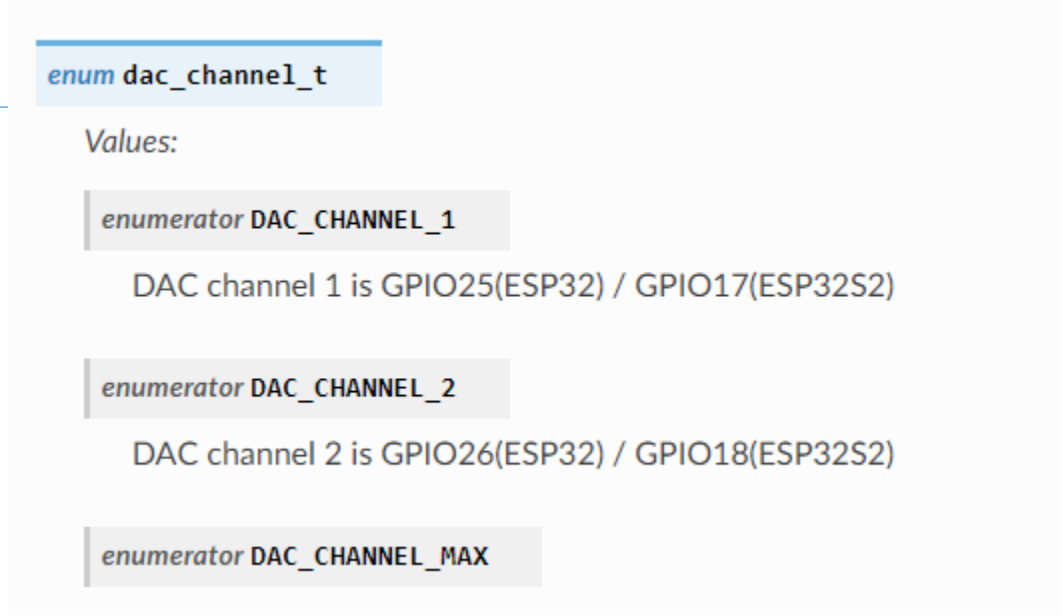


API- Reference> DAC> https://docs.espressif.com/projects/esp-idf/en/release-v4.1/api-reference/peripherals/dac.html

Convertidores D/A
ESP32. Ejemplo con esp-idf

Habilitamos el DAC que se va a usar





API- Reference> DAC> https://docs.espressif.com/projects/esp-idf/en/release-v4.1/api-reference/peripherals/dac.html

Convertidores D/A
ESP32. Ejemplo con esp-idf

Lo configuramos con el valor digital a convertir

esp_err_t **dac_output_voltage**(dac_channel_t channel, uint8_t dac_value)

Set DAC output voltage. DAC output is 8-bit. Maximum (255) corresponds to VDD3P3_RTC.

Note

Need to configure DAC pad before calling this function. DAC channel 1 is attached to GPIO25, DAC channel 2 is attached to GPIO26

Parameters:
- **channel** – DAC channel
- **dac_value** – DAC output value

Returns:
- ESP_OK success

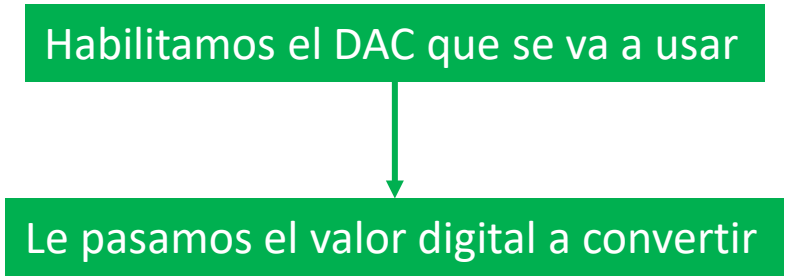
API- Reference> DAC>https://docs.espressif.com/projects/esp-idf/en/release-v4.1/api-reference/peripherals/dac.html

Convertidores D/A
ESP32. Ejemplo con esp-idf

- Configurar el canal 1 (GPIO25) para que ofrezca una tensión analógica aprox. 78% de VDD_A (siendo VDD_A=3.3V)

  (VDD_A * 200 / 255) para VDD_A=3.3 salida a 2.59V

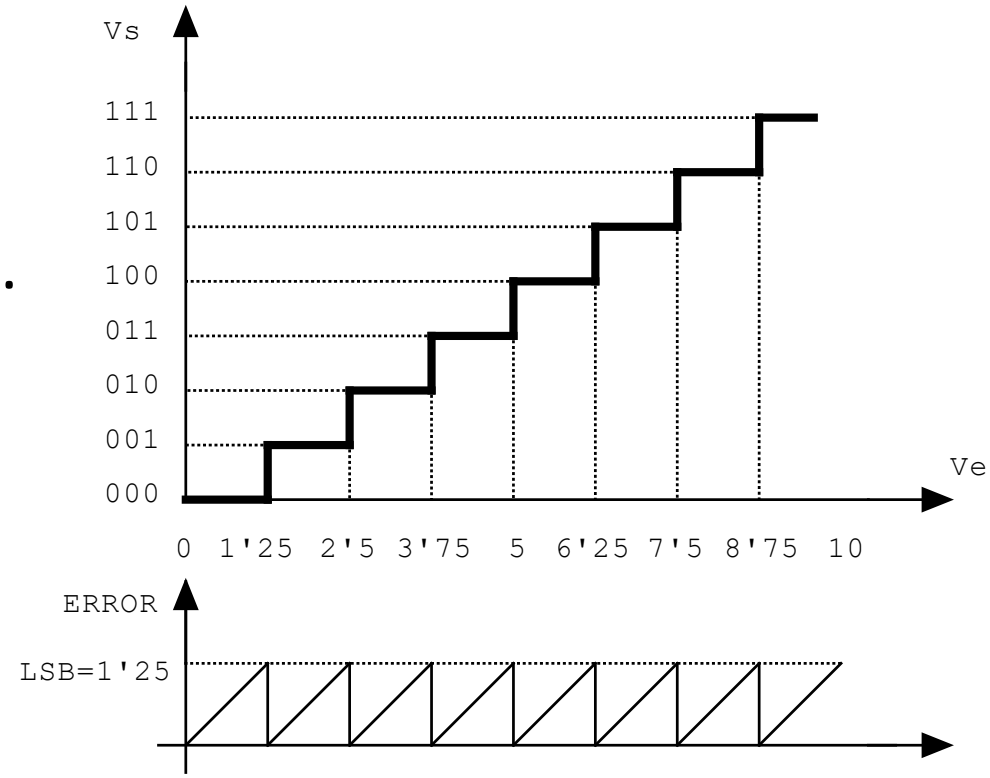
```
#include <driver/dac.h>

...

    dac_output_enable(DAC_CHANNEL_1);
    dac_output_voltage(DAC_CHANNEL_1, 200);
```



API- Reference> DAC> https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/dac.html

Convertidores A/D

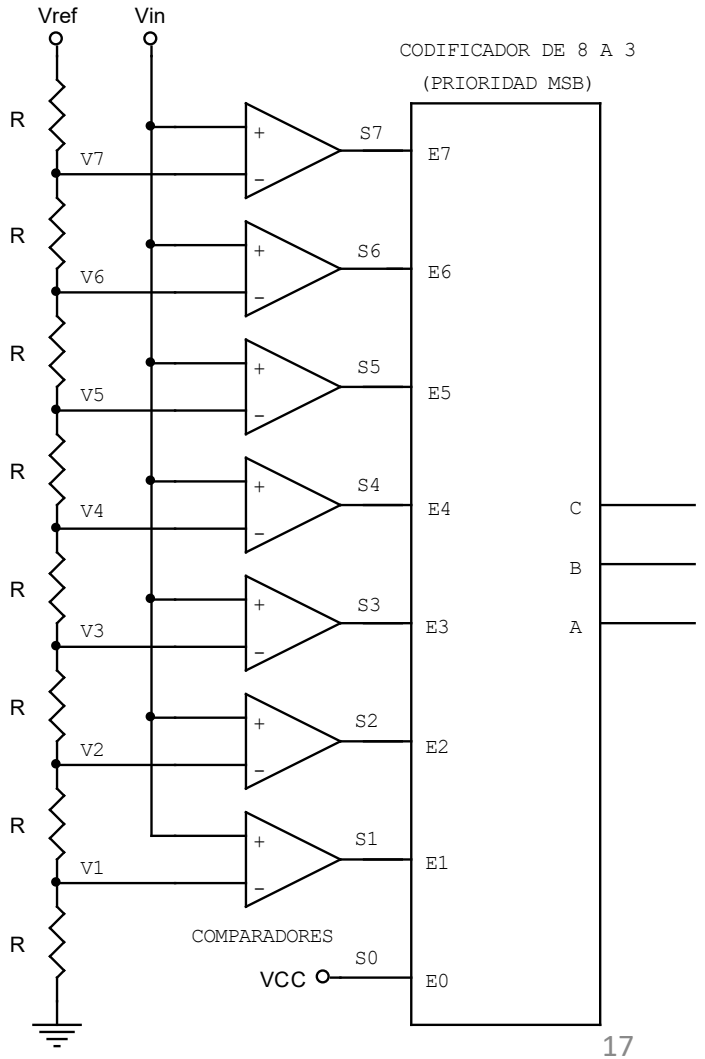
- Circuito que genera un valor digital de la magnitud analógica que tenga a la entrada.
- Tipos:
  - Convertidor A/D de bucle abierto
    - Convertidor con comparadores
    - Conversión por contaje
      - De rampa analógica sencilla
      - De doble rampa analógica
  - Convertidor A/D de bucle cerrado
    - Conversión con contadores
      - Con rampa en escalera
      - De cuenta continua
    - Conversión por aproximaciones sucesivas (SAR)



$$\frac{10}{2^3} = \frac{10}{8} = 1'25\text{V}$$

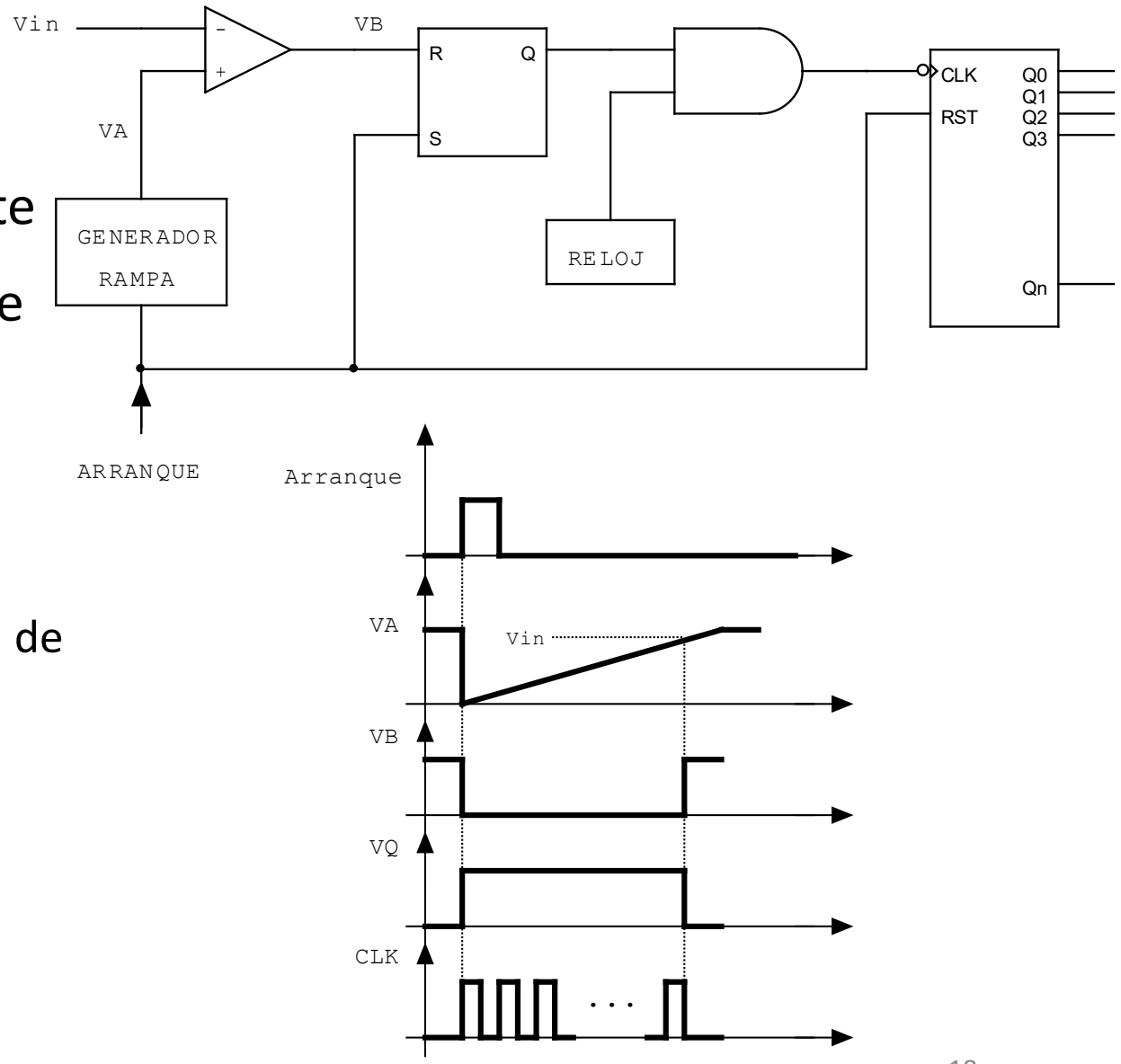
Convertidores A/D
A/D con comparadores

- Circuito codificador de 3 bits.
- Codificador con prioridad
- Funcionamiento:
  - Si Vin < V1 todas las salidas serán 0 (000).
  - Si V1 < Vin < V2 se activa el bit (001).
  - Así sucesivamente.
  - Si Vin > V7 todas las salidas a 1 (111).
- Conversión directa y muy rápida.
- Inconvenientes:
  - Valor de las resistencias afecta a la precisión.
  - Distorsión por el cambio de parámetros (temperatura)
  - Elevado número de comparadores $2^n - 1$ (caro)



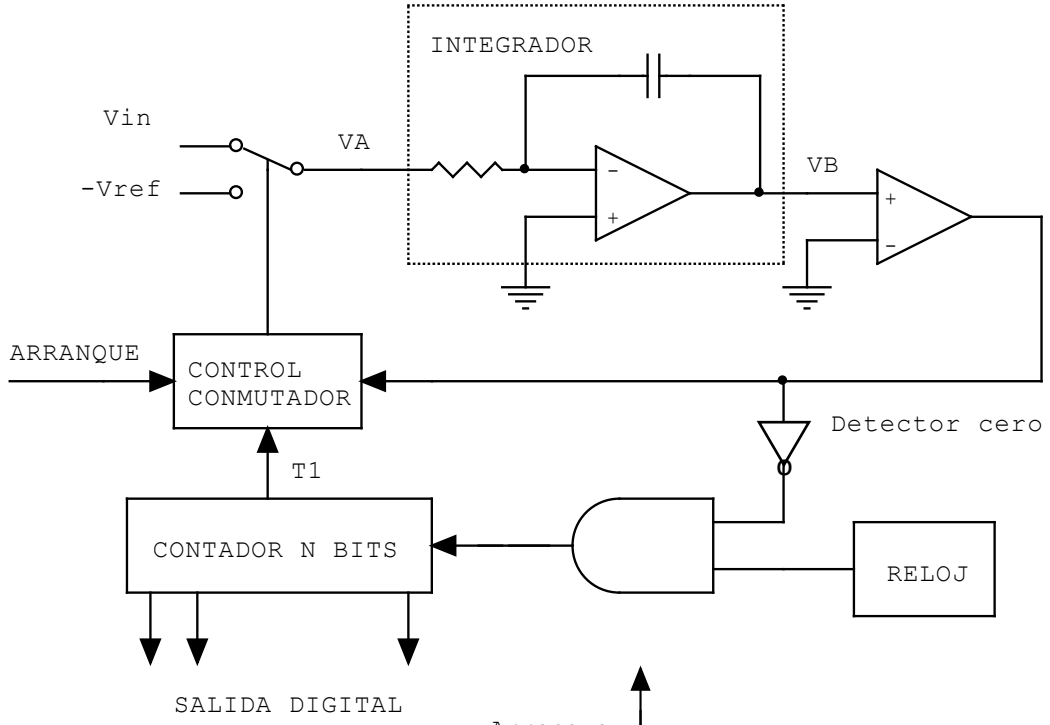
Convertidores A/D
A/D de rampa simple

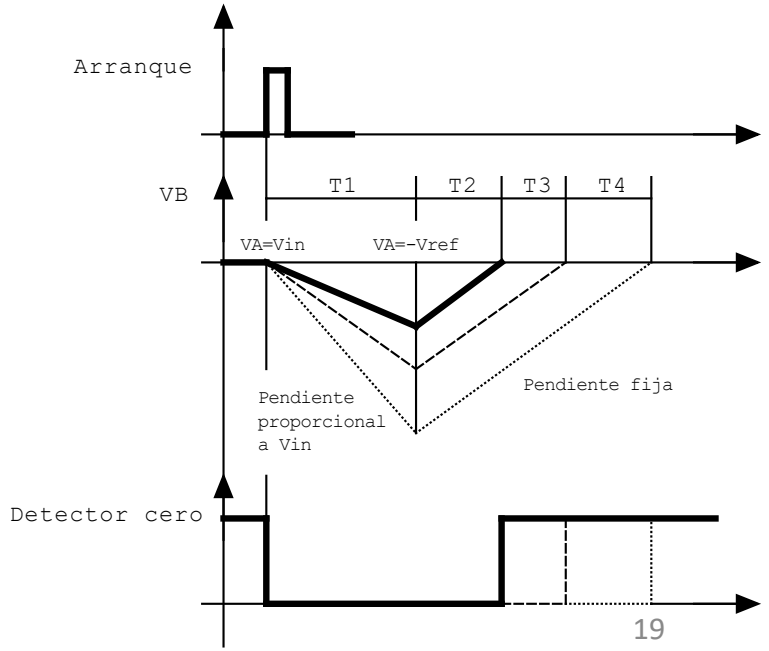
- Utiliza un integrador de simple pendiente de forma que la tensión de entrada se transforma en un periodo de tiempo que se mide con un contador.
- Funcionamiento:
  - Se genera la rampa VA
  - Se activa el biestable RS (salida 1)
  - Contador empieza a contar desde 0.
  - Si tensión VA = Vin, salida comparador pasa de 0 a 1.
  - Reset al biestable RS (salida 0)
  - Deja de contar el contador
- Inconvenientes:
  - Tiempo de conversión variable
  - Falta de linealidad en la rampa afecta conversión.



Convertidores A/D
A/D de doble rampa

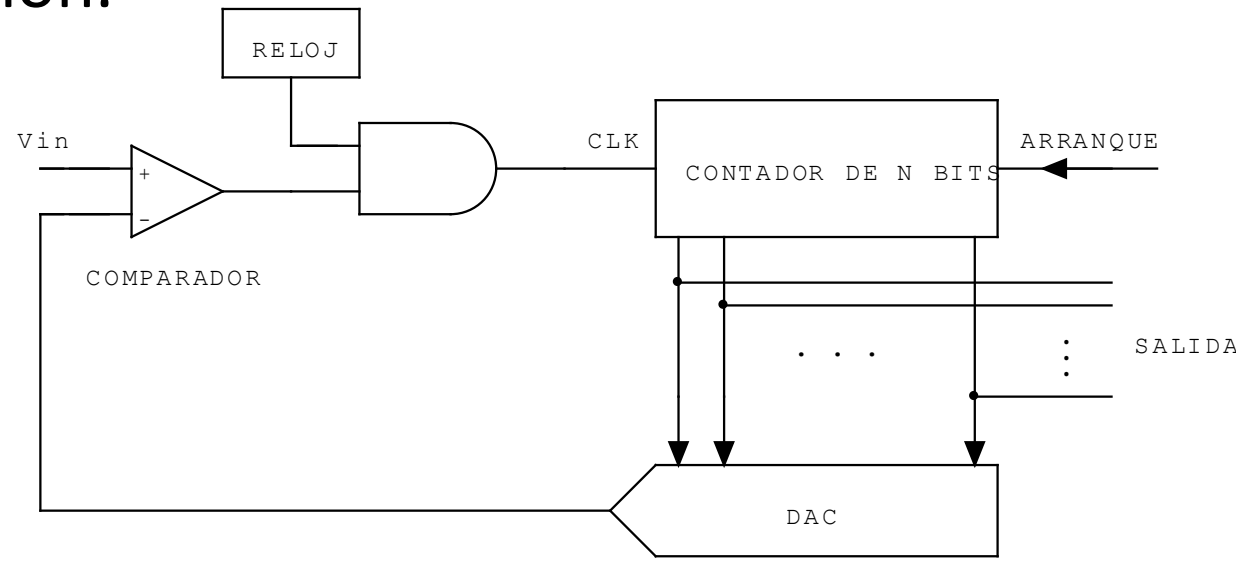
- Circuito codificador de 3 bits.
- Muy usado en multímetros digitales
- Características:
  - Gran precisión
  - Elevado tiempo de conversión
- Funcionamiento:
  - Aplicar Vin al integrador durante un tiempo fijo T1. Se hace coincidir con el desbordamiento del contador.
  - Se carga el condensador con un valor de tensión negativo proporcional a Vin.
  - Se conecta a –Vref provocando que vuelva a cero el valor de tensión en el condensador. Contador cuenta desde cero.
  - Leer el valor del contador. El tiempo que ha tardado en volver a ser cero la tensión del condensador dependerá del valor de Vin.





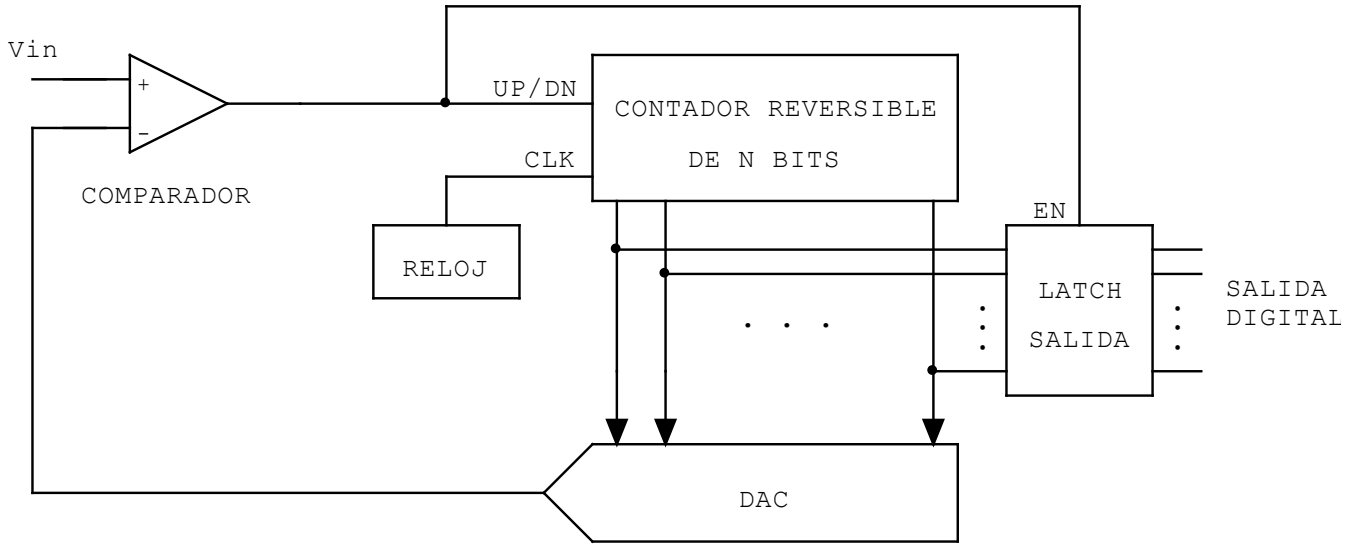
Convertidores A/D
A/D con contador y rampa en escalera

- Circuito codificador de N bits.
- Método sencillo con realimentación.
- Funcionamiento:
  - Poner a cero el contador.
  - Salida del DAC cero.
  - Comparador (DAC < Vin) salida 1
  - Aumenta la cuenta
  - Hasta que (DAC > Vin)
- Inconveniente:
  - Tiempo de conversión depende del valor Vin.
  - Mayor cuanto más nº bits del convertidor.



Convertidores A/D
A/D con contador continuo

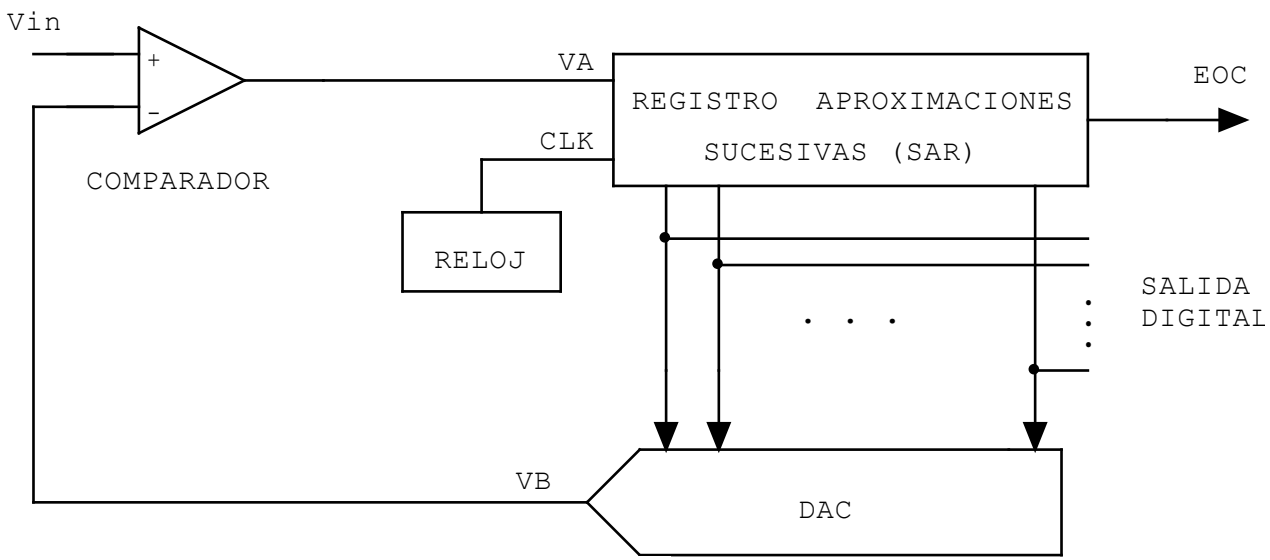
- Circuito codificador de N bits.
- Reduce los inconvenientes del convertidor anterior.
- Funciona de forma continua
  - Mientras la salida del DAC < Vin, contador UP → Salida del DAC aumenta.
  - Cuando la salida del DAC > Vin, contador DOWN → Salida del DAC disminuye.
  - En ese momento se captura el valor digital con el LATCH de salida.



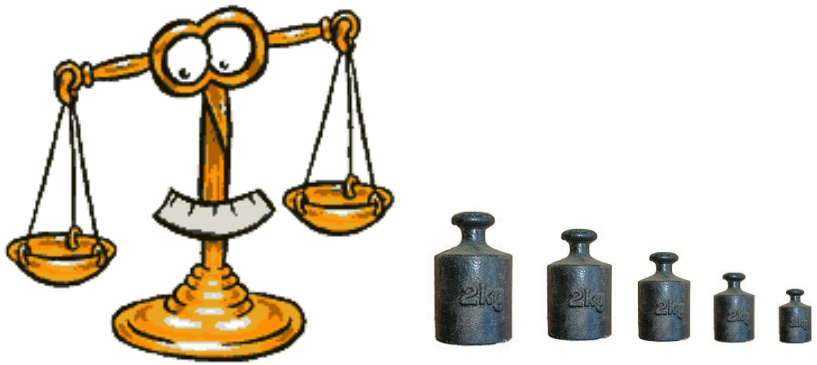
- Como la salida del DAC vuelve a ser menor que Vin volverá a contar UP. Y así sucesivamente.
- Hace un seguimiento de la señal Vin.
- Tiempo de conversión pequeño para señales que no varían rápidamente.
- Sólo funciona bien si tiene un único canal.

Convertidores A/D
A/D de aproximaciones sucesivas (SAR)

- SAR: Registro de aproximaciones sucesivas

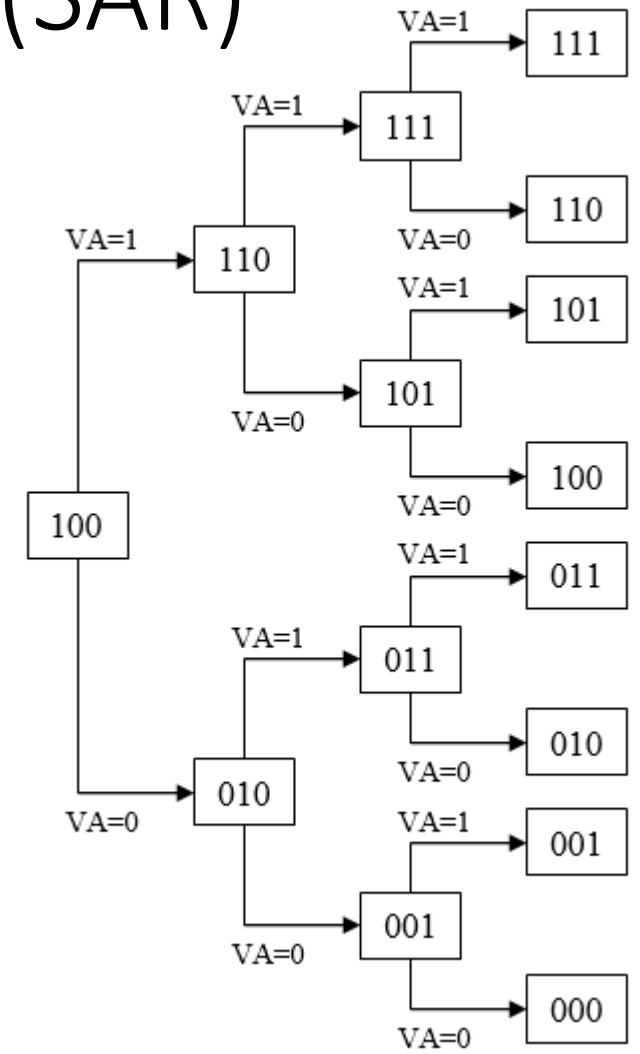


- El proceso es similar al que se emplea para pesar un objeto mediante una balanza y pesos.
- Empezamos colocando la pesa más grande y vamos poniendo o quitando pesas según indique la balanza.



Convertidores A/D
A/D de aproximaciones sucesivas (SAR)

- Circuito codificador de 3 bits.
- Diagrama que representa las iteraciones del SAR.
  - Inicialmente se carga el registro con el valor MSB=1 (100).
  - El DAC genera una tensión VB.
  - Se compara VB con la entrada Vin obteniéndose la señal VA.
    - Si VA=1, VB > Vin y se añade un 1 al SAR poniendo (110)
    - Si VA=0, VB < Vin y se quita el (100) para poner (010)
  - Se repite el proceso con el resto de bits.
  - El tiempo de conversión es fijo y depende del nº de bits.



Convertidores A/D
Parámetros de los convertidores A/D

- **Resolución**: indica el incremento o decremento de la tensión necesaria para modificar el bit de menor peso, es decir, para incrementar o decrementar en un bit el valor digital a la salida

$$\text{Resolución} = \frac{V_{FS}}{2^n - 1}$$

- **Tiempo de conversión**: es el tiempo que transcurre desde que se da la orden de inicio de la conversión hasta que se obtiene a la salida el dato digital.
- **Margen de tensiones analógicas de entrada**: nos indica el valor máximo y mínimo admisibles a la entrada analógica.
- **Códigos de salida**: indica el código utilizado para expresar el valor digital a la salida.
  - Además del código binario suelen utilizarse otros códigos como el BCD Natural, el binario desplazado, complemento a 1 (o a 2), etc. En muchas ocasiones podremos programar el rango de la tensión de entrada y el código de salida entre varias opciones.

Convertidores A/D
Parámetros de los convertidores A/D

- **Error de cuantificación**: este error se debe a que la señal analógica se divide en 2^n partes, de manera que todos los valores analógicos dentro de una parte se representan por un único código digital. Este error es siempre de: ½ LSB.
- **Error de offset**: este error se presenta cuando la curva de transferencia del convertidor A/D está desplazada respecto a la ideal.
- **Error de ganancia**: este error se produce cuando las funciones de transferencia real e ideal tienen pendientes distintas.
- **Precisión**: este parámetro engloba parámetros como el error de cuantificación, el offset, error de ganancia, etc., y nos da una idea de la calidad del convertidor.

Convertidor A/D ESP32.

- 2 convertidores SAR de 12 bits (ADC1 y ADC2)
- 18 pines analógicos
- Permite configurar atenuación
- Permite configurar precisión (12, 11, 10 y 9 bits)
- Permite operar durante el Deep sleep usando el coprocesador ULP (ultra low power)
- El ULP también se usa cuando se está muestreando en modo continuo
- ADC1
  - 8 canales (GPIO32 - GPIO39)
  - Lectura interna del sensor de efecto Hall (GPIO 36 y 39)
- ADC2 (mayor prioridad)
  - 10 canales (GPIO 0, 2, 4, 12 a 15 y 25 a 27)
  - Se utiliza para la Wifi (Desactivarlo si vamos a usar la Wifi)
  - Permite configurar la precisión con cada lectura

Table 6: Peripheral Pin Configurations

Interface	Signal	Pin	Function
ADC	ADC1_CH0	SENSOR_VP	Two 12-bit SAR ADCs
	ADC1_CH1	SENSOR_VN	
	ADC1_CH2	SENSOR_CAPP	
	ADC1_CH3	SENSOR_CAPN	
	ADC1_CH4	32K_XP	
	ADC1_CH5	32K_XN	
	ADC1_CH6	VDET_1	
	ADC1_CH7	VDET_2	
	ADC2_CH0	GPIO4	
	ADC2_CH1	GPIO0	
	ADC2_CH2	GPIO2	
	ADC2_CH3	MTDO	
	ADC2_CH4	MTCK	
	ADC2_CH5	MTDI	
	ADC2_CH6	MTMS	
	ADC2_CH7	GPIO27	
	ADC2_CH8	GPIO25	
	ADC2_CH9	GPIO26	
DAC	DAC_1	GPIO25	Two 8-bit DACs
	DAC_2	GPIO26	
Touch Sensor	TOUCH0	GPIO4	Capacitive touch sensors
	TOUCH1	GPIO0	
	TOUCH2	GPIO2	
	TOUCH3	MTDO	
	TOUCH4	MTCK	
	TOUCH5	MTDI	
	TOUCH6	MTMS	
	TOUCH7	GPIO27	
	TOUCH8	32K_XN	
	TOUCH9	32K_XP	
JTAG	MTDI	MTDI	JTAG for software debugging
	MTCK	MTCK	
	MTMS	MTMS	
	MTDO	MTDO	

Convertidor A/D ESP32.

8 canales de entrada

Sensor hall

10 canales de entrada

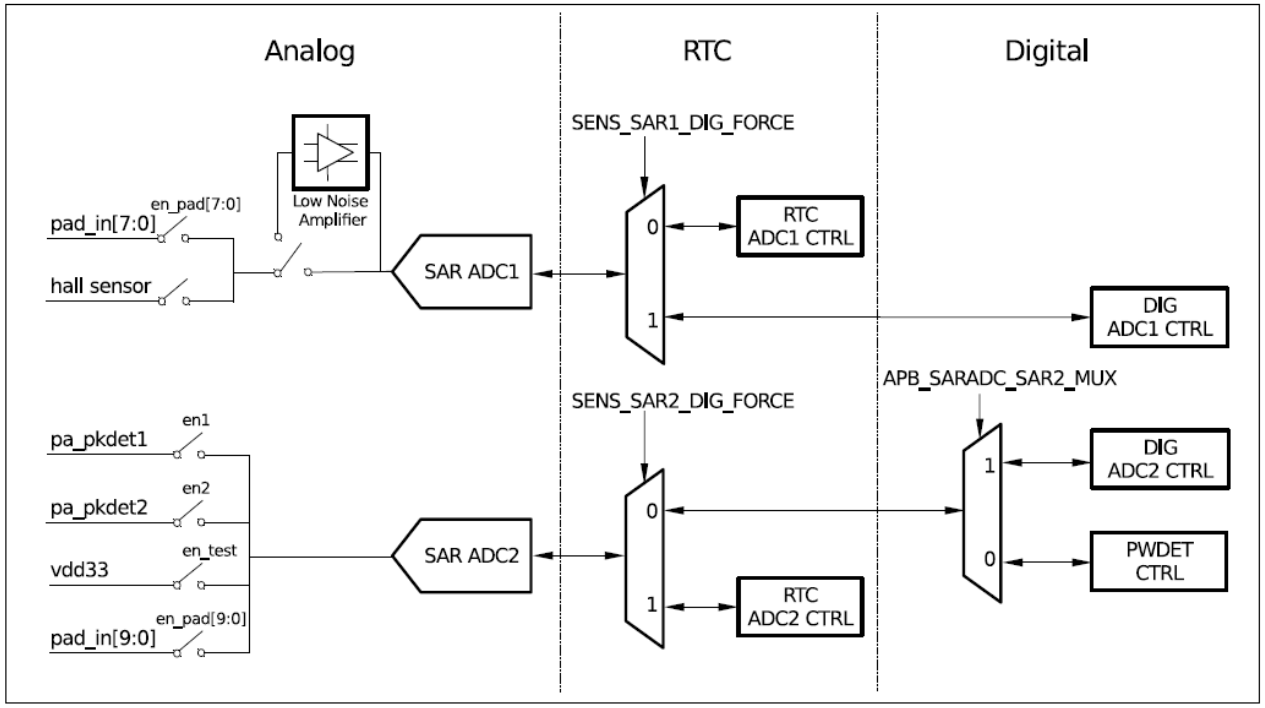


Figure 144: SAR ADC Outline of Function

Fuente: [esp32 technical refence manual](#) (capítulo 30.3)

Convertidor A/D ESP32.

Controladores del ADC

- RTC ADC1 CTRL y RTC ADC2 CTRL: controlan la medida del ADC con mínimo consumo de potencia a baja frecuencia, utilizando el coprocesador ULP. Se utilizan cuando:
  - el procesador está en Deep-sleep
  - se configura una medida en modo continuo

- DIG ADC1 CTRL y DIG ADC2 CTRL : controlan el ADC cuando se configura con una frecuencia de muestreo elevada
  - Modo de escaneado multi-canal: modo simple, doble o alterno
  - Posibilidad de iniciar el escaneado por sw o por el bus de comunicaciones I2S
  - Conexión a la DMA: posibilidad de generar una interrupción cuando el escaneado a terminado
- PWDET: detector de potencia o de pico (se usa para detectar señales Wifi)

Convertidor A/D ESP32.

Controlador DIG ADCx CTRL

Para configurar la lectura en cada instante.

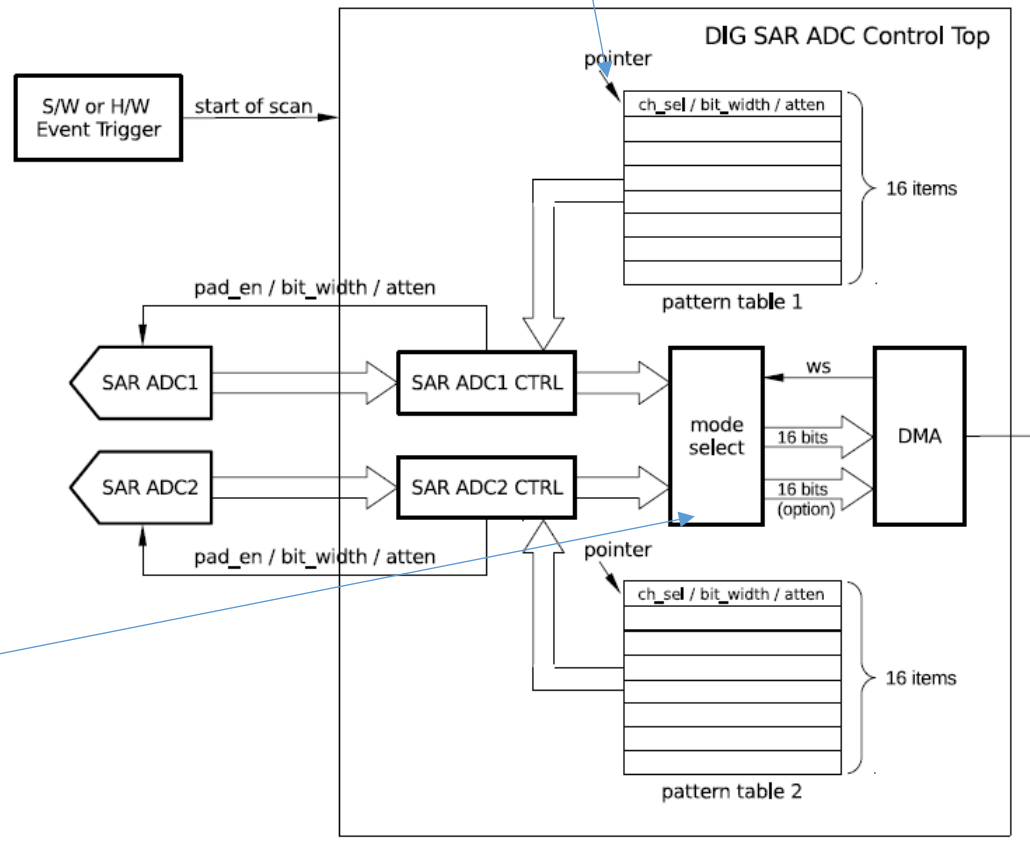


Figure 146: Diagram of DIG SAR ADC Controllers

Selección modo:

- Simple (solo un canal)
- Doble (dos canales a la vez)
- Alterno (cambia de uno a otro)

Produce interrupción cuando termina el escaneo

Fuente: esp32 technical refence manual (capítulo 30.3)

Convertidor A/D
ESP32. Funciones en Arduino

API Reference Arduino: https://espressif-docs.readthedocs-hosted.com/projects/arduino-esp32/en/latest/api/adc.html

- analogRead(36); /* Get ADC value for pin */
- analogReadResolution(12); // Set the resolution of analogRead return values. Default is 12 bits (0-4095)
- analogSetWidth(12); // Sets the sample bits and read resolution* Default is 12bit (0 - 4095) * Range is 9 - 12
- analogSetCycles(8); // Set number of cycles per sample * Default is 8 and seems to do well * Range is 1 – 255
- analogSetClockDiv(1); // * Set the divider for the ADC clock. * Default is 1 * Range is 1 - 255
- analogSetAttenuation(ADC_11db); //ADC_0db, ADC_2_5db, ADC_6db, ADC_11db * Set attenuation for all channels * Default is 11db
- analogSetPinAttenuation(36, ADC_0db); //ADC_0db, ADC_2_5db, ADC_6db, ADC_11db *Set attenuation for particular pin *Default is 11db
- hallRead(); //* Get value for HALL sensor (without LNA) * connected to pins 36(SVP) and 39(SVN)
- adcAttachPin(36); // * Attach pin to ADC (will also clear any other analog mode that could be on)
- adcStart(36); //* Start ADC conversion on attached pin's bus
- adcBusy(uint8_t pin); //* Check if conversion on the pin's ADC bus is currently running
- adcEnd(uint8_t pin); // * Get the result of the conversion (will wait if it have not finished)

Convertidor A/D
ESP32. Programación con esp-idf

- API Reference > Peripherals > ADC

https://docs.espressif.com/projects/esp-idf/en/release-v4.1/api-reference/peripherals/adc.html

- ADC1: configurar la precisión y la atenuación

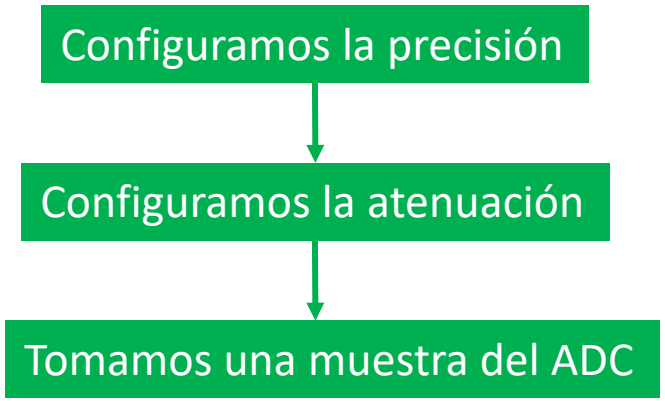
adc1_config_width() and adc1_config_channel_atten()

- ADC2 Configurar sólo la atenuación ya que la precisión se configura en cada lectura

adc2_config_channel_atten()

Convertidor A/D
ESP32. Programación con esp-idf

Ejemplo ADC1 canal 0 GPIO36 precisión 12bits (entrada desde 0 V hasta 1.1 V (0dB de atenuación)



https://docs.espressif.com/projects/esp-idf/en/release-v4.1/api-reference/peripherals/adc.html

Convertidor A/D
ESP32. Programación con esp-idf

- Ejemplo ADC1 canal 0 GPIO36 precisión 12bits (entrada desde 0 V hasta 1.1 V (0dB de atenuación)

```
#include <driver/adc.h>

...

    adc1_config_width(ADC_WIDTH_BIT_12);
    adc1_config_channel_atten(ADC1_CHANNEL_0,ADC_ATTEN_DB_0);
    int val = adc1_get_raw(ADC1_CHANNEL_0);
```

enum `adc_bits_width_t`

Values:

`ADC_WIDTH_BIT_9` = 0

ADC capture width is 9Bit

`ADC_WIDTH_BIT_10` = 1

ADC capture width is 10Bit

`ADC_WIDTH_BIT_11` = 2

ADC capture width is 11Bit

`ADC_WIDTH_BIT_12` = 3

ADC capture width is 12Bit

enum `adc_atten_t`

V_{REF}=1.1V

Values:

`ADC_ATTEN_DB_0` = 0

The input voltage of ADC will be reduced to about 1/1

Rango [0, 1.1V[

`ADC_ATTEN_DB_2_5` = 1

The input voltage of ADC will be reduced to about 1/1.34

Rango [0, 1.47V[

`ADC_ATTEN_DB_6` = 2

The input voltage of ADC will be reduced to about 1/2

Rango [0, 2.2V[

`ADC_ATTEN_DB_11` = 3

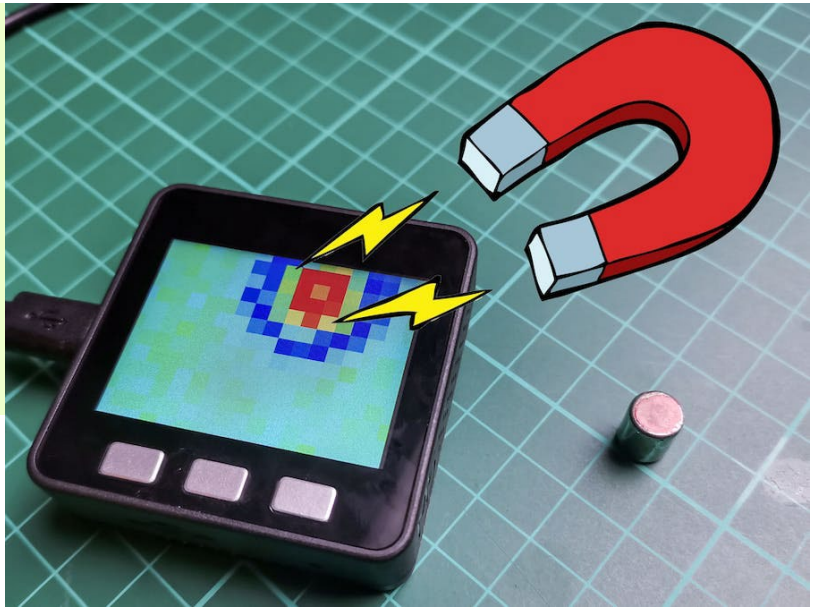
The input voltage of ADC will be reduced to about 1/3.6

Rango [0, 3.96V[

Convertidor A/D ESP32.

- Ejemplo ADC1 lectura sensor efecto hall interno con precisión 12bits

```
#include <driver/adc.h>

...

    adc1_config_width(ADC_WIDTH_BIT_12);
    int val = hall_sensor_read();
```



Aplicación: https://www.hackster.io/hague/where-on-earth-is-the-hall-effect-sensor-of-the-esp32-21d7b3

Convertidor A/D ESP32.

- ADC2 (más prioridad)
  - 10 canales (GPIO 0, 2, 4, 12 a 15 y 25 a 27)
  - Se utiliza para WiFi(sólo usar si no se usa WiFi) esp_wifi_stop()
  - Configurar la atenuación (la precisión se hace con cada lectura)
    - adc2_config_channel_atten()
  - Lectura adc2_get_raw()
  - Ejemplo: configurar el canal 7 del ADC2 (GPIO27) para leer datos con 12 bits de precisión

```
#include <driver/adc.h>

...

    int read_raw;
    adc2_config_channel_atten( ADC2_CHANNEL_7, ADC_ATTEN_0db );

    esp_err_t r = adc2_get_raw( ADC2_CHANNEL_7, ADC_WIDTH_12Bit, &read_raw);
    if ( r == ESP_OK ) {
        printf("%d\n", read_raw );
    } else if ( r == ESP_ERR_TIMEOUT ) {
        printf("ADC2 used by Wi-Fi.\n");
    }
```

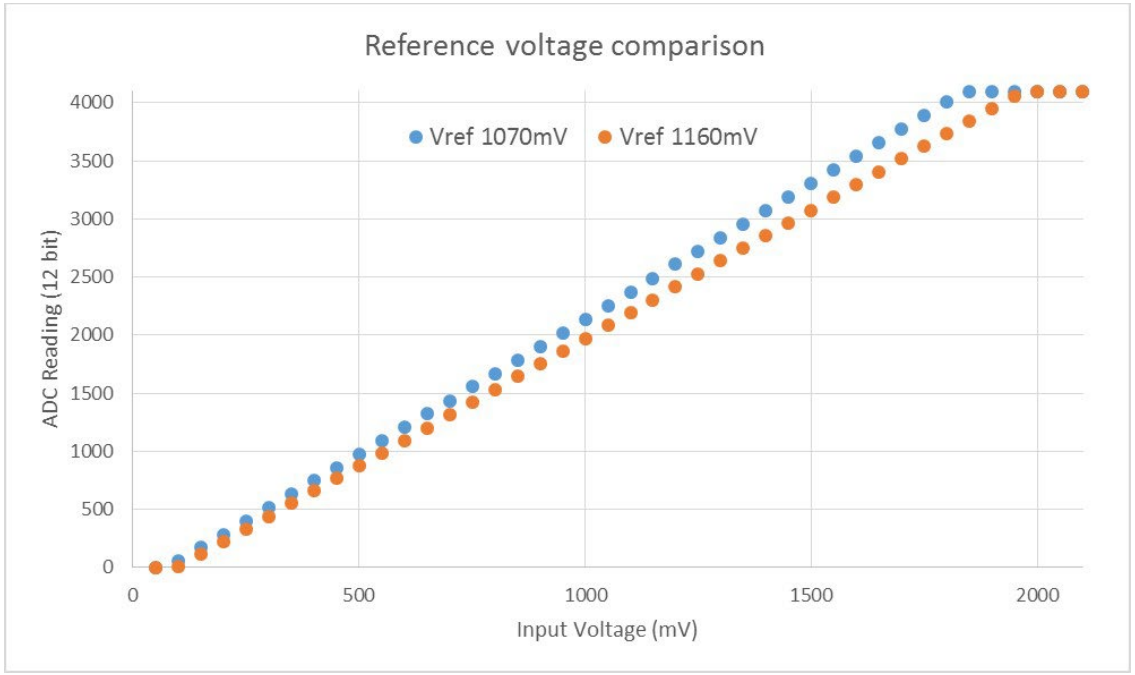
Convertidor A/D ESP32. Calibración

- El driver del ADC posee una función que permite calibrar el ADC
- Normalmente V_{REF}=1100mV, pero puede variar desde 1000mV hasta 1200mV entre diferentes ESP32
- Al calibrarlo se calculan los parámetros (coeff_a y coeff_b) que habría que aplicar para corregir la desviación

$$y = coeff\_a * x + coeff\_b$$

Lectura del ADC

Valor en mV calibrado



Curvas que caracterizan dos ADCs con diferentes valores de referencia

Convertidor A/D
ESP32. Calibración

Función para caracterizar la curva del ADC y obtener los parámetros:

```
esp_adc_cal_value_t esp_adc_cal_characterize(adc_unit_t adc_num,
                                             adc_atten_t atten,
                                             adc_bits_width_t bit_width,
                                             uint32_t default_vref,
                                             esp_adc_cal_characteristics_t *chars);
```

Función para aplicar los parámetros de calibración y obtener la lectura en mV

```
/* Converts Raw ADC Reading To Calibrated Value & Return the results in mV */
uint32_t esp_adc_cal_raw_to_voltage(uint32_t adc_reading, const esp_adc_cal_characteristics_t *chars);
```

Función para aplicar los parámetros de calibración y obtener la lectura en mV a través de un puntero al dato

```
/* Reads an ADC channel, calibrate the result, and save it in the voltage pointer (in mV unit) */
esp_err_t esp_adc_cal_get_voltage(adc_channel_t channel, const esp_adc_cal_characteristics_t *chars, uint32_t *voltage);
```

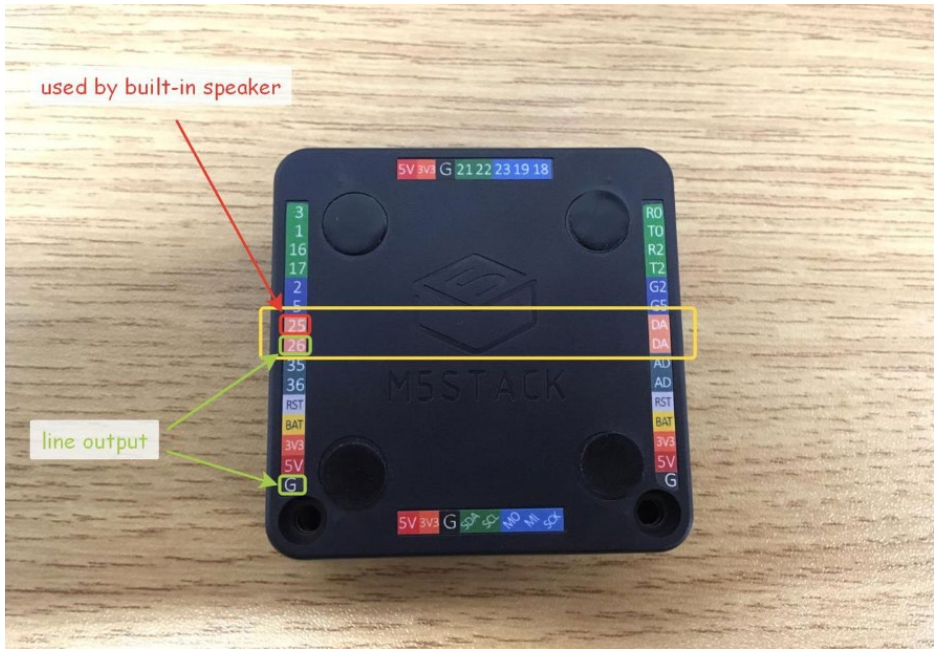
Convertidor A/D ESP32.

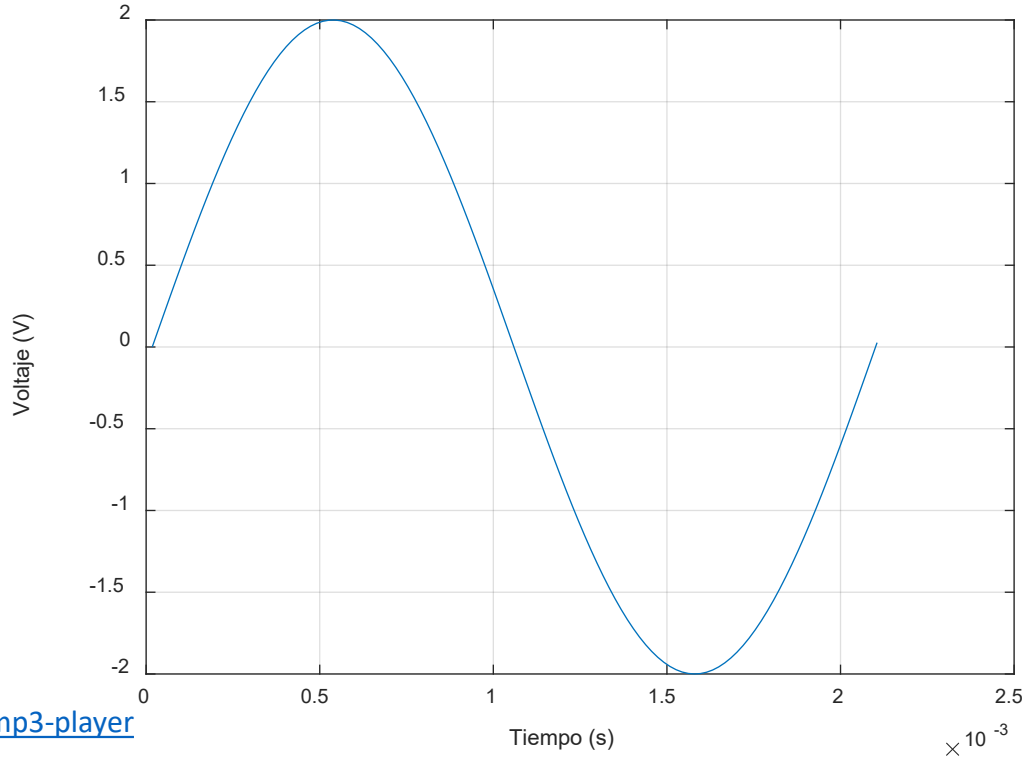
- Ejemplo lectura ADC2 canal 7 GPIO27

```
#include <driver/adc.h>

...

    int read_raw;
    adc2_config_channel_atten( ADC2_CHANNEL_7, ADC_ATTEN_0db );

    esp_err_t r = adc2_get_raw( ADC2_CHANNEL_7, ADC_WIDTH_12Bit, &read_raw);
    if ( r == ESP_OK ) {
        printf("%d\n", read_raw );
    } else if ( r == ESP_ERR_TIMEOUT ) {
        printf("ADC2 used by Wi-Fi.\n");
    }
```

Ejercicios

1- Generar una señal sinusoidal de amplitud 2Vpp y frecuencia 480Hz a través del convertidor DAC2 (DAC_CHANNEL_2) conectado al GPIO26, sabiendo que la frecuencia de reloj que ataca al DAC es 57kHz y que el DAC tiene una precisión de 8 bits.





Fuente: https://community.m5stack.com/topic/143/lesson-6-1-speaker-mp3-player

Ejercicios

2- ¿Qué valor leerá el ADC configurado con las instrucciones del programa, cuando se le aplica una entrada de 1.5V?

```
adc1_config_width(ADC_WIDTH_BIT_9);
adc1_config_cannel_att(ADC1_CHANNEL_0, ADC_ATTEN_DB_6);
```

Bibliografía

- Programación ESP32 con Arduino:
  - Convertidor D/A https://espressif-docs.readthedocs-hosted.com/projects/arduino-esp32/en/latest/api/dac.html
  - Convertidor A/D https://espressif-docs.readthedocs-hosted.com/projects/arduino-esp32/en/latest/api/adc.html
- Programación ESP32 con ESP-IDF
  - Convertidor D/A https://docs.espressif.com/projects/esp-idf/en/release-v4.1/api-reference/peripherals/dac.html
  - Convertidor A/D https://docs.espressif.com/projects/esp-idf/en/release-v4.1/api-reference/peripherals/adc.html
- Manual de referencia del ESP32: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf