

Señales analógicas/digitales y conversor ADC

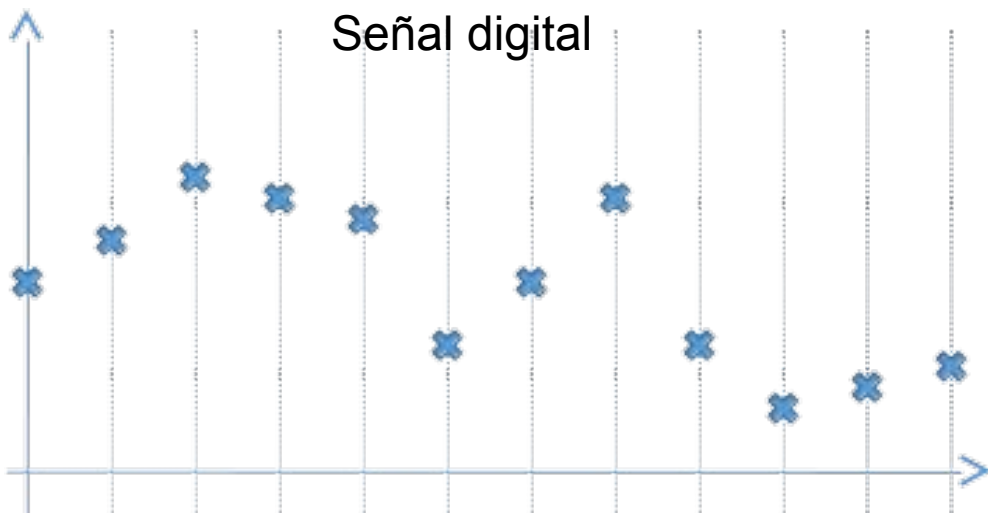
Informática II - R2004
2021

Señales analógicas y digitales

Señal analógica



Señal digital



Una **señal analógica** es la representación TEMPORAL de un fenómeno físico (temperatura, presión, humedad, etc.)

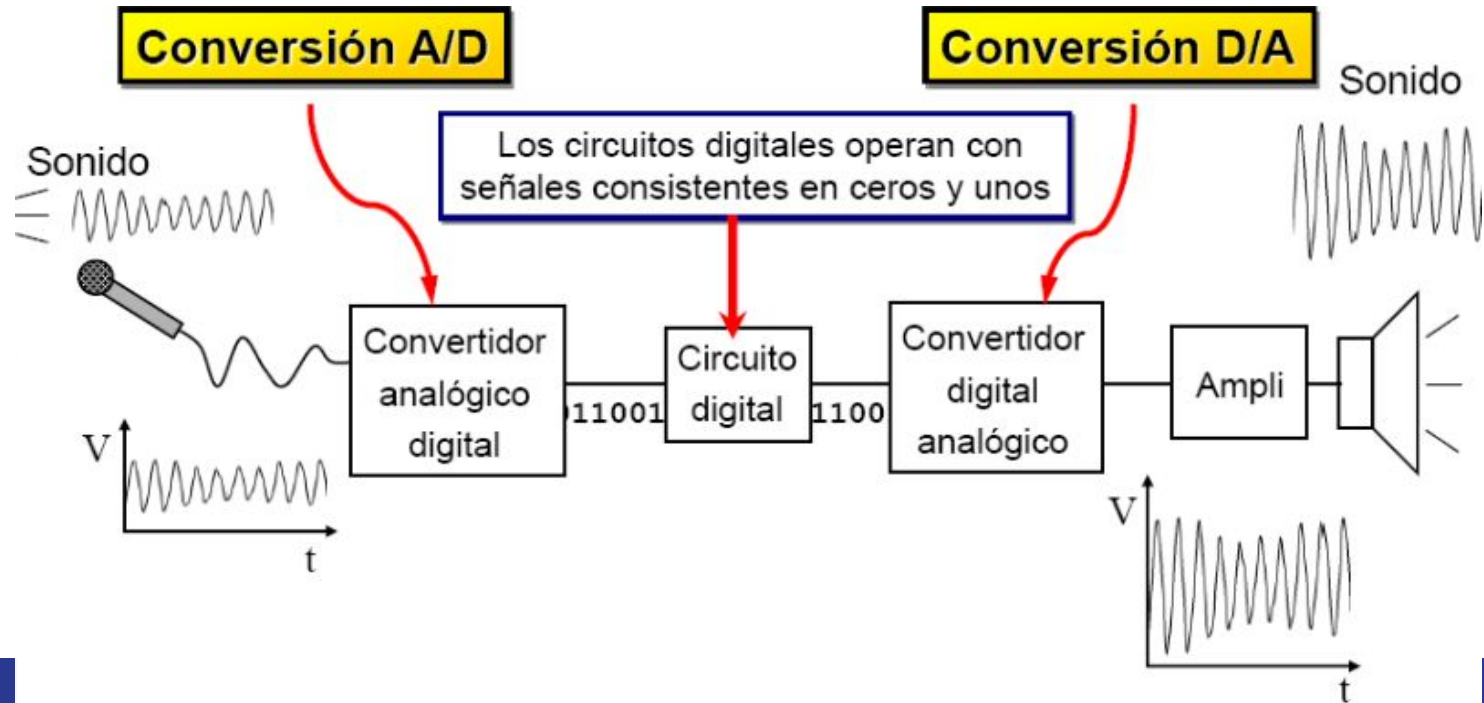
Es **continua**, tanto en el tiempo (no hay ningún instante en que la señal no tenga un valor), como en amplitud (la señal puede tener infinitos valores posibles en un instante)

Para poder almacenarla en una memoria FINITA, la señal debe **digitalizarse**.

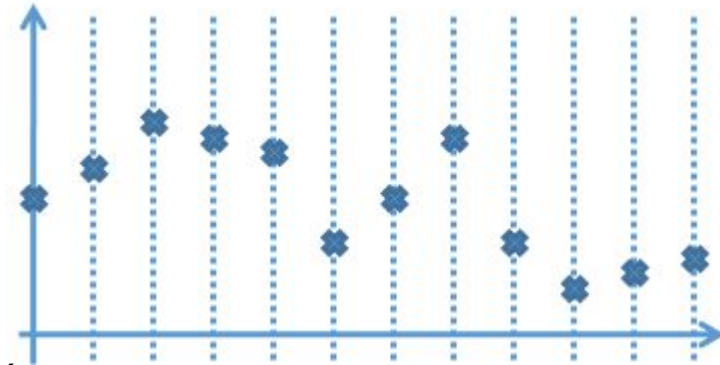
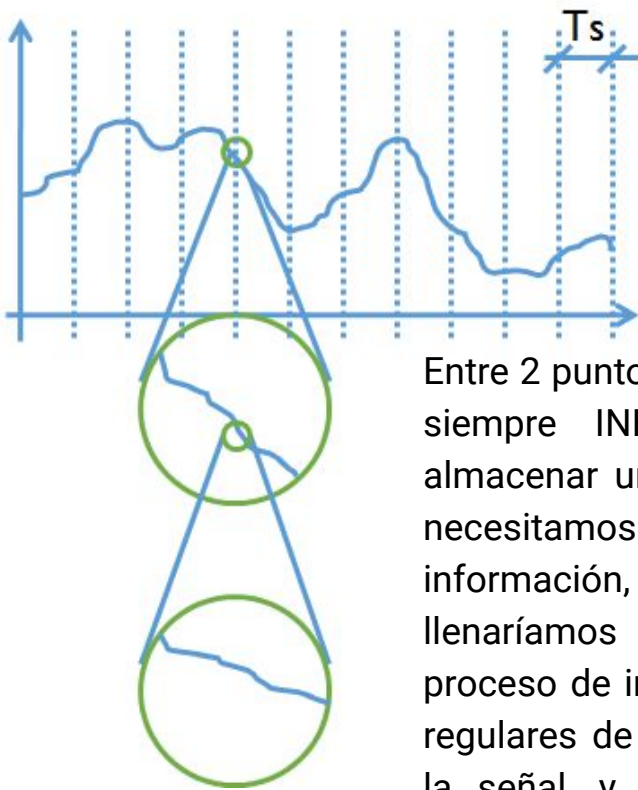
Una **señal digital** es discreta, tanto en el tiempo (es un conjunto de valores tomados cada un período de tiempo T), como en amplitud (su amplitud es un valor entero finito)

Conversor Analógico/Digital y Digital/Analógico

Para poder trabajar en forma digital las señales y luego devolverlas al mundo analógico, se necesitan de dispositivos llamados CONVERSORES AD (analógico a digital) o DA (digital a analógico). En inglés ADC y DAC.



¿Cómo digitalizo una señal? Proceso de Muestreo



La señal es ahora un vector de N valores, que puede almacenarse en memoria

Entre 2 puntos cualesquiera de la señal habrá siempre INFINITOS valores. Para poder almacenar una señal en el microcontrolador necesitamos poder acotar esta cantidad de información, ya que de otra manera llenaríamos la memoria casi al instante. El proceso de ir tomando muestras a intervalos regulares de tiempo se llama MUESTREO de la señal, y nos permite acotar una señal continua a un conjunto de valores discretos.

Frecuencia de Muestreo

- Para poder almacenar la señal en el uC, hay que MUESTREARLA. Esto implica tomar muestras a un período regular de tiempo (T_s)
- A la inversa del tiempo de muestreo (T_s) se denomina FRECUENCIA DE MUESTREO (f_s)
- Para no perder información de la señal, la frecuencia de muestreo debe ser más alta que la máxima frecuencia de la señal analógica (por lo menos el doble)
- La frecuencia de una señal está relacionada con la velocidad a la que cambia de valor



Un poquito de análisis de señales...

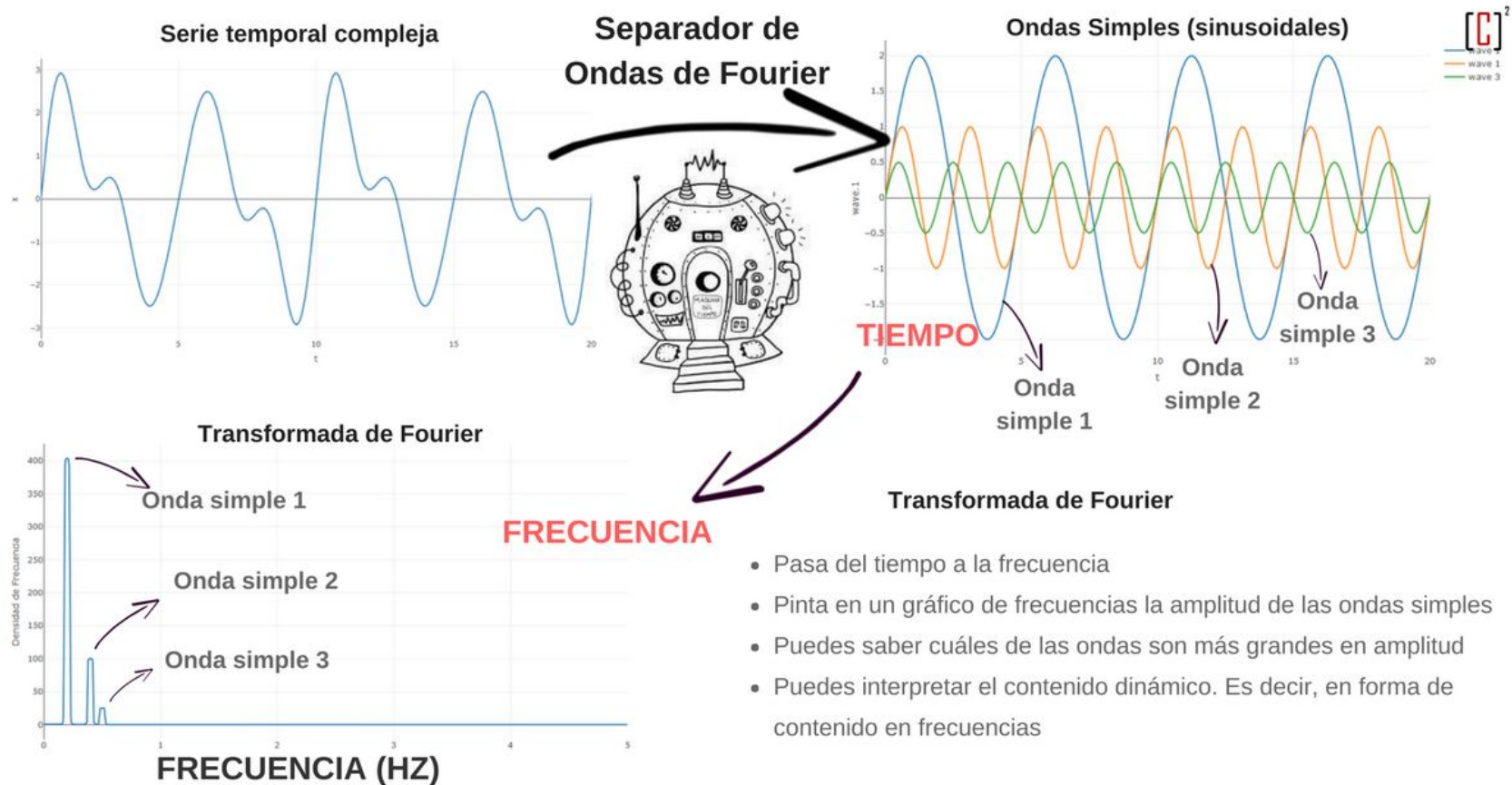
Toda señal periódica se puede descomponer como una combinación (suma) de señales senoidales, de diferente FRECUENCIA y AMPLITUD. Las señales de mayor frecuencia son los componentes que generan transiciones más rápidas en la señal.

Pensadas de esta manera, las señales tienen varios componentes distintos, pero finalmente habrá componentes hasta una determinada frecuencia máxima, que serán las causantes de las transiciones más rápidas de la señal. Identificar estas componentes es a lo que nos referimos cuando hablamos de conocer la máxima frecuencia de la señal.

La frecuencia de muestreo deberá ser al menos el doble de la componente senoidal de máxima frecuencia

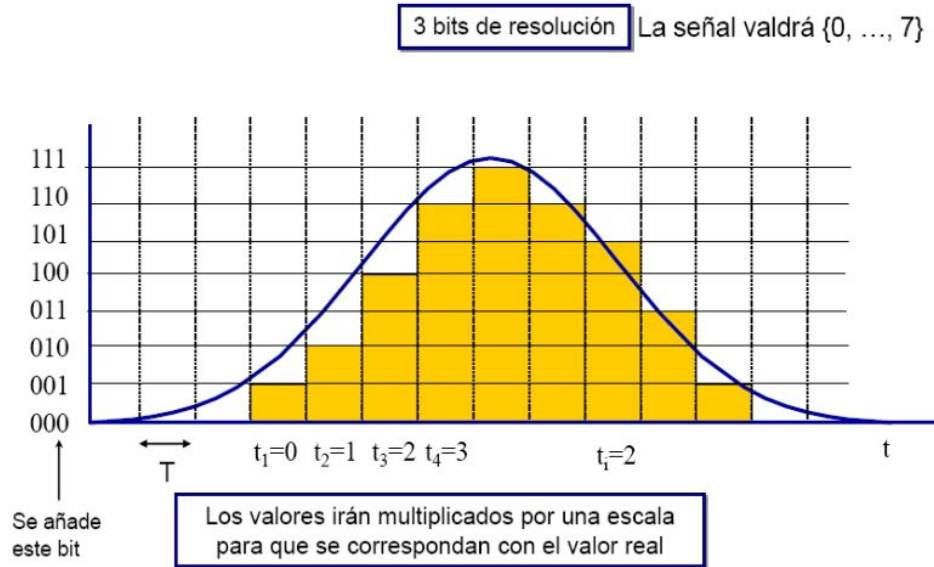
TODO ESTO TOMA RELEVANCIA CUANDO TRABAJAMOS CON SEÑALES DE ALTA FRECUENCIA (audio, vibraciones, señales electromagnéticas). Los fenómenos físicos con los que solemos trabajar en info 2 no tienen mayores limitaciones de frecuencia, porque son señales de progresión lenta (temperatura, peso, humedad, etc.). Este tipo de señales pueden ser muestreadas a intervalos mayores de tiempo

Lo mismo, gráficamente



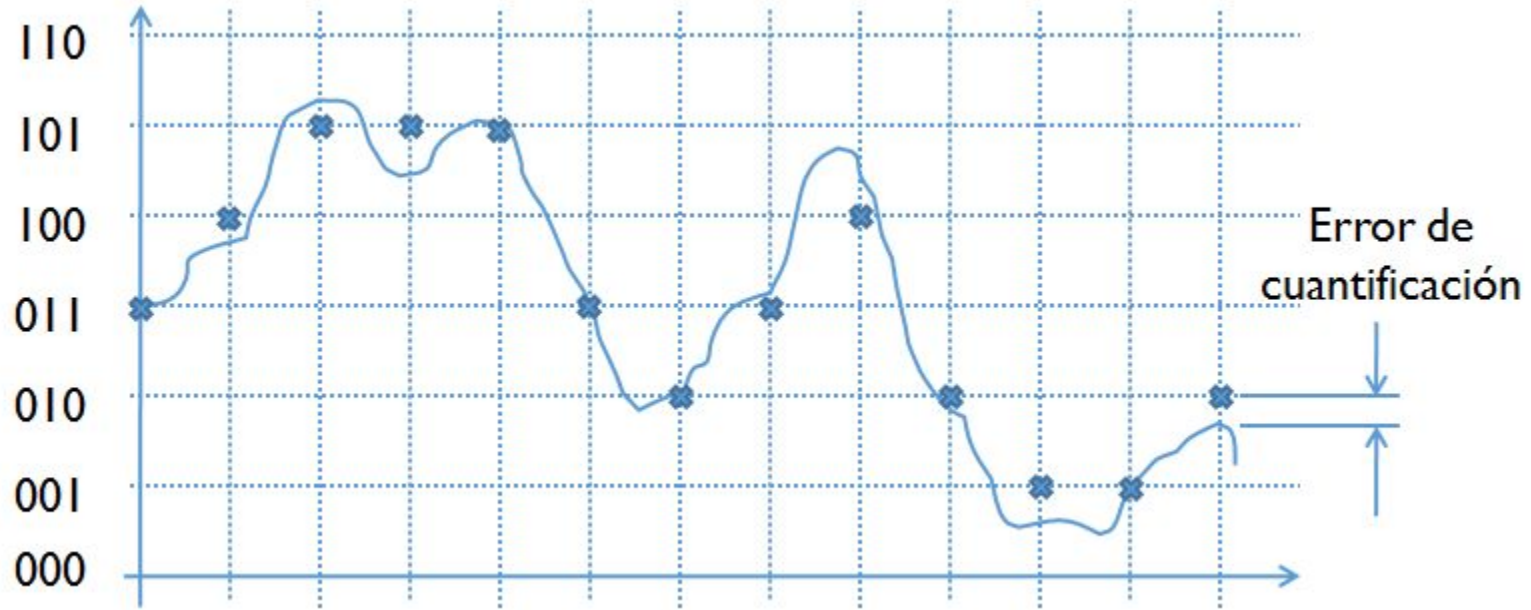
Digitalización en amplitud: Resolución

De la misma manera que al muestrear la señal obtengo una serie de valores discretos en el TIEMPO, voy a tener que digitalizar también la AMPLITUD de la señal. La **resolución** (cantidad de bits) del conversor A/D es la cantidad de saltos (finitos) que el conversor puede hacer entre un valor y el siguiente. Mientras mayor sea la resolución, menor será el **error de muestreo**.



Muestreo: Conversión de una señal continua a una discreta

Cuantificando
con 3 bits tengo
8 niveles =>



La señal original quedará entonces muestreada como se ve en los puntos, convirtiéndose así en un vector de valores enteros (cuentas), que para este período de tiempo será:

$\text{Senial}[] = \{ 3, 4, 5, 5, 5, 3, 2, 3, 4, 2, 1, 1, 2 \}$

Cuantificación

111	←	$V_{ref} * 7/8$
110	←	$V_{ref} * 6/8$
101	←	$V_{ref} * 5/8$
100	←	$V_{ref} * 4/8$
011	←	$V_{ref} * 3/8$
010	←	$V_{ref} * 2/8$
001	←	$V_{ref} * 1/8$
000	←	0v



Para reconstruir la señal (si quiero saber qué valor de tensión se corresponde con una determinada cantidad de CUENTAS), voy a dividir el RANGO DINÁMICO del ADC por la cantidad de cuentas que posee el mismo (si es un ADC de n bits, la cantidad de cuentas es 2^n). Me queda como se muestra en el cuadro:

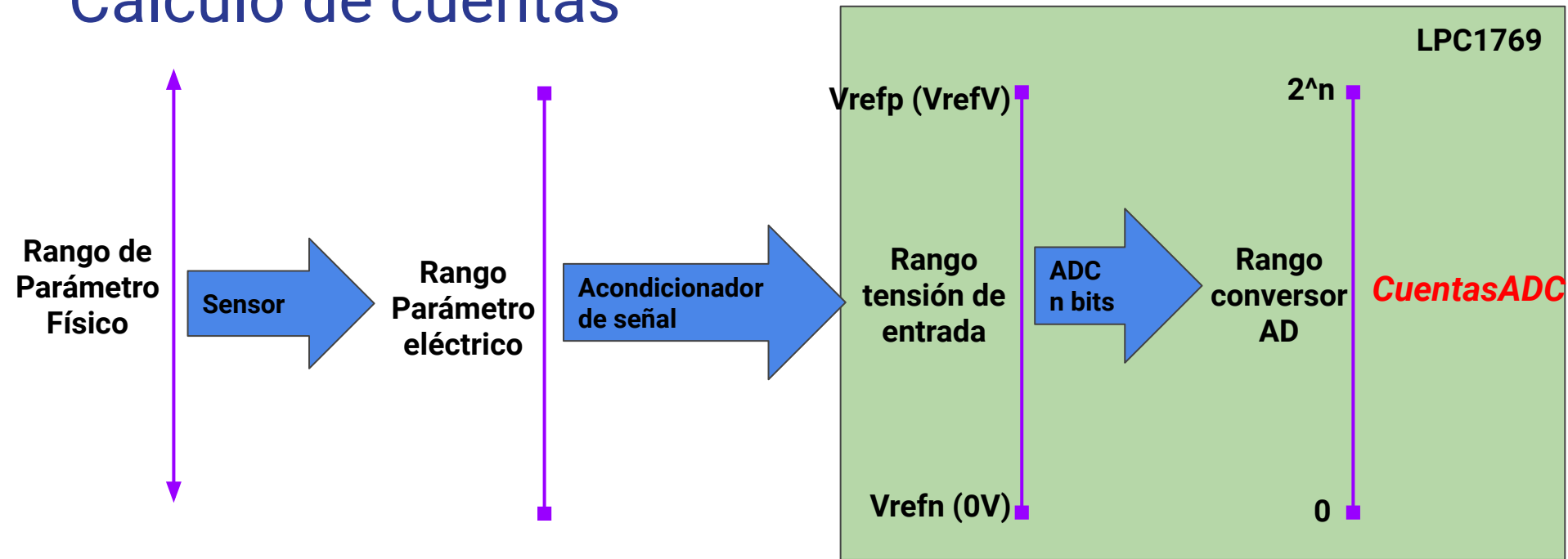
$$1 \text{ LSB} = V_{ref} / 2^n$$

En caso que V_{refMin} sea distinto de 0v

$$1 \text{ LSB} = (V_{refMax} - V_{refMin}) / 2^n$$

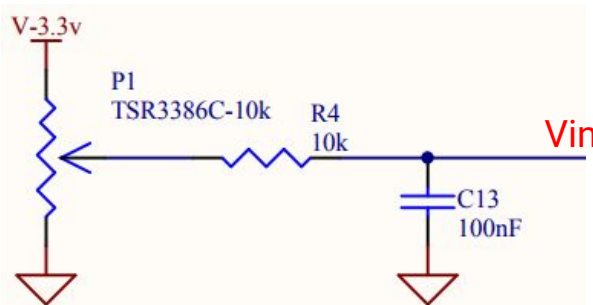
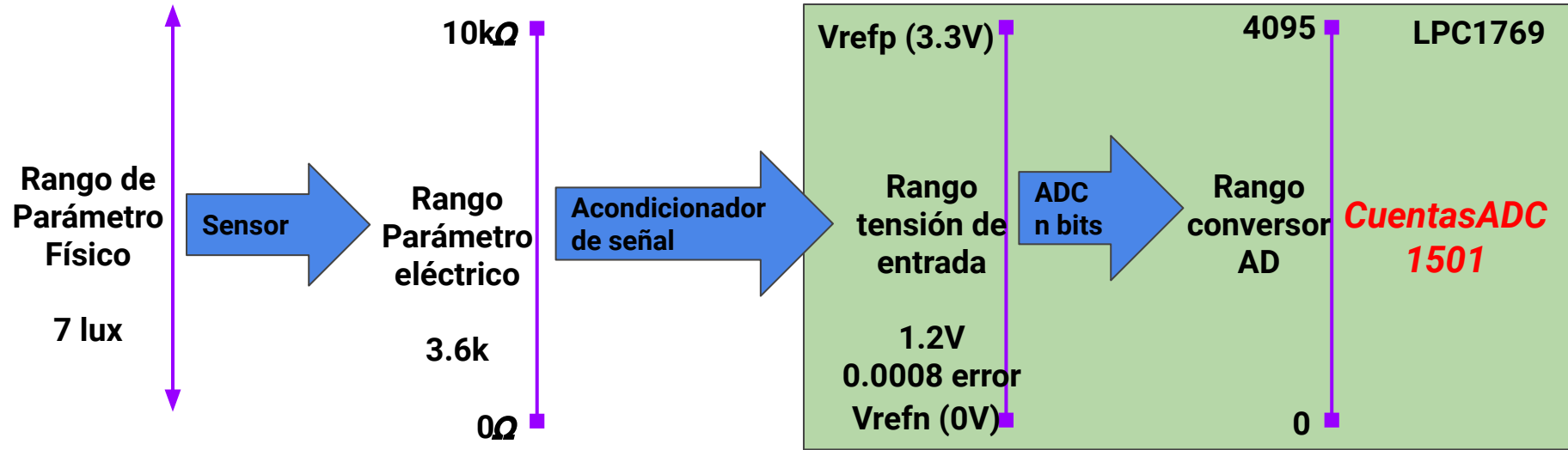


Cálculo de cuentas



$$CuentasADC = \frac{V_{in} * 2^n}{V_{ref}}$$

Cálculo de Cuentas (Ejemplo Potenciómetro)

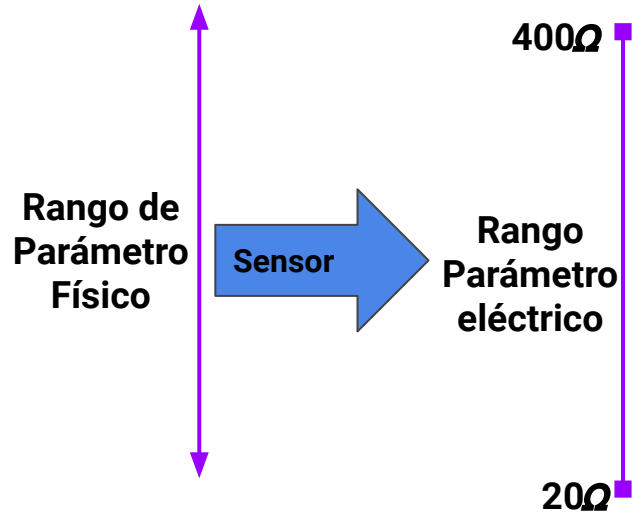


$$V_{in} = \frac{V_{cc} * R_x}{R_{tot}}$$

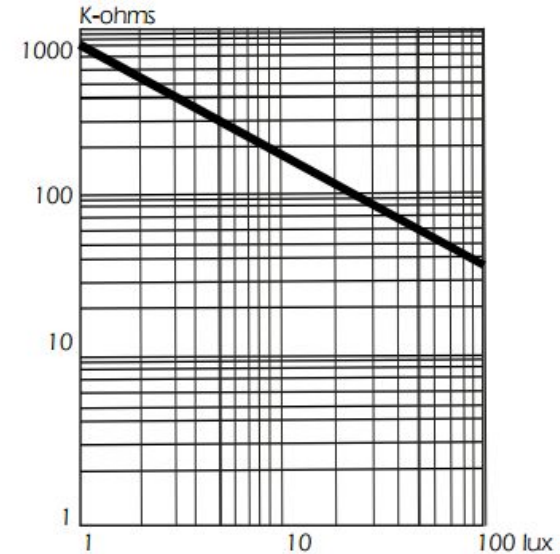
$$CuentasADC = \frac{V_{in} * 2^n}{V_{ref}}$$

$$CuentasADC = \frac{\frac{V_{cc} * R_x}{R_{tot}} * 2^n}{V_{ref}} \quad R_x = \frac{CuentasADC * V_{ref} * R_{tot}}{2^n * V_{cc}}$$

Calculo de cuentas con un sensor



$$R_X = \frac{\text{Cuentas}_{ADC} * V_{ref} * R_{tot}}{2^n * V_{CC}}$$



Resistencia de la célula en función de la iluminancia

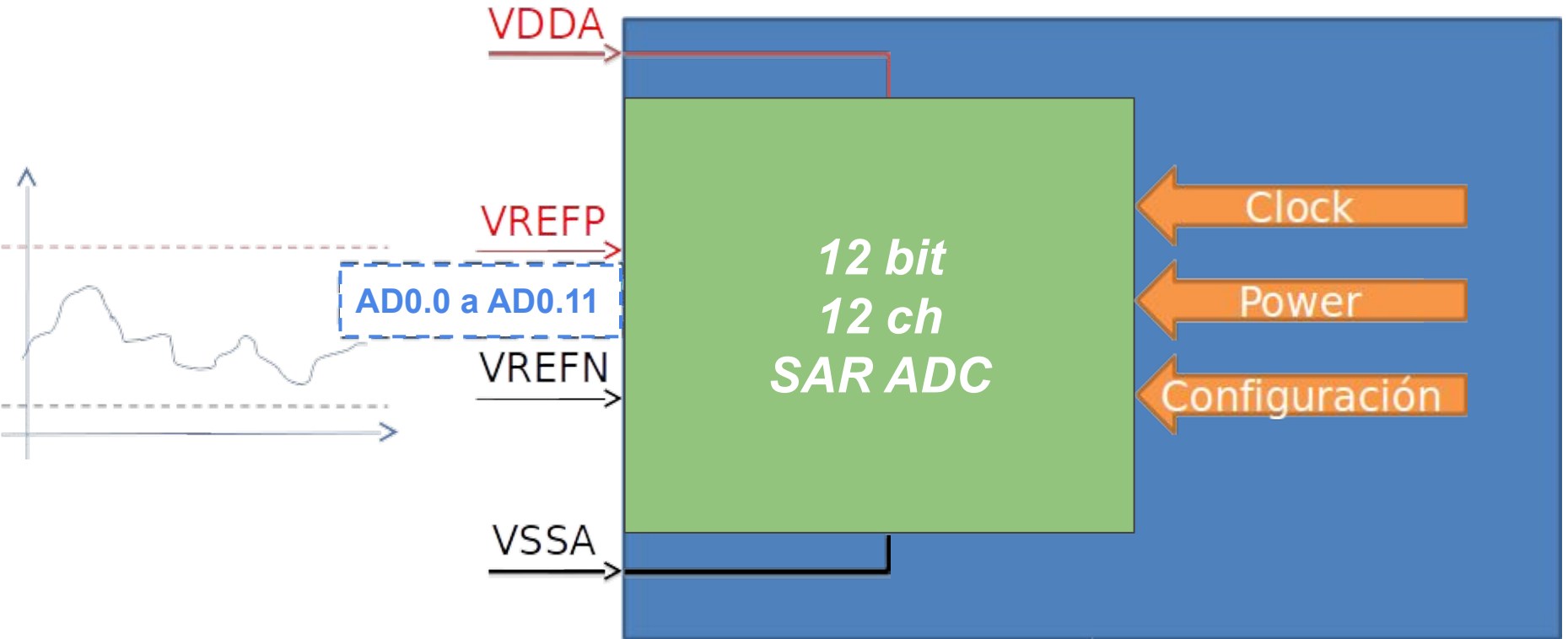
El gráfico se obtiene de la hoja de datos del sensor, en este caso es una resistencia variable que los valores de la resistencia equivalen a los valores de iluminancia que indica el gráfico.

ADC y DAC en el LPC845

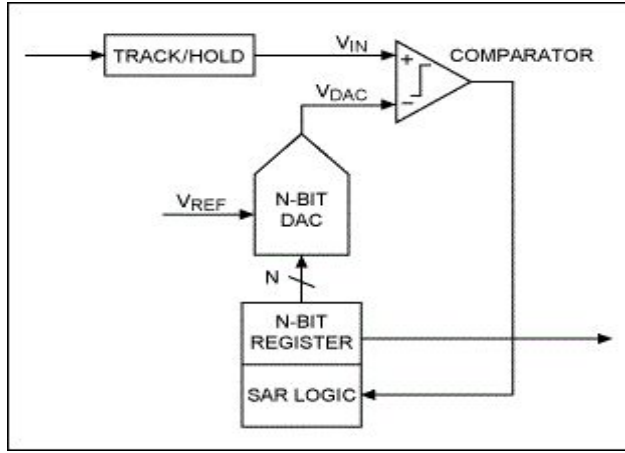
El LPC845 tiene solo 1 ADC, pero que posee 12 canales. Esto significa que tendrá 12 entradas analógicas distintas, pero tiene que repartir su frecuencia de muestreo entre las 12 entradas si las uso al mismo tiempo. Si la señal que quiero muestrear es MUY rápida, no podré muestrear OTRAS señales (o tengo que usar otro ADC externo). La frecuencia de muestreo MÁXIMA del ADC es de 1,2MHz (repartida entre los canales que use). Tiene a su vez 2 DACs, que permiten una frecuencia máxima de 1MHz cada uno:

8	PMU	0x4002 0000
7	ADC	0x4001 C000
6	DAC1	0x4001 8000
5	DAC0	0x4001 4000

ADC del LPC845

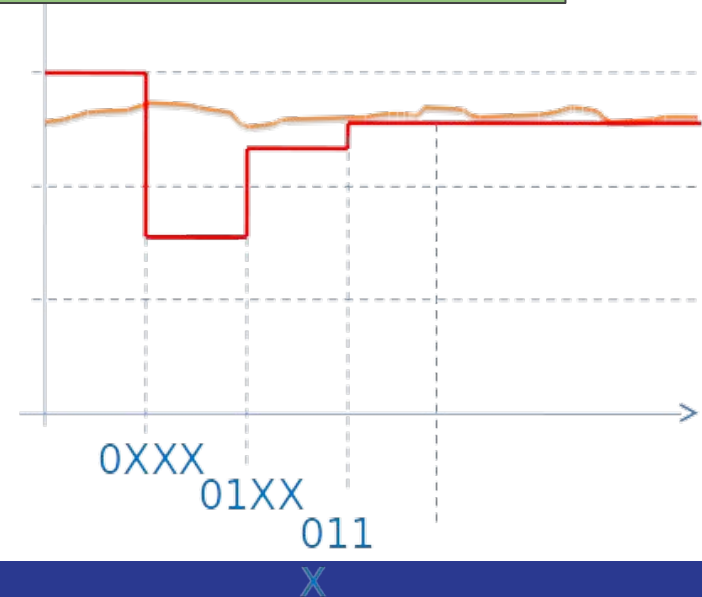


SAR - Registro de aproximaciones sucesivas



El ADC del LPC845 toma 25 ciclos de clock para garantizar que el valor del registro de salida contiene el valor de la entrada analógica

1. El circuito de S&H toma una muestra de la señal
2. La lógica del SAR pone la salida en la mitad del valor total
3. El comparador chequea si la señal es mayor o menor a la muestreada, si es mayor mantiene el 1 y pasa al siguiente bit, sino pone en 0



Pasos para configurar el ADC

1. Energizar el dispositivo (PDRUNCFG)
2. Habilitar los clocks y seleccionar su frecuencia (SYSAHBCLKCTRL)
3. Seleccionar los pines que se utilizarán para el ADC (PINENABLE)
4. Calibrar el ADC
5. Configurar las características del ADC y las secuencias



1. Energizar el ADC

Ciertos periféricos tienen la posibilidad de ser desenergizados, para disminuir el consumo. Esto se realiza mediante el registro PDRUNCFG, perteneciente al bloque de registros SYSCON:

Table 172. Power configuration register (PDRUNCFG, address 0x4004 8238) bit description

Bit	Symbol	Value	Description	Reset value
0	FROOUT_PD		FRO oscillator output power	0
		0	Powered	
		1	Powered down	
1	FRO_PD		FRO oscillator power down	0
		0	Powered	
		1	Powered down	
2	FLASH_PD		Flash power down	0
		0	Powered	
		1	Powered down	
3	BOD_PD		BOD power down	1
		0	Powered	
		1	Powered down	
4	ADC_PD		ADC wake-up configuration	1
		0	Powered	
		1	Powered down	
5	SYSCON_PD		System controller power down	1

```
//Energizo el ADC  
SYSCON->PDRUNCFG &= ~(1<<4);
```

2. Habilitar los clocks y seleccionar su frecuencia

Table 146. System clock control 0 register (SYSAHBCLKCTRL0, address 0x4004 8080) bit description

Bit	Symbol	Value	Description	Reset value
0	SYS		Enables the clock for the AHB, the APB bridge, the Cortex-M0+ core clocks, SYSCON, and the PMU. This bit is read only and always reads as 1.	1
1	ROM		Enables clock for ROM.	1
		0	Disable	
		1	Enable	
		0	Disable	
		1	Enable	
24	ADC		Enables clock to ADC.	0
		0	Disable	
		1	Enable	
25	CTIMER0		Enables clock for CTIMER0	0
		0	Disable	

//Habilito el CLK del ADC:

SYSCON->SYSAHBCLKCTRL0 |= 1<<24;

//Selecciono el clk del ADC a 30MHz:

SYSCON->ADCCLKSEL = 0;

//Sin divisor:

SYSCON->ADCCLKDIV = 0;

3. Seleccionar los pines que se utilizarán para el ADC

Table 195. Pin enable register 0 (PINENABLE0, address 0x4000 C1C0) bit description

Bit	Symbol	Value	Description	Reset value
14	ADC_0		ADC_0 function select.	1
		0	ADC_0 enabled on pin PIO0_7.	
		1	ADC_0 disabled.	
15	ADC_1		ADC_1 function select.	1
		0	ADC_1 enabled on pin PIO0_6.	
		1	ADC_1 disabled.	
16	ADC_2		ADC_2 function select.	1
		0	ADC_2 enabled on pin PIO0_14.	
		1	ADC_2 disabled.	
17	ADC_3		ADC_3 function select.	1
		0	ADC_3 enabled on pin PIO0_23.	
		1	ADC_3 disabled.	
18	ADC_4		ADC_4 function select.	1
		0	ADC_4 enabled on pin PIO0_22.	
		1	ADC_4 disabled.	
19	ADC_5		ADC_5 function select.	1
		0	ADC_5 enabled on pin PIO0_21.	
		1	ADC_5 disabled.	
20	ADC_6		ADC_6 function select.	1
		0	ADC_6 enabled on pin PIO0_20.	

```
//Primero habilito el clk de la SWM:  
SYSCON->SYSAHBCLKCTRL0 |= 1<<7;
```

```
//Habilito la entrada ADC1, que se  
encuentra como función especial en  
el pin P0.6:  
PINENABLE0 &= ~(1<<15);
```

4. Calibrar el ADC

El manual indica que cada vez que se reinicie el microcontrolador, se debe realizar una secuencia de calibración. Esto se hace poniendo un divisor de frecuencia que haga que el ADC funcione a 500kHz, y poniendo un 1 en el bit de calibración. Luego se debe esperar a que este bit se ponga en estado bajo para garantizar que el ADC se encuentre calibrado.

Table 453. A/D Control Register (CTRL, addresses 0x4001 C000) bit description

Bit	Symbol	Value	Description
7:0	CLKDIV		The system clock is divided by this value plus one to produce the sampling clock. The sampling clock should be less than or equal to 30 MHz for 1.2 Msamples/s. Typically, software should program the smallest value in this field that yields this maximum clock rate or slightly less, but in certain cases (such as a high-impedance analog source) a slower clock may be desirable.
8	ASYNMODE		Asynchronous operation mode
29:11			Reserved, do not write ones to reserved bits.
30	CALMODE		Writing a 1 to this bit initiates a self-calibration cycle. This bit will be automatically cleared by hardware after the calibration cycle is complete. To calibrate the ADC, set the ADC clock to 500 kHz. Remark: Other bits of this register may be written to concurrently with setting this bit, however once this bit has been set no further writes to this register are permitted until the full calibration cycle has ended.

```
//Activo el bit de calibracion  
(bit30) y pongo el divisor a 500kHz  
ADC->CTRL = (60 | (1<<30));
```

```
//Espero a que se termine la  
calibracion (bloqueante)  
while ( ADC->CTRL & (1<<30) );
```

¿Puedo hacer una instrucción bloqueante?

Configurando las características del ADC:

Frecuencia de muestreo

De acuerdo al manual, cada conversión toma 25 ciclos de clock. El ADC, a su vez, se alimenta con una frecuencia de 30MHz. Tengo que dividir entonces esta frecuencia para alcanzar una frecuencia 25 veces superior a la frecuencia de muestreo deseada:

Table 453. A/D Control Register (CTRL, addresses 0x4001 C000) bit description

Bit	Symbol	Value	Description
7:0	CLKDIV		The system clock is divided by this value plus one to produce the sampling clock. The sampling clock should be less than or equal to 30 MHz for 1.2 Msamples/s. Typically, software should program the smallest value in this field that yields this maximum clock rate or slightly less, but in certain cases (such as a high-impedance analog source) a slower clock may be desirable.
8	ASYNCMODE		Asynchronous operation mode

//Configuro el ADC:

```
ADC->CTRL = 11 | //DIV - 1
              (0 << 8 ) |
              (0 << 10) |
              (0 << 30);
```

Si quiero muestrear a 100kHz (por ejemplo):

$$30\text{MHz}/\text{DIVISOR} = 100\text{kHz} \times 25$$

$$\text{DIVISOR} = 30\text{MHz} / 2,5\text{MHz}$$

$$\text{DIVISOR} = 12$$

Configurando las características del ADC:

Modo sincrónico/asincrónico

Puedo seleccionar si quiero disparar una conversión ante un determinado evento (por ejemplo, cuando finalice un timer o se dispare una interrupción), o si quiero que el muestreo sea SINCRÓNICO (se muestree según la frecuencia configurada)

		high-impedance analog source) a slower clock may be desirable.		
8	ASYNCMODE	Asynchronous operation mode	0	
		Synchronous mode. The ADC clock is derived from the main system clock based on the divide value selected in the CLKDIV field. The ADC clock starts in response to a trigger to eliminate any uncertainty in the launching of an ADC conversion in response to any synchronous (on-chip) trigger. In synchronous mode with the SYNC-BYPASS bit set, sampling of the A/D input and start of a conversion initiates two system clocks after the leading edge of a (synchronous) trigger pulse.	0	
		Asynchronous mode. The ADC clock is based on an alternative independent clock source. The nature of this clock source and the mechanism for programming it is chip-specific. The frequency of this clock is limited to 15 MHz max. In addition, the ADC clock must never be faster than 10 times the APB bus clock rate.	1	
9		Reserved. Do not write a one to these bits.	0	

//Configuro el ADC:

ADC->CTRL = 11

(0 << 8)

(0 << 10)

(0 << 30);

//Sincrónico

Configurando las características del ADC: Low Power mode

Una forma de apagar o prender el ADC sin tener que volver a configurarlo, es ponerlo en modo bajo consumo. Este modo deja al ADC configurado y calibrado, pero apagado:

9	-	Reserved. Do not write a one to these bits.	0
10	LPWRMODE	Select low-power ADC mode. The analog circuitry is automatically powered-down when no conversions are taking place. When any (hardware or software) triggering event is detected, the analog circuitry is enabled. After the required start-up time, the requested conversion will be launched. Once the conversion completes, the analog-circuitry will again be powered-down provided no further conversions are pending. Using this mode can save an appreciable amount of current when conversions are required relatively infrequently. The penalty for using this mode is an approximately 15 ADC clock delay, based on the frequency specified in the CLKDIV field, from the time the trigger event occurs until sampling of the A/D input commences. Remark: This mode will NOT power-up the ADC when the ADC analog block is powered down in the system control block.	0
		0 Disabled. The low-power ADC mode is disabled. The analog circuitry remains activated even when no conversions are requested.	
		1 Enabled. The low-power ADC mode is enabled.	
29:11		Reserved, do not write ones to reserved bits.	0
30	CALMODE	Writing a 1 to this bit initiates a self-calibration cycle. This bit will be automatically cleared by hardware after the calibration cycle is complete. To calibrate the ADC, set the ADC clock to 500 kHz. Remark: Other bits of this register may be written to concurrently with setting this bit, however once this bit has been set no further writes to this register are permitted until the full calibration cycle has ended.	0

//Configuro el ADC:

```
ADC->CTRL = 11 |  
              (0 << 8) |  
              (0 << 10) | //LPM off  
              (0 << 30); //Calib off
```


Secuencias

El ADC del LPC845 muestrea las diferentes entradas armando “secuencias”. Puedo configurar 2 secuencias, cada una de las cuales muestreará diferentes canales, y generarán diferentes interrupciones. Esto me permite habilitar y deshabilitar varios canales todos juntos, y asignarles configuraciones grupales a todos ellos (frecuencia de muestreo menor, trigger independiente, etc). Si leemos una sola entrada, o si no nos sirve agrupar las entradas en diferentes secuencias, podemos seleccionar una sola secuencia y deshabilitar la otra.



Secuencias - Configuración:

Table 454. A/D Conversion Sequence A Control Register (SEQA_CTRL, address 0x4001 C008) bit description

Bit	Symbol	Value	Description	Reset value
11:0	CHANNELS		Selects which one or more of the twelve channels will be sampled and converted when this sequence is launched. A 1 in any bit of this field will cause the corresponding channel to be included in the conversion sequence, where bit 0 corresponds to channel 0, bit 1 to channel 1 and so forth. When this conversion sequence is triggered, either by a hardware trigger or via software command, A/D conversions will be performed on each enabled channel, in sequence, beginning with the lowest-ordered channel. Remark: This field can ONLY be changed while the SEQA_ENA bit (bit 31) is LOW. It is allowed to change this field and set bit 31 in the same write.	0x00
14:12	TRIGGER		Selects which of the available hardware trigger sources will cause this conversion sequence to be initiated. Program the trigger input number in this field. Remark: In order to avoid generating a spurious trigger, it is recommended writing to this field only when the SEQA_ENA bit (bit 31) is low. It is safe to change this field and set bit 31 in the same write.	0x0
17:15	-		Reserved.	-
18	TRIGPOL		Select the polarity of the selected input trigger for this conversion sequence. Remark: In order to avoid generating a spurious trigger, it is recommended writing to this field only when the SEQA_ENA bit (bit 31) is low. It is safe to change this field and set bit 31 in the same write.	0
		0	Negative edge. A negative edge launches the conversion sequence on the selected trigger input.	
		1	Positive edge. A positive edge launches the conversion sequence on the selected trigger input.	

Muestreo solo el canal 1 (P0.6) y
deshabilito otros triggers (solo "triggereos")
la conversión con el ADCCLK)

//Configuro la secuencia A:

```
ADC->SEQA_CTRL = (1 << 1) |  
(0 << 12) |  
(0 << 19) |  
(0 << 24) |  
(1 << 27) |  
(1 << 31);
```

Secuencias - Configuración:

19	SYNCBYPASS	<p>Setting this bit allows the hardware trigger input to bypass synchronization flip-flops stages and therefore shorten the time between the trigger input signal and the start of a conversion. There are slightly different criteria for whether or not this bit can be set depending on the clock operating mode:</p> <p>Synchronous mode: Synchronization may be bypassed (this bit may be set) if the selected trigger source is already synchronous with the main system clock (eg. coming from an on-chip, system-clock-based timer). Whether this bit is set or not, a trigger pulse must be maintained for at least one system clock period.</p> <p>Asynchronous mode: Synchronization may be bypassed (this bit may be set) if it is certain that the duration of a trigger input pulse will be at least one cycle of the ADC clock (regardless of whether the trigger comes from and on-chip or off-chip source). If this bit is NOT set, the trigger pulse must at least be maintained for one system clock period.</p>	0
		0 Enable synchronization. The hardware trigger bypass is not enabled.	
		1 Bypass synchronization. The hardware trigger bypass is enabled.	
25	-	Reserved	-
24-20	TSAMP	<p>The default sample period (TSAMP = "00000") at the beginning of each new conversion is 6.5 ADC clock periods. Depending on a variety of factors including ADC clock rate, output impedance of the analog source driver, ADC resolution, and the selection of channels, the sample time may need to be increased.</p> <p>The value programmed into the TSAMP fields dictates the number of additional ADC clock cycles (beyond 6.5) that the sample period will be extended by.</p> <p>Note: Any additional clocks of sample time inserted will add directly to the overall number of clocks required for a conversion, effectively reducing the overall conversion throughput rate.</p>	0
26	START	<p>Writing a 1 to this field will launch one pass through this conversion sequence. The behavior will be identical to a sequence triggered by a hardware trigger. Do not write 1 to this bit if the BURST bit is set.</p> <p>Remark: This bit is only set to a 1 momentarily when written to launch a conversion sequence. It will consequently always read-back as a zero.</p>	0
27	BURST	Writing a 1 to this bit will launch a burst of conversions. The behavior will be identical to a sequence triggered by a hardware trigger. Do not write 1 to this bit if the START bit is set.	0

**Modo sincrónico (CLKADC habilitado),
frecuencia de muestreo sin delay:**

//Configuro la secuencia A:

```
ADC->SEQA_CTRL = (1 << 1) |
                  (0 << 12) |
                  (0 << 19) |
                  (0 << 24) |
                  (1 << 27) |
                  (1 << 31);
```

Secuencias - Configuración:

		overall conversion throughput rate.	
26	START	Writing a 1 to this field will launch one pass through this conversion sequence. The behavior will be identical to a sequence triggered by a hardware trigger. Do not write 1 to this bit if the BURST bit is set. Remark: This bit is only set to a 1 momentarily when written to launch a conversion sequence. It will consequently always read-back as a zero.	0
27	BURST	Writing a 1 to this bit will cause this conversion sequence to be continuously cycled through. Other sequence A triggers will be ignored while this bit is set. Repeated conversions can be halted by clearing this bit. The sequence currently in progress will be completed before conversions are terminated.	0
28	SINGLESTEP	When this bit is set, a hardware trigger or a write to the START bit will launch a single conversion on the next channel in the sequence instead of the default response of launching an entire sequence of conversions. Once all of the channels comprising a sequence have been converted, a subsequent trigger will repeat the sequence beginning with the first enabled channel. Interrupt generation will still occur either after each individual conversion or at the end of the entire sequence, depending on the state of the MODE bit.	0
31	SEQA_ENA	Sequence Enable. In order to avoid spuriously triggering the sequence, care should be taken to only set the SEQA_ENA bit when the selected trigger input is in its INACTIVE state (as defined by the TRIGPOL bit). If this condition is not met, the sequence will be triggered immediately upon being enabled. 0 Disabled. Sequence A is disabled. Sequence A triggers are ignored. If this bit is cleared while sequence A is in progress, the sequence will be halted at the end of the current conversion. After the sequence is re-enabled, a new trigger will be required to restart the sequence beginning with the next enabled channel. 1 Enabled. Sequence A is enabled.	0

Modo BURST (habilita la conversión sucesiva del ADC a partir del ADC_CLK) y habilitación de la secuencia:

//Configuro la secuencia A:

```
ADC->SEQA_CTRL = (1 << 1) |
                  (0 << 12) |
                  (0 << 19) |
                  (0 << 24) |
                  (1 << 27) |
                  (1 << 31);
```

Interrupciones

Puedo habilitar una interrupción individual para el muestreo de cada secuencia, también puedo habilitar interrupciones en caso de que se supere un valor determinado de tensión en la entrada del ADC, o si se termina una conversión antes de que se haya leído la anterior. Todas estas fuentes son independientes, y hay que habilitarlas o deshabilitarlas en el periférico y en el NVIC:

Table 464. A/D Interrupt Enable register (INTEN, address 0x4001 C064) bit description

Bit	Symbol	Value	Description	Reset value
0	SEQA_INTEN		Sequence A interrupt enable.	0
		0	Disabled. The sequence A interrupt/DMA trigger is disabled.	
		1	Enabled. The sequence A interrupt/DMA trigger is enabled and will be asserted either upon completion of each individual conversion performed as part of sequence A, or upon completion of the entire A sequence of conversions, depending on the MODE bit in the SEQA_CTRL register.	
1	SEQB_INTEN		Sequence B interrupt enable.	0
		0	Disabled. The sequence B interrupt/DMA trigger is disabled.	
		1	Enabled. The sequence B interrupt/DMA trigger is enabled and will be asserted either upon completion of each individual conversion performed as part of sequence B, or upon completion of the entire B sequence of conversions, depending on the MODE bit in the SEQB_CTRL register.	
2	OVR_INTEN		Overrun interrupt enable.	0
		0	Disabled. The overrun interrupt is disabled.	

//Habilitacion en el periférico:
`ADC->INTEN = 1;`

//Habilitación en el NVIC:
`ISER0 |= (1 << 16);`

Inicialización del ADC(I):

```
void InitADC ( void )  
{  
    //Energizo el ADC  
    SYSCON->PDRUNCFG &= ~(1<<4);  
  
    //Habilito el CLK del ADC:  
    SYSCON->SYSAHBCLKCTRL0 |= 1<<24;  
  
    //Selecciono el clk del ADC a 30MHz  
    SYSCON->ADCCLKSEL = 0;  
  
    //Sin divisor:  
    SYSCON->ADCCLKDIV = 0;  
  
    //Para modificar la matriz de Switc  
    //primero habilito el clk:  
    SYSCON->SYSAHBCLKCTRL0 |= 1<<7;  
  
    //Habilito la entrada ADC1, que se  
    PINENABLE0 &= ~(1<<15);  
  
    //Calibración del ADC:  
    //Activo el bit de calibracion (bit  
    ADC->CTRL = (60 | (1<<30));  
    //Espero a que se termine la calibr  
    while ( ADC->CTRL & (1<<30) );  
}
```



Energizo el ADC



**Habilito el CLK
de 30MHz al
periférico**



**Selecciono el pin
P0.6 para tomar las
muestras**



Calibro el ADC

Inicialización del ADC(I):

```
ADC->CTRL = 119 |  
             (0 << 8) |  
             (0 << 10) |  
             (0 << 30);
```



Configuro el ADC

```
//Deshabilito la secuencia B:  
ADC->SEQB_CTRL = 0;  
  
//Selecciono VDD por encima de 2  
ADC->TRM &= ~(1<<5);  
  
//Configuro la secuencia A:  
ADC->SEQA_CTRL = (1 << 1) |  
                 (0 << 19) |  
                 (0 << 24) |  
                 (1 << 27) |  
                 (1 << 31);
```



**Configuro las
secuencias**

```
//Habilito la interrupcion por e  
ADC->INTEN = 1;  
  
//Habilito la interrupcion en el  
ISER0 |= (1 << 16);
```



**Habilito
interrupciones**

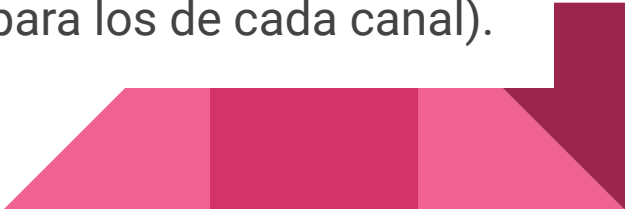
```
}
```


Utilizando el ADC - Lectura del resultado

Una vez que se haya completado una muestra, se generará la interrupción correspondiente, invocándose el handler `ADC_SEQA_IRQHandler`. En dicha interrupción, puedo leer el resultado de la conversión mediante dos registros:

- `SEQA_GDAT` o `SEQB_GDAT`, que contiene la ultima conversión de la secuencia A o B.
- `DATn` (n de 0 a 11), que contiene la última conversión del canal n del ADC

Es importante utilizar siempre el mismo registro para realizar la lectura. El registro utilizado debe configurarse en el registro de configuración `SEQA_CTRL` o `SEQB_CTRL`, bit 30 (0 para leer los registros globales o 1 para los de cada canal).



Función de interrupción

```
void ADC_SEQA_IRQHandler ( void )  
{  
    //Tomo el registro GDAT, me quedo con los bits del resultado:  
    resultadoADC = ( ADC->SEQA_GDAT >> 4 ) & 0x0FFF;  
}
```

Table 456. A/D Sequence A Global Data Register (SEQA_GDAT, address 0x4001 C010) bit description

Bit	Symbol	Description
3:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
15:4	RESULT	<p>This field contains the 12-bit A/D conversion result from the most recent conversion performed under conversion sequence associated with this register.</p> <p>The result is the a binary fraction representing the voltage on the currently-selected input channel as it falls within the range of V_{REFP} to V_{REFN}. Zero in the field indicates that the voltage on the input pin was less than, equal to, or close to that on V_{REFN}, while 0xFFF indicates that the voltage on the input was close to, equal to, or greater than that on V_{REFP}.</p> <p>DATAVALID = 1 indicates that this result has not yet been read.</p>
17:16	THCMPRANGE	Indicates whether the result of the last conversion performed was above, below or

Función de interrupción - filtrando el resultado

Una pequeña variación en la tensión de entrada hará que los bits menos significativos de la conversión fluctúen ligeramente. Para evitar este ruido en el resultado se puede hacer un promedio de las últimas muestras, utilizando un buffer circular:

```
void ADC_SEQA_IRQHandler ( void )
{
    static uint8_t posicion = 0;
    uint32_t auxiliar = 0;

    buffer_muestras[posicion] = ( ADC->SEQA_GDAT >> 4 ) & 0x0FFF;
    posicion++;
    posicion %= TAM_BUFFER_ADC;

    for ( int i = 0 ; i < TAM_BUFFER_ADC ; i++)
        auxiliar+=buffer_muestras[i];

    resultadoADC =  auxiliar/TAM_BUFFER_ADC;
```