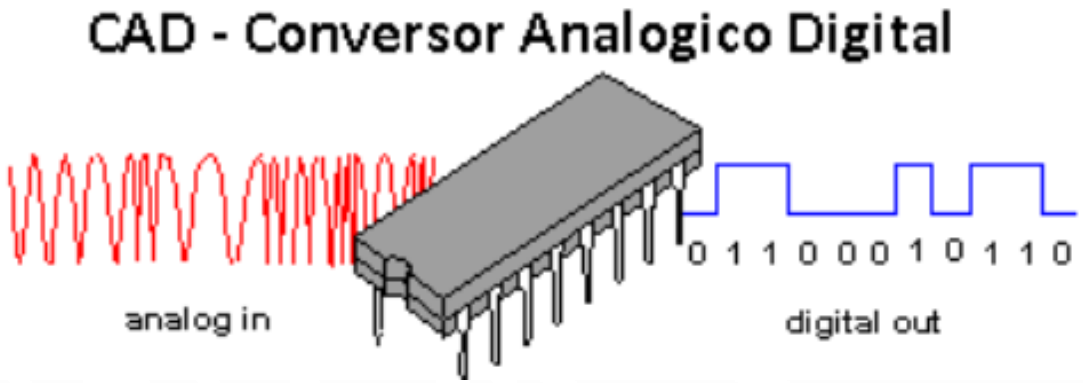
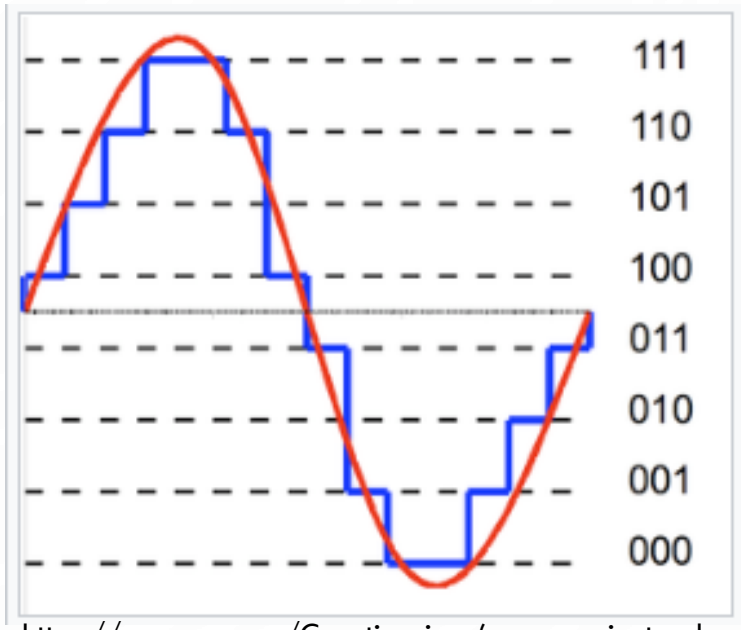
MICROS 32 BITS
STM – MODULO ADC

ROBINSON JIMENEZ MORENO

ADC- STM32F



https://sites.google.com/a/educ.ar/eti/micros/picbasic-pro/3_cad

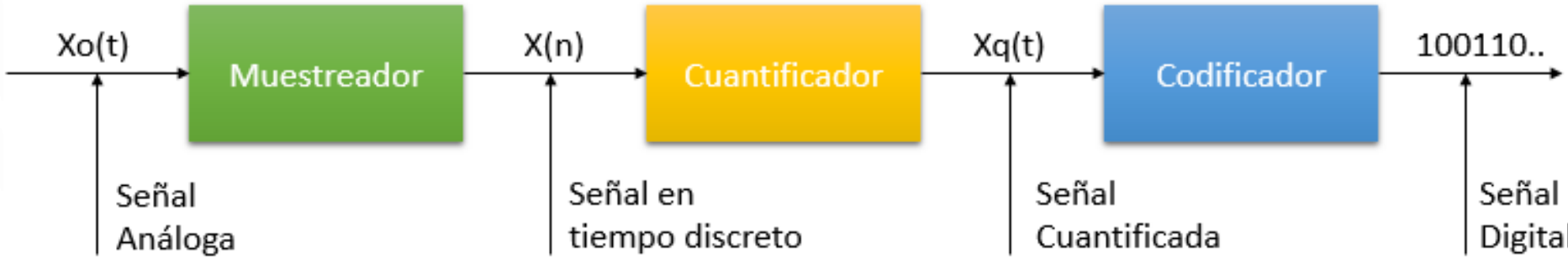


https://copro.com.ar/Cuantizacion_(procesamiento_de_senal).html

ADC- STM32F

Módulo ADC:

Permite la conversión de una señal analógica a digital, mediante el siguiente procedimiento:



La STM32F cuenta con tres canales de conversión y guarda el valor análogo en un código binario de 12 bits. Además, posee hasta 19 entradas multiplexadas que le permiten medir señales de 16 fuentes externas, dos fuentes internas, y una del canal VBAT (voltaje de batería)

ADC- STM32F

Módulo ADC:

La conversión ADC de los canales se puede realizar en diferentes modos: modo único, continuo, escaneo o discontinuo. El resultado de la conversión ADC se almacena en un registro de 16 bits, en donde el código generado es por defecto de 12 bits y puede ser justificado a la derecha o a la izquierda del registro.

ADC- STM32F

Módulo ADC:

Las características principales del módulo ADC son:

- Resolución configurable a 12 bits, 10 bits, 8 bits o 6 bits.
- Generación de una interrupción al final de la conversión.
- Modos de conversión único y continuo.
- Opción de activación externa con polaridad configurable
- Modo discontinuo.
- Modo Doble / Triple (en dispositivos con 2 ADCs o más).
- Retardo configurable entre conversiones en modo entrelazado Dual / Triple
- Suministro de energía al módulo ADC: 2,4 V a 3,6 V a toda velocidad y hacia abajo a 1,8 V a velocidad más lenta
- Rango de entrada ADC: VREF- ≤ ≤ VIN VREF +

ADC- STM32F

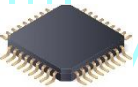
Módulo ADC: Piner principales del módulo ADC

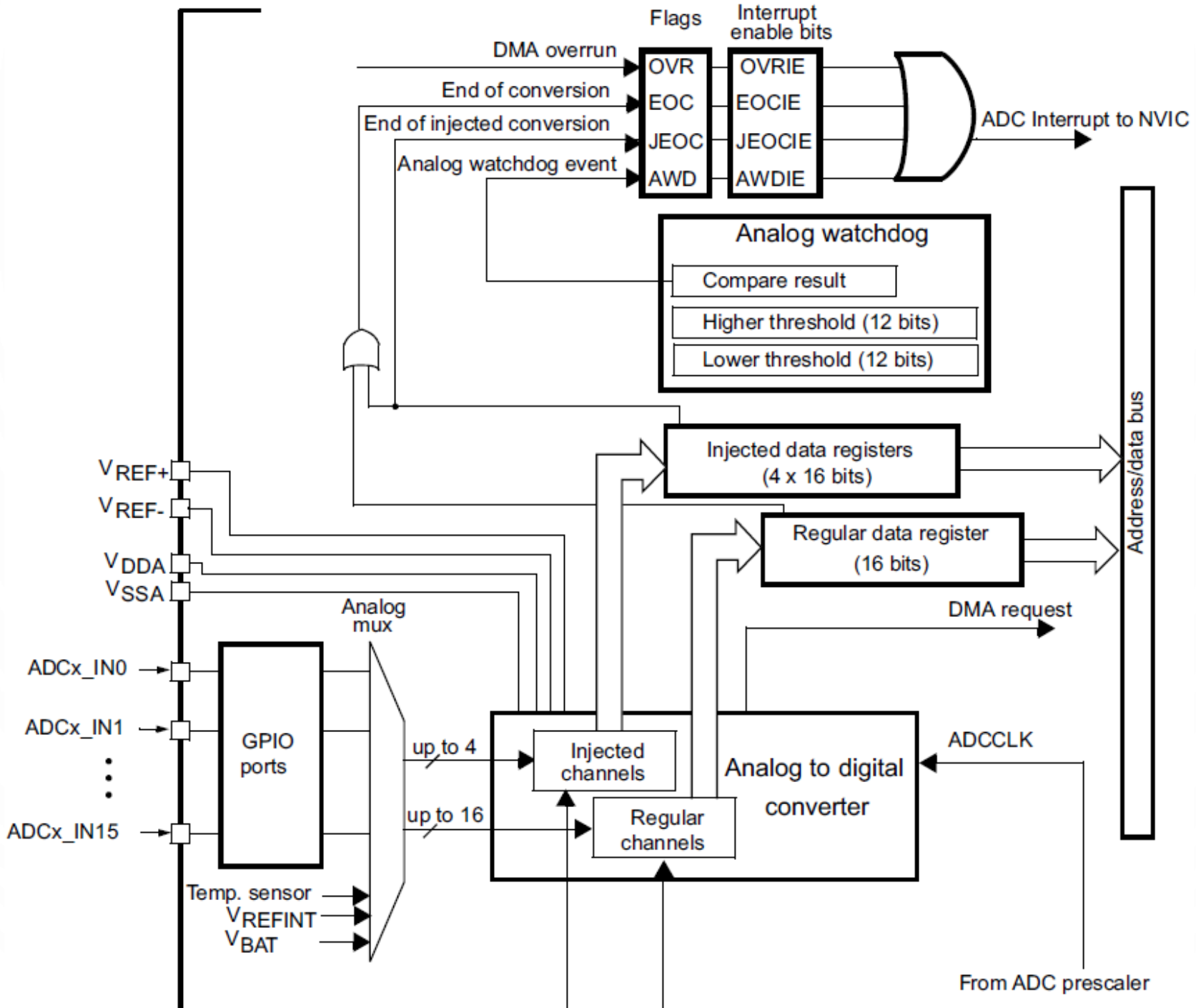
NOMBRE	TIPO DE SEÑAL	DESCRIPCION
V_{REF+}	Entrada, Referencia Análoga Positiva	El mayor voltaje (positivo) para ADC 1.8 v ≤ V_{REF+} ≤ V_{DDA}
V_{DDA}	Entrada, Suministro Análogo	Suministro de voltaje análogo igual a VDD: • 2.4 v ≤ V_{DDA} ≤ V_{DD} (3.6 v) para máxima velocidad • 1.8 v ≤ V_{DDA} ≤ V_{DD} (3.6 v) para velocidad reducida
V_{REF-}	Entrada, Referencia Análoga Negativa	El menor voltaje (negativo) para ADC 1.8 V_{REF-} = V_{SSA}
V_{SSA}	Entrada, Tierra Análoga	Tierra para suministro de voltaje análogo igual a V_{SS}
ADCX_IN[15:0]	Señales de Entrada Análogas	16 Canales Análogos de entrada

ADC introduction

The 12-bit ADC is a successive approximation analog-to-digital converter. It has up to 19 multiplexed channels allowing it to measure signals from 16 external sources, two internal sources, and the V_{BAT} channel. The A/D conversion of the channels can be performed in single, continuous, scan or discontinuous mode. The result of the ADC is stored into a left- or right-aligned 16-bit data register.

The analog watchdog feature allows the application to detect if the input voltage goes beyond the user-defined, higher or lower thresholds.




Flags
Interrupt enable bits
DMA overrun
End of conversion
End of injected conversion
Analog watchdog event
OVR
EOC
JEOC
AWD
OVRIE
EOCIE
JEOCIE
AWDIE
ADC Interrupt to NVIC
Analog watchdog
Compare result
Higher threshold (12 bits)
Lower threshold (12 bits)
Injected data registers
(4 x 16 bits)
Regular data register
(16 bits)
Address/data bus
DMA request
VREF+
VREF-
VDDA
VSSA
Analog mux
ADCx_IN0
ADCx_IN1
ADCx_IN15
GPIO ports
up to 4
up to 16
Injected channels
Regular channels
Analog to digital converter
ADCCLK
Temp. sensor
VREFINT
VBAT
From ADC prescaler

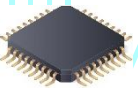
ADC on-off control

The ADC is powered on by setting the ADON bit in the ADC_CR2 register. When the ADON bit is set for the first time, it wakes up the ADC from the Power-down mode.

The conversion starts when either the SWSTART or the JSWSTART bit is set.

The user can stop conversion and put the ADC in power down mode by clearing the ADON bit. In this mode the ADC consumes almost no power (only a few µA).

Name	Signal type	Remarks
V_{REF+}	Input, analog reference positive	The higher/positive reference voltage for the ADC, 1.8 V ≤ V_{REF+} ≤ V_{DDA}
V_{DDA}	Input, analog supply	Analog power supply equal to V_{DD} and 2.4 V ≤V_{DDA} ≤V_{DD} (3.6 V) for full speed 1.8 V ≤V_{DDA} ≤V_{DD} (3.6 V) for reduced speed
V_{REF-}	Input, analog reference negative	The lower/negative reference voltage for the ADC, V_{REF-} = V_{SSA}
V_{SSA}	Input, analog supply ground	Ground for analog power supply equal to V_{SS}
ADCx_IN[15:0]	Analog input signals	16 analog input channels



ADC clock

The ADC features two clock schemes:

- Clock for the analog circuitry: ADCCLK, common to all ADCs

  This clock is generated from the APB2 clock divided by a programmable prescaler that allows the ADC to work at f_{PCLK2}/2, /4, /6 or /8. Refer to the datasheets for the maximum value of ADCCLK.

- Clock for the digital interface (used for registers read/write access)

  This clock is equal to the APB2 clock. The digital interface clock can be enabled/disabled individually for each ADC through the RCC APB2 peripheral clock enable register (RCC_APB2ENR).

5.3.14 RCC APB2 peripheral clock enable register (RCC_APB2ENR)

Address offset: 0x44

Reset value: 0x0000 0000

Access: no wait state, word, half-word and byte access.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	LTDC EN	Res.	Res.	SAI2EN	SAI1EN	SPI6EN	SPI5EN	Res.	TIM11 EN	TIM10 EN	TIM9 EN
					rw			rw	rw	rw	rw		rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SYSCFG EN	SPI4 EN	SPI1 EN	SDMMC1 EN	ADC3 EN	ADC2 EN	ADC1 EN	Res.	Res.	USART6 EN	USART1 EN	Res.	Res.	TIM8 EN	TIM1 EN
	rw	rw	rw	rw	rw	rw	rw			rw	rw			rw	rw

```
RCC->APB2ENR |=(1UL<<10);
```

Channel selection

There are 16 multiplexed channels. It is possible to organize the conversions in two groups: regular and injected. A group consists of a sequence of conversions that can be done on any channel and in any order. For instance, it is possible to implement the conversion sequence in the following order: ADC_IN3, ADC_IN8, ADC_IN2, ADC_IN2, ADC_IN0, ADC_IN2, ADC_IN2, ADC_IN15.

- A **regular group** is composed of up to 16 conversions. The regular channels and their order in the conversion sequence must be selected in the ADC_SQRx registers. The total number of conversions in the regular group must be written in the L[3:0] bits in the ADC_SQR1 register.
- An **injected group** is composed of up to 4 conversions. The injected channels and their order in the conversion sequence must be selected in the ADC_JSQR register. The total number of conversions in the injected group must be written in the L[1:0] bits in the ADC_JSQR register.

```
ADC3->SQR3 =4; // CANAL 4 ADC_3 PIN-PF6
```

15.13.11 ADC regular sequence register 3 (ADC_SQR3)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	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

Bits 24:20 **SQ5[4:0]:** 5th conversion in regular sequence

Bits 19:15 **SQ4[4:0]:** 4th conversion in regular sequence

Bits 14:10 **SQ3[4:0]:** 3rd conversion in regular sequence

Bits 9:5 **SQ2[4:0]:** 2nd conversion in regular sequence

Bits 4:0 **SQ1[4:0]:** 1st conversion in regular sequence

15.13.10 ADC regular sequence register 2 (ADC_SQR2)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SQ12[4:0]					SQ11[4:0]					SQ10[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
SQ10_0	SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15.13.9 ADC regular sequence register 1 (ADC_SQR1)

Address offset: 0x2C

Reset value: 0x0000 0000

L[3:0]: Regular channel sequence length
0000: 1 conversion
0001: 2 conversions
...
1111: 16 conversions

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	L[3:0]				SQ16[4:1]			
								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ16_0	SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

SELECCIÓN DEL CANAL

A0	ADC	PA3	ADC1_IN3
A1	ADC	PC0	ADC1_IN10
A2	ADC	PC3	ADC1_IN13
A3	ADC	PC1	ADC1_IN11
A4	ADC	PC4 or PB9(1)	ADC1_IN14 (PC4) or I2C1_SDA (PB9)
A5	ADC	PC5 or PB8(1)	ADC1_IN15 (PC5) or I2C1_SCL (PB8)

PF3	I/O	FT	-	FMC_A3, EVENTOUT	ADC3_IN9
PF4	I/O	FT	-	FMC_A4, EVENTOUT	ADC3_IN14
PF5	I/O	FT	-	FMC_A5, EVENTOUT	ADC3_IN15
VSS	S	-	-	-	-
VDD	S	-	-	-	-
PF6	I/O	FT	-	TIM10_CH1, SPI5_NSS, SAI1_SD_B, UART7_Rx, QUADSPI_BK1_IO3, EVENTOUT	ADC3_IN4
PF7	I/O	FT	-	TIM11_CH1, SPI5_SCK, SAI1_MCLK_B, UART7_Tx, QUADSPI_BK1_IO2, EVENTOUT	ADC3_IN5
PA3	I/O	FT	(4)	TIM2_CH4, TIM5_CH4, TIM9_CH2, USART2_RX, OTG_HS_ULPI_D0, ETH_MII_COL, LCD_B5, EVENTOUT	ADC123_IN3
VSS	S	-	-	-	-
BYPASS_REG	I	FT	-	-	-
VDD	S	-	-	-	-
PA4	I/O	TTa	(4)	SPI1_NSS/I2S1_WS, SPI3_NSS/I2S3_WS, USART2_CK, OTG_HS_SOF, DCMI_HSYNC, LCD_VSYNC, EVENTOUT	ADC12_IN4, DAC_OUT1
PA5	I/O	TTa	(4)	TIM2_CH1/TIM2_ETR, TIM8_CH1N, SPI1_SCK/I2S1_CK, OTG_HS_ULPI_CK, LCD_R4, EVENTOUT	ADC12_IN5, DAC_OUT2
PA6	I/O	FT	(4)	TIM1_BKIN, TIM3_CH1, TIM8_BKIN, SPI1_MISO, TIM13_CH1, DCMI_PIXCLK, LCD_G2, EVENTOUT	ADC12_IN6
PF8	I/O	FT	-	SPI5_MISO, SAI1_SCK_B, UART7_RTS, TIM13_CH1, QUADSPI_BK1_IO0, EVENTOUT	ADC3_IN6
PF9	I/O	FT	-	SPI5_MOSI, SAI1_FS_B, UART7_CTS, TIM14_CH1, QUADSPI_BK1_IO1, EVENTOUT	ADC3_IN7
PF10	I/O	FT	-	DCMI_D11, LCD_DE, EVENTOUT	ADC3_IN8
PH0-OSC_IN(PH0)	I/O	FT	-	EVENTOUT	OSC_IN(4)
PH1-OSC_OUT(PH1)	I/O	FT	-	EVENTOUT	OSC_OUT(4)
NRST	I/O	RST	-	-	-
PC0	I/O	FT	(4)	SAI2_FS_B, OTG_HS_ULPI_STP, FMC_SDNWE, LCD_R5, EVENTOUT	ADC123_IN10
PC1	I/O	FT	(4)	TRACED0, SPI2_MOSI/I2S2_SD, SAI1_SD_A, ETH_MDC, EVENTOUT	ADC123_IN11, RTC_TAMP3, WKUP3
PC2	I/O	FT	(4)	SPI2_MISO, OTG_HS_ULPI_DIR, ETH_MII_TXD2, FMC_SDNE0, EVENTOUT	ADC123_IN12
PC3	I/O	FT	(4)	SPI2_MOSI/I2S2_SD, OTG_HS_ULPI_NXT, ETH_MII_TX_CLK, FMC_SDCKE0, EVENTOUT	ADC123_IN13
VDD	S	-	-	-	-
VSS	S	-	-	-	-
VSSA	S	-	-	-	-
VREF-	S	-	-	-	-
VREF+	S	-	-	-	-
VDDA	S	-	-	-	-
PA0-WKUP(PA0)	I/O	FT	(5)	TIM2_CH1/TIM2_ETR, TIM5_CH1, TIM8_ETR, USART2_CTS, UART4_TX, SAI2_SD_B, ETH_MII_CRS, EVENTOUT	ADC123_IN0, WKUP1(4)
PA1	I/O	FT	(4)	TIM2_CH2, TIM5_CH2, USART2_RTS, UART4_RX, QUADSPI_BK1_IO3, SAI2_MCK_B, ETH_MII_RX_CLK/ETH_RMII_REF_CLK, LCD_R2, EVENTOUT	ADC123_IN1
PA2	I/O	FT	(4)	TIM2_CH3, TIM5_CH3, TIM9_CH1, USART2_TX, SAI2_SCK_B, ETH_MDIO, LCD_R1, EVENTOUT	ADC123_IN2, WKUP2

```
ADC3->SQR3 =4; // CANAL 4 ADC_3 PIN-PF6
```

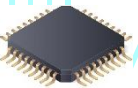


Figure 71. ADC1 connectivity

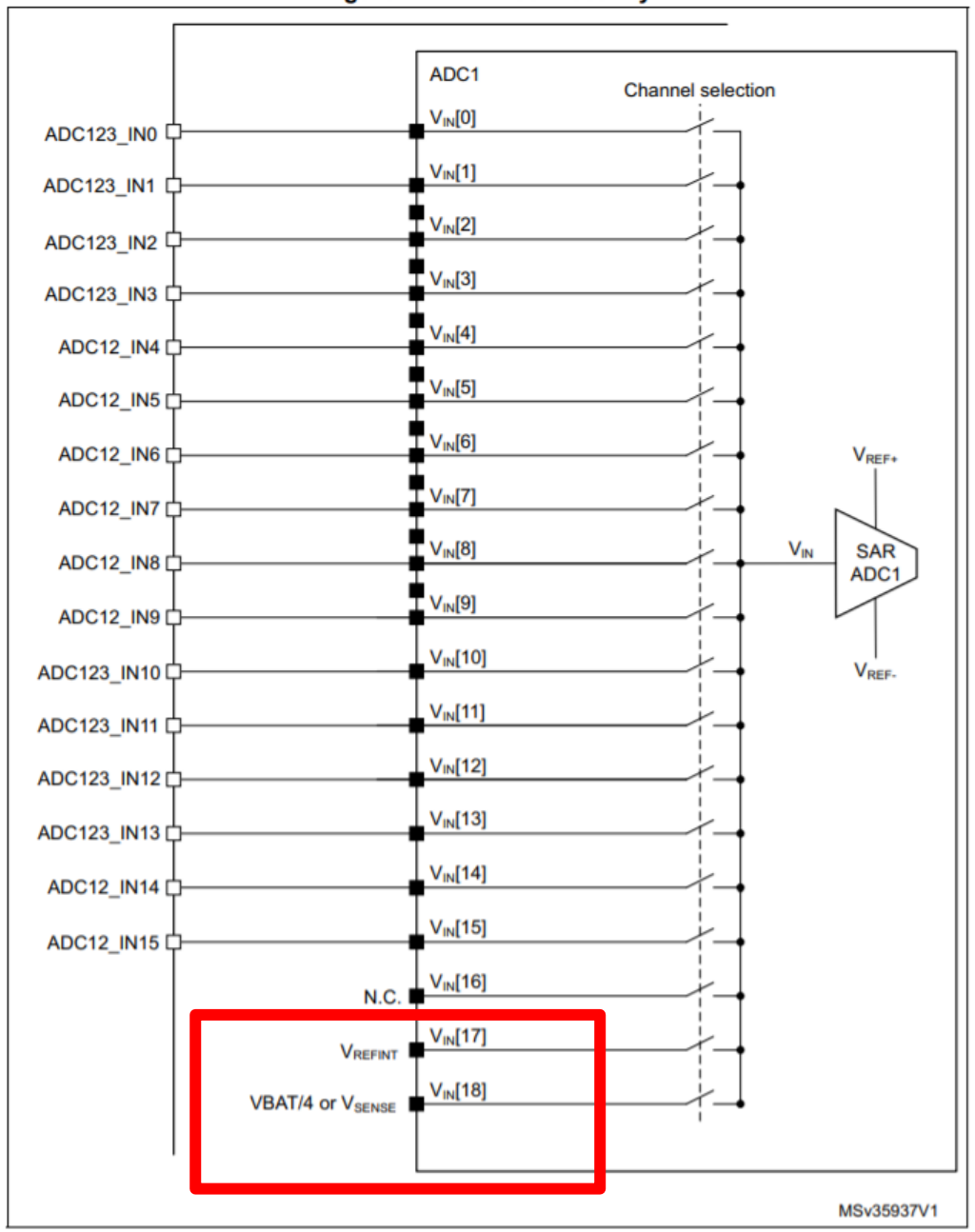
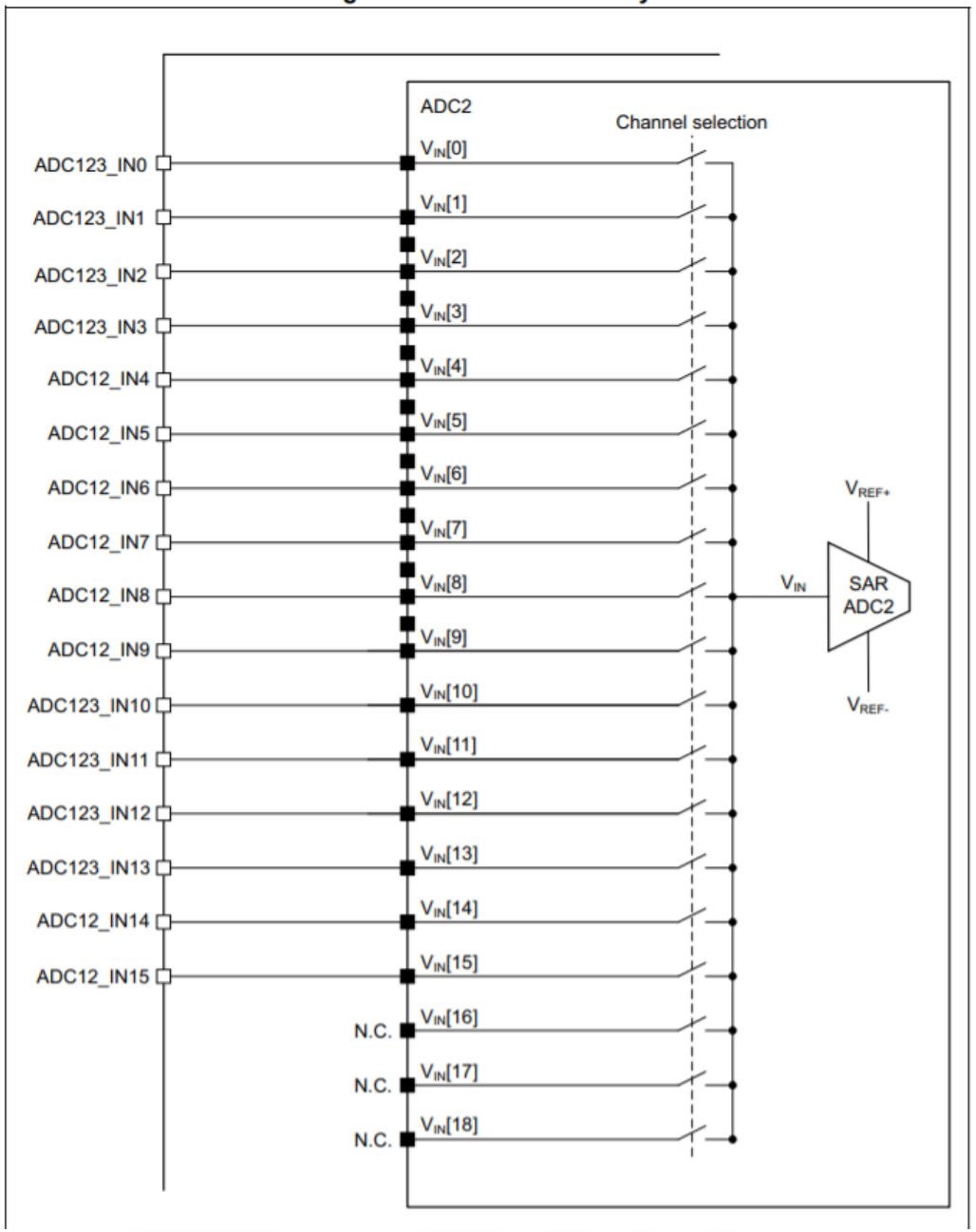


Figure 72. ADC2 connectivity



INGENIERÍA MECATRÓNICA UMNG

UNIVERSIDAD MILITAR NUEVA GRANADA

Single conversion mode

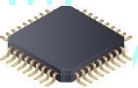
In Single conversion mode the ADC does one conversion. This mode is started with the CONT bit at 0 by either:

- setting the SWSTART bit in the ADC_CR2 register (for a regular channel only)
- setting the JSWSTART bit (for an injected channel)
- external trigger (for a regular or injected channel)

Once the conversion of the selected channel is complete:

- If a regular channel was converted:
  - The converted data are stored into the 16-bit ADC_DR register
  - The EOC (end of conversion) flag is set
  - An interrupt is generated if the EOCIE bit is set
- If an injected channel was converted:
  - The converted data are stored into the 16-bit ADC_JDR1 register
  - The JEOC (end of conversion injected) flag is set
  - An interrupt is generated if the JEOCIE bit is set

Then the ADC stops.



15.13.2 ADC control register 1 (ADC_CR1)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	OVRIE	RES		AWDEN	JAWDEN	Res.	Res.	Res.	Res.	Res.	Res.
					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

```
ADC2->CR1 |= (1UL << 5);
```

Bit 5 **EOCIE:** Interrupt enable for EOC

This bit is set and cleared by software to enable/disable the end of conversion interrupt.

0: EOC interrupt disabled

1: EOC interrupt enabled. An interrupt is generated when the EOC bit is set.

15.13.2 ADC control register 1 (ADC_CR1)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	OVRIE	RES		AWDEN	JAWDEN	Res.	Res.	Res.	Res.	Res.	Res.
					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

Bits 25:24 **RES[1:0]**: Resolution

These bits are written by software to select the resolution of the conversion.

00: 12-bit (minimum 15 ADCCLK cycles)
01: 10-bit (minimum 13 ADCCLK cycles)
10: 8-bit (minimum 11 ADCCLK cycles)
11: 6-bit (minimum 9 ADCCLK cycles)

Tiempo de conversión

Hexa	Decimal	Voltaje
0	0	0V
FFF	4095	3,3V

15.13.3 ADC control register 2 (ADC_CR2)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	SWSTART	EXTEN		EXTSEL[3:0]				Res.	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
Res.	Res.	Res.	Res.	ALIGN	EOCS	DDS	DMA	Res.	Res.	Res.	Res.	Res.	Res.	CONT	ADON
				rw	rw	rw	rw							rw	rw

Bit 10 **EOCS:** End of conversion selection

This bit is set and cleared by software.

0: The EOC bit is set at the end of each sequence of regular conversions. Overrun detection is enabled only if DMA=1.

1: The EOC bit is set at the end of each regular conversion. Overrun detection is enabled.

Bit 0 **ADON:** A/D Converter ON / OFF

This bit is set and cleared by software.

Note: 0: Disable ADC conversion and go to power down mode

1: Enable ADC

15.13.14 ADC regular data register (ADC_DR)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **DATA[15:0]:** Regular data

These bits are read-only. They contain the conversion result from the regular channels. The data are left- or right-aligned as shown in *Figure 77* and *Figure 78*.

Data alignment

The ALIGN bit in the ADC_CR2 register selects the alignment of the data stored after conversion. Data can be right- or left-aligned as shown in *Figure 77* and *Figure 78*.

The converted data value from the injected group of channels is decreased by the user-defined offset written in the ADC_JOFRx registers so the result can be a negative value. The SEXT bit represents the extended sign value.

For channels in a regular group, no offset is subtracted so only twelve bits are significant.

Figure 77. Right alignment of 12-bit data

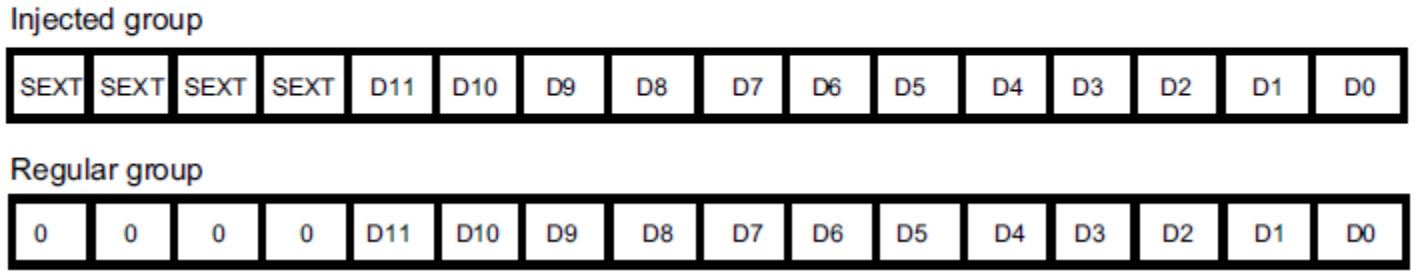


Figure 78. Left alignment of 12-bit data

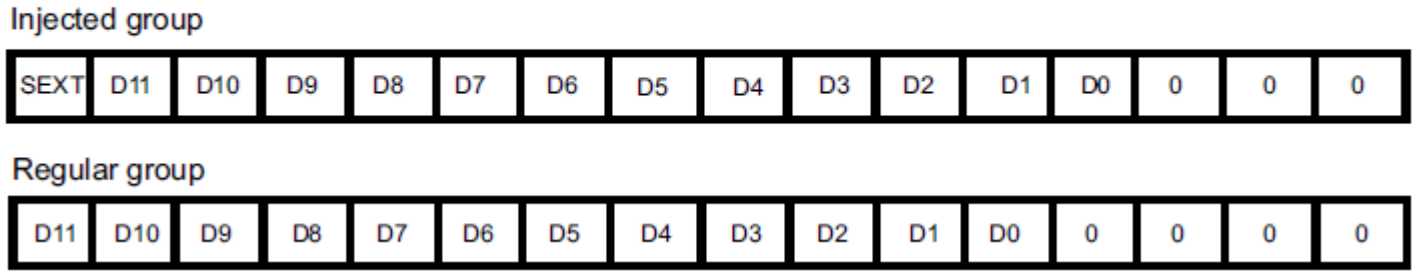
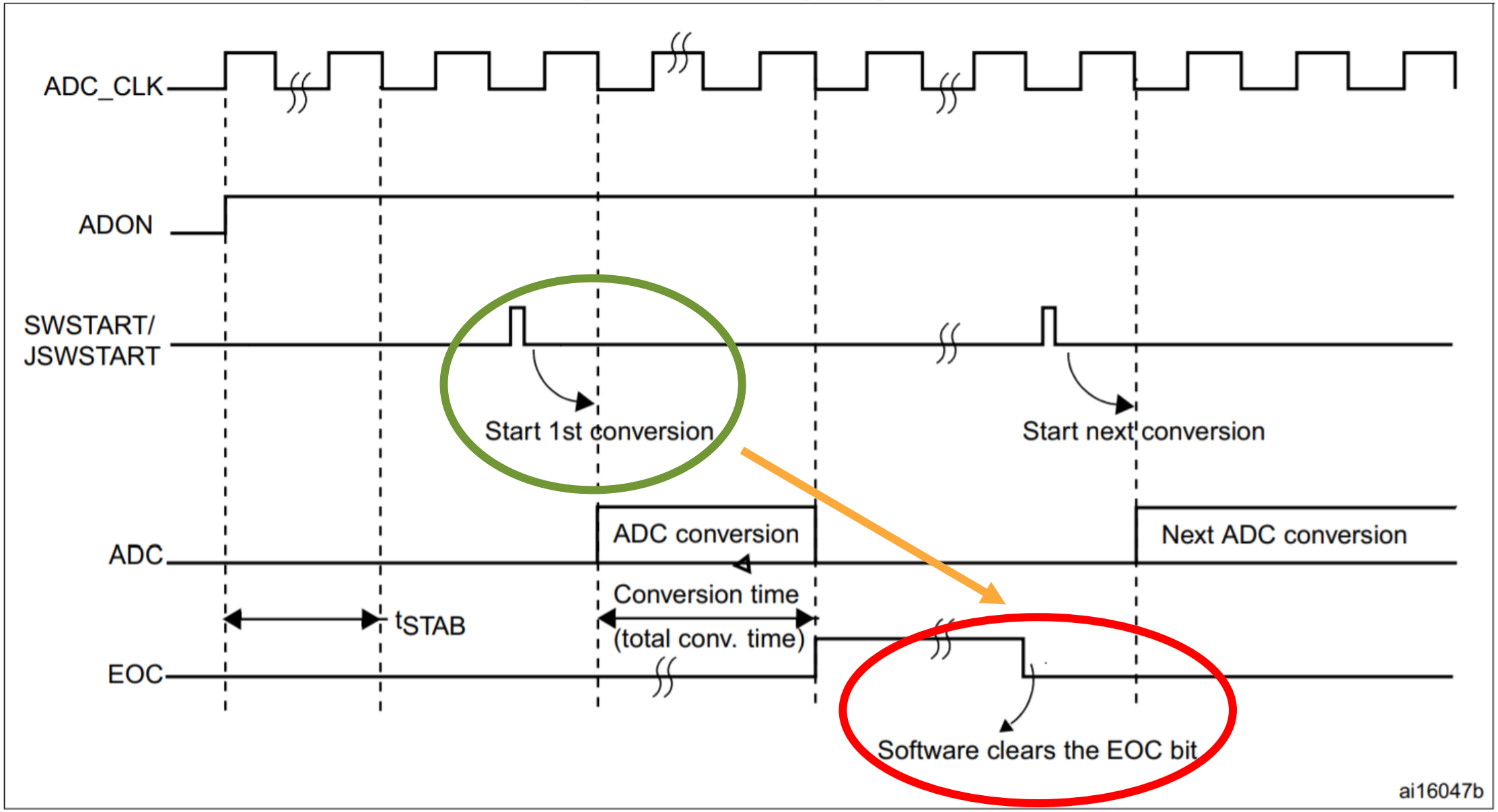
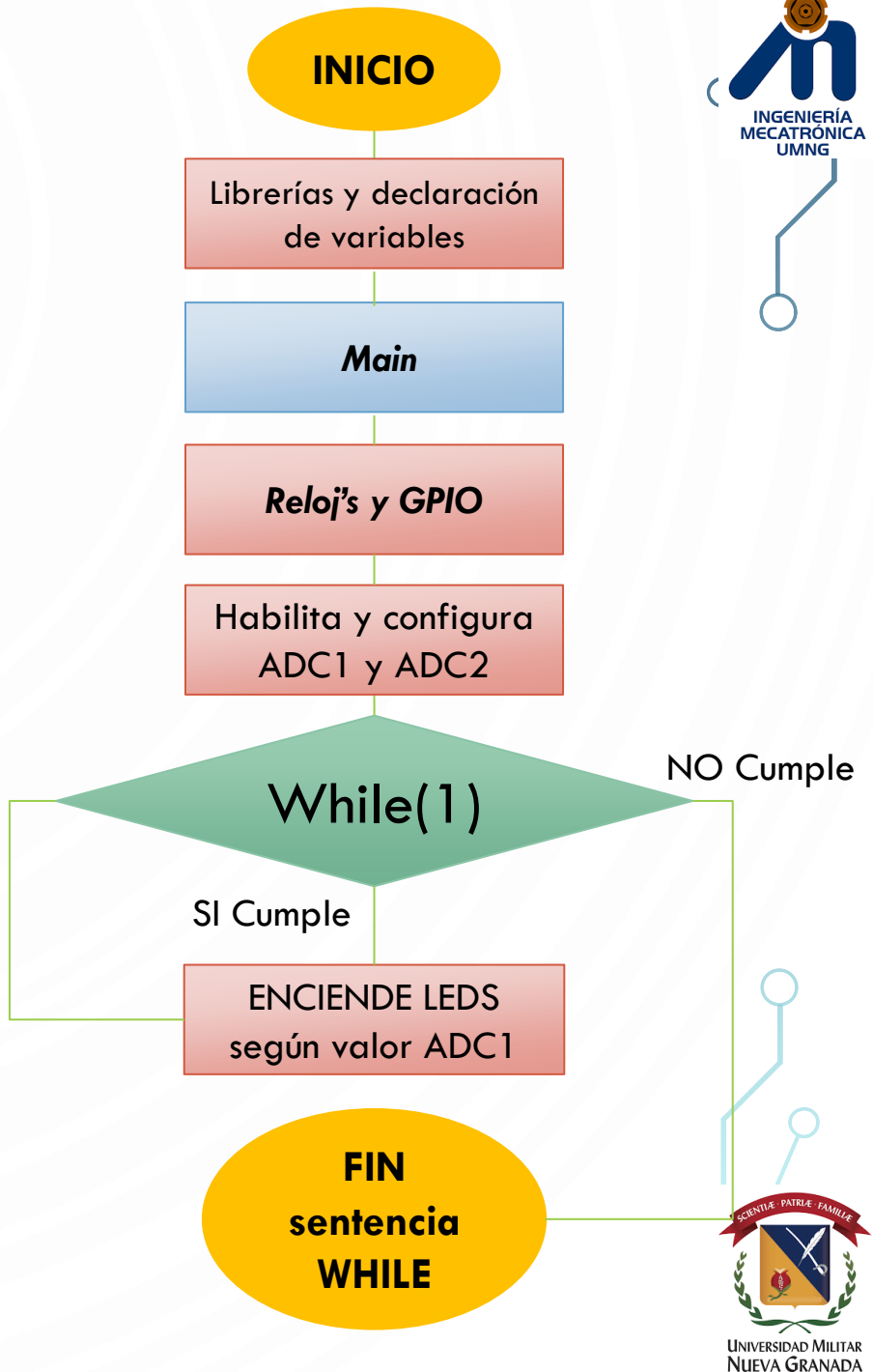


Figure 74. Timing diagram

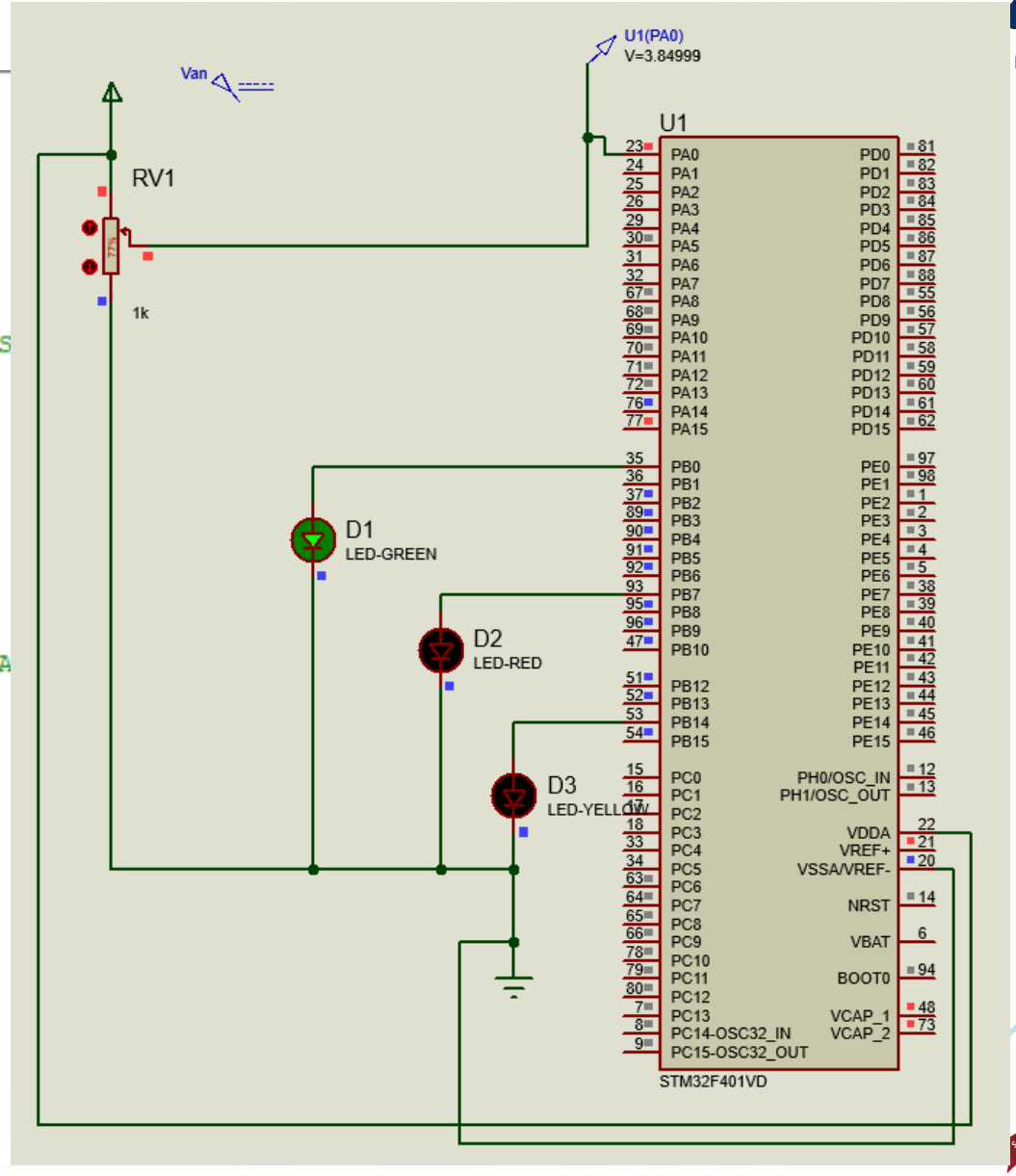


```
//*******************************************************************************
#include "STM32F7xx.h"
int codigo;
extern "C"{
    void ADC_IRQHandler(void){
    codigo = ADC2->DR;
  }
}

int main(void){
  RCC->AHB1ENR = 0xF;     //PRENDER EL CLOCK
  RCC->APB2ENR |=(1UL<<9);
  ADC2->CR2 = 1;
  ADC2->CR1 |= (1UL << 24);  //Activa resolucion a 10 bits
  ADC2->CR1 |= (1UL << 5);  //Activa la interrupcion POR FINAL DE CONVERSION
  ADC2->CR2 |= (1UL << 10); //Set to 1 the bit EOCS
  NVIC_EnableIRQ(ADC_IRQn);   //HABILITA INTERRUPCION DE LOS ADC
  ADC2->SQR3 =10; // CANAL 10
  GPIOC->MODER =3; //ADC_2 PIN-PC0
  GPIOB->MODER |=(1UL<<0); //led de alarma
  while(true){
  ADC2->CR2 |= (1UL << 30); //INICIO DE LA CONVERSION
  while ((ADC2->SR &= 0x2)==1);
    if(codigo>512){GPIOB->ODR=1;}
    else GPIOB->ODR=0;
    }
}
```

```c
#include <stdio.h>
#include "stm32f7xx.h"

int main(void){

  RCC -> AHB1ENR = 0X6; //PUERTO B Y C
  RCC -> APB2ENR = 0X100; //HABILITAR EL ADC 1
//  RCC -> APB2ENR = 0X200; //HABILITAR EL ADC 2

  GPIOB -> MODER = 0X10004001; //COLOCAR EN SALIDA PARA ENCENDER LOS LEDS
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 0X10004001; //VELOCIDAD MEDIA
  GPIOB -> PUPDR = 0X10004001; //PULL UP

  GPIOC -> MODER = 0X3; //ANALOGO PARA EL PIN 0

  ADC1 -> CR1 = 0X0; //RESOLUCION DE 12 BITS (POR DEFECTO)
  ADC1 -> CR2 = 0X1; //ENCENDER EL ADC
  ADC1 -> SMPR1 = 0X7FFFFFF; //TIEMPO DE MUESTREO EN CICLOS
  ADC1 -> SMPR2 = 0X37777777; //TIEMPO DE MUESTREO EN CICLOS
  ADC1 -> SQR3 = 10; //CANAL 10 DEL ADC
  while(1){
    ADC1 -> CR2 |= 0X40000000; //INICIAR LA CONVERSION EN CANALES REGULARES
    if(ADC1 -> DR == 2048){
      GPIOB -> ODR = 0X4000;
    }
    if(ADC1 -> DR == 4095){
      GPIOB -> ODR = 0X1;
    }
    if(ADC1 -> DR == 0){
      GPIOB -> ODR = 0X80;
    }
```



```c
#include "STM32F4xx.h"
int temp;


int main(void){

  RCC -> AHB1ENR = 0XF; //PUERTOS
  RCC -> APB2ENR = 0X100; //HABILITAR EL ADC 1
  GPIOB -> MODER = 0X55555555; //COLOCAR EN SALIDA PARA ENCENDER LOS LEDS
  GPIOA -> MODER |= 0X3; //ANALOGO PARA EL PIN 0

  ADC1 -> CR1 = 0X0; //RESOLUCION DE 12 BITS (POR DEFECTO)
  ADC1 -> CR2 = 0X1; //ENCENDER EL ADC
  ADC1 -> SMPR1 = 0X7FFFFFF; //TIEMPO DE MUESTREO EN CICLOS
  ADC1 -> SMPR2 = 0X37777777; //TIEMPO DE MUESTREO EN CICLOS
  ADC1 -> SQR3 = 0; //CANAL 0 DEL ADC PIN PA0

  while(1){
    ADC1 -> CR2 |= (1UL<<30); //0X40000000; //INICIAR LA CONVERSION EN CA
    while((ADC1->SR & 0X2)==1);
     if(ADC1 -> DR >2048){
     GPIOB -> ODR = 0X4000;
    }
      if(ADC1 -> DR >= 4080){
      GPIOB -> ODR = 0X80;
    }
      if(ADC1 -> DR >= 20){
      GPIOB -> ODR = 0X1;
    }
  }
}
```



INGENIERÍA MECATRÓNICA UMNG

```c
#include <stdio.h>
#include "stm32f7xx.h"
int temp=0;
int main(void){

  RCC -> AHB1ENR |= 0X6; //PUERTO B Y C
  RCC -> APB2ENR |= 0X400; //HABILITAR EL ADC 3

  GPIOB -> MODER |= 0X10004001; //COLOCAR EN SALIDA PARA ENCENDER LOS LEDS
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 0X10004001; //VELOCIDAD MEDIA
  GPIOB -> PUPDR |= 0X10004001; //PULL UP

  GPIOC -> MODER |= 15; //ANALOGO PARA EL PIN 0 y 1

  ADC3 -> CR1 |= 0X0; //RESOLUCION DE 12 BITS (POR DEFECTO)
  ADC3 -> CR2 |= 0X201; //ENCENDER EL ADC

  while(1){
    ADC3 -> SQR3 =10; //CANAL 10 DEL ADC
    ADC3 -> CR2 |= 0X40000000; //INICIAR LA CONVERSION EN CANALES
    while((ADC3->SR &=0X2)==1);
    temp=(ADC3 -> DR)*8058/100000;
    if(temp> 50){
      GPIOB -> ODR |= 0X1;
    }
    else {GPIOB -> ODR &= 0;}
      //
  }
}
```

SENSOR DE TEMPERATURA LM35

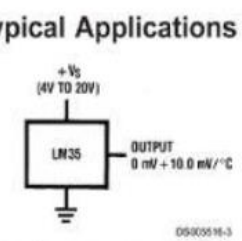
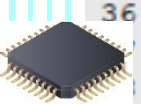


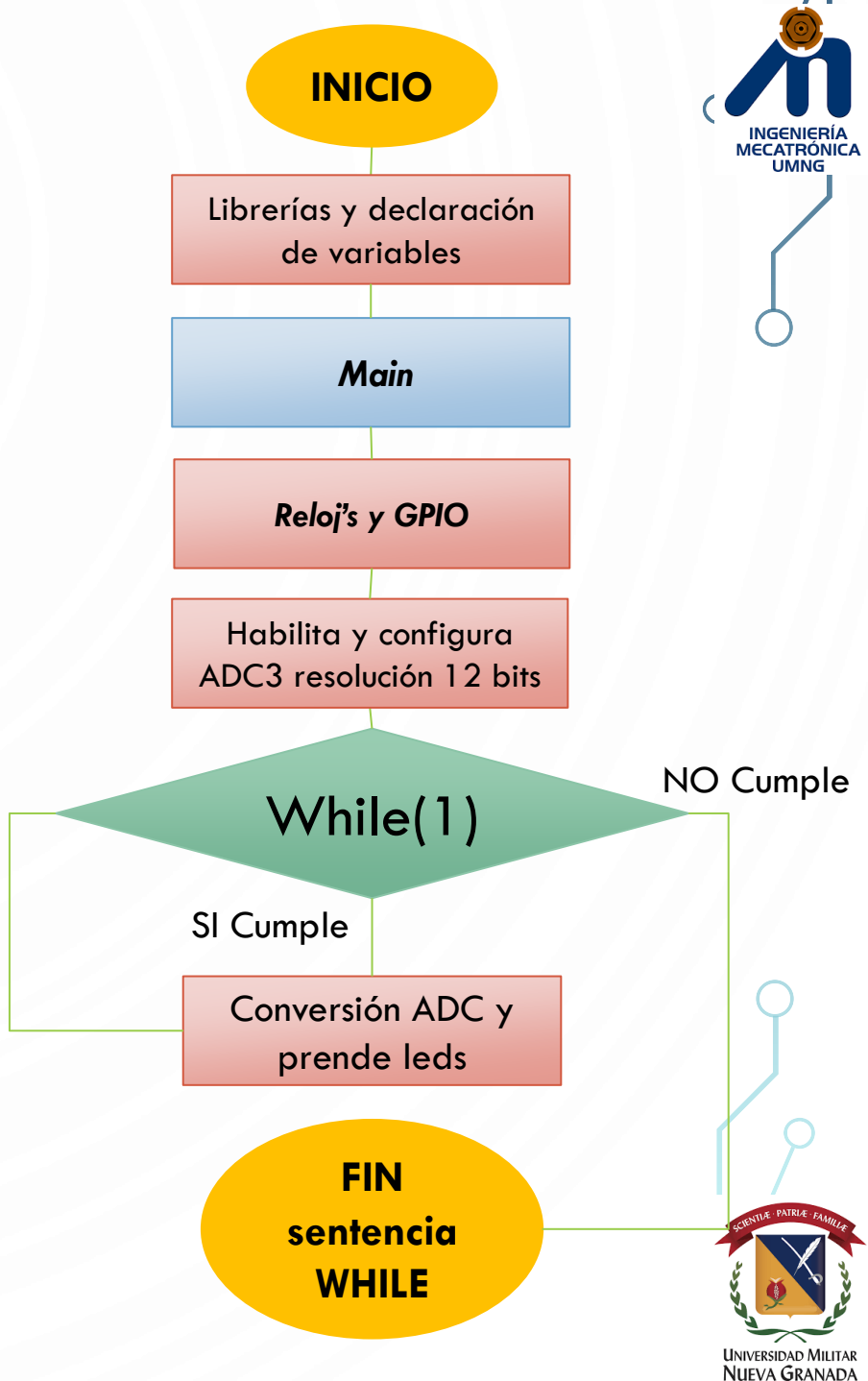
FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)

Supply Voltage: 4 to 30 V
Temp. Range: -55 to +150 °C
Accuracy: ±2 °C over range
Output: +10mV/°C

UNIVERSIDAD MILITAR NUEVA GRANADA

```c
#include <stdio.h>
#include "stm32f7xx.h"
int temp=0;
int main(void){

  RCC -> AHB1ENR |= 0X6; //PUERTO B Y C
  RCC -> APB2ENR |= 0X400; //HABILITAR EL ADC 3

  GPIOB -> MODER |= 0X10004001; //COLOCAR EN SALIDA PARA ENCENDER LOS LEDS
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 0X10004001; //VELOCIDAD MEDIA
  GPIOB -> PUPDR |= 0X10004001; //PULL UP

  GPIOC -> MODER |= 15; //ANALOGO PARA EL PIN 0 y 1

  ADC3 -> CR1 |= 0X0; //RESOLUCION DE 12 BITS (POR DEFECTO)
  ADC3 -> CR2 |= 0X201; //ENCENDER EL ADC

  while(1){
    ADC3 -> SQR3 =10; //CANAL 10 DEL ADC
    ADC3 -> CR2 |= 0X40000000; //INICIAR LA CONVERSION EN CANALES REGULARES
    while((ADC3->SR &=0X2)==1);
    temp=(ADC3 -> DR)*8058/100000;
    if(temp> 50){
      GPIOB -> ODR |= 0X01;
    }
    else {GPIOB -> ODR &= 0xFFFFFFF0;}
      //
    ADC3 -> SQR3 =11; //CANAL 11 DEL ADC
    ADC3 -> CR2 |= 0X40000000; //INICIAR LA CONVERSION EN CANALES REGULARES
    while((ADC3->SR &=0X2)==1);
    temp=(ADC3 -> DR)*8058/100000;
    if(temp > 40){
      GPIOB -> ODR |= 0X80;
    }
    else {GPIOB -> ODR &= 0XFFFFFF01;}
  }
}
```





INGENIERÍA MECATRÓNICA UMNG

UNIVERSIDAD MILITAR NUEVA GRANADA

TAREA

Emplear sensores lm35 para visualizar en una LCD grafica de 128*64 pixeles la temperatura de 2 sensores LM 35 de forma numérica y con barras de desplazamiento para llenado a nivel de la temperatura.