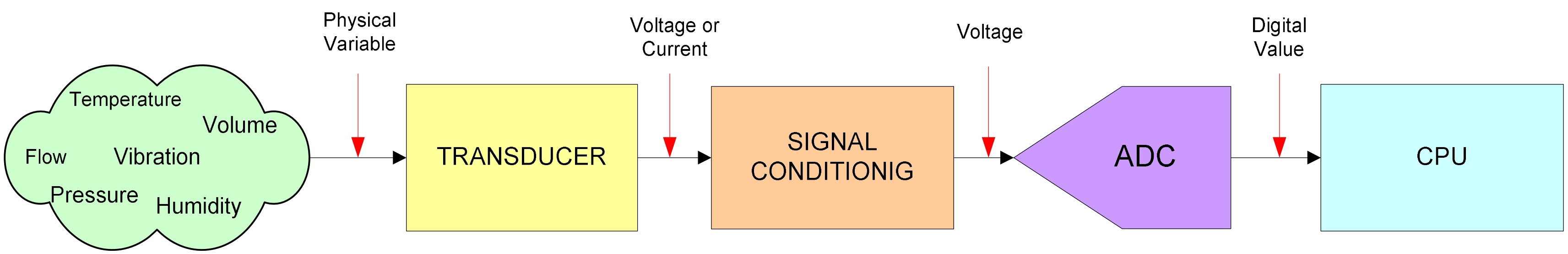
Subject10
THE ANALOG TO DIGITAL CONVERTER

The A/D converter

- Many application require to interact with electrical and other physical variables that are continuous both in amplitude and in time. (analog)
- The A/D converter enables the microcontroller with a hardware that allow to discretize in amplitude and time continuous signal so they can be processed by the CPU

Data acquisition system



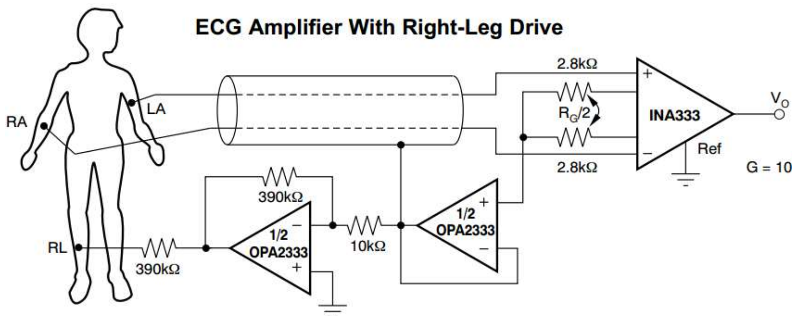
Transducer

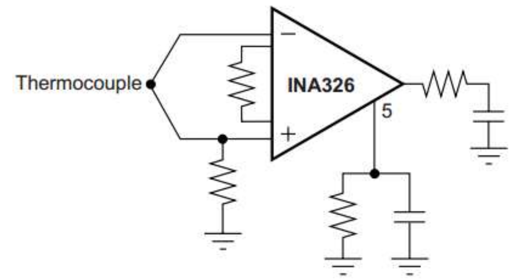
- Converts a physical variable in to an electrical signal .
- Can also be know as “sensor”, some examples are:
  - Strain gauges
  - Thermistor
  - Hall effect
  - Piezoelectric
  - Electromagnetic
  - Potenciométros etc.

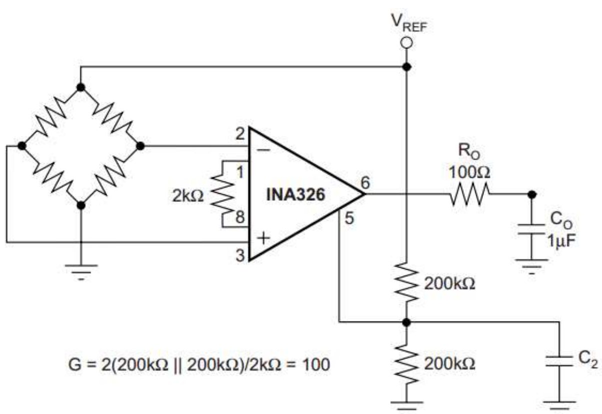
Signal Conditioning

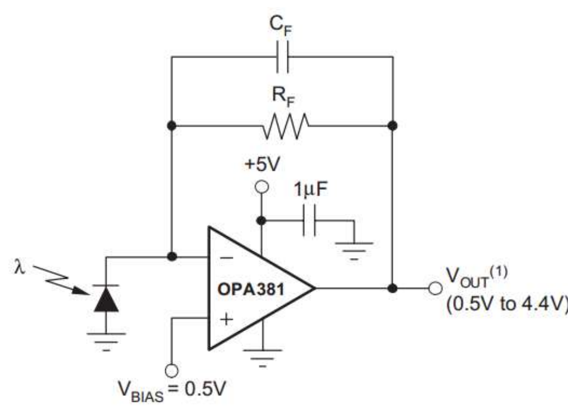
- It convert the signal from the sensor to one that can be fully exploited by the A / D converter (input range)
- It also cleans the signal from noise and limits its bandwidth in order to comply with the Nyquist sampling theorem when this is required.

Examples of signal conditioning circuits









The A/D converter

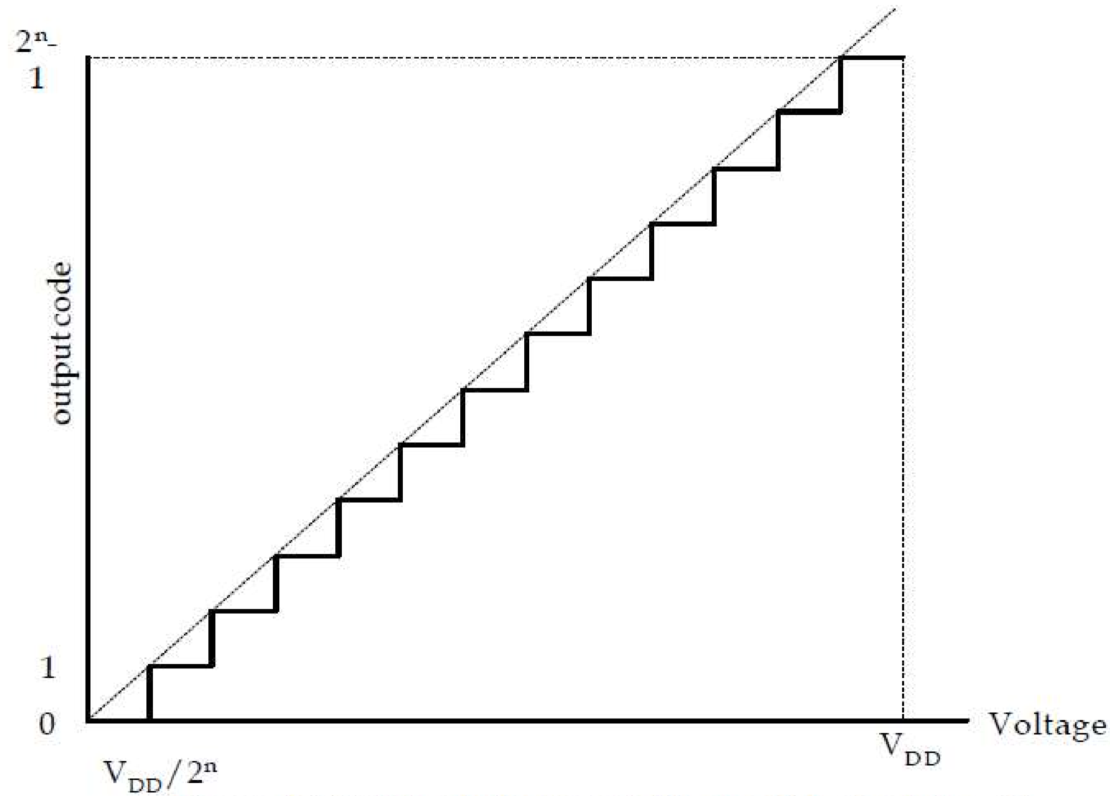
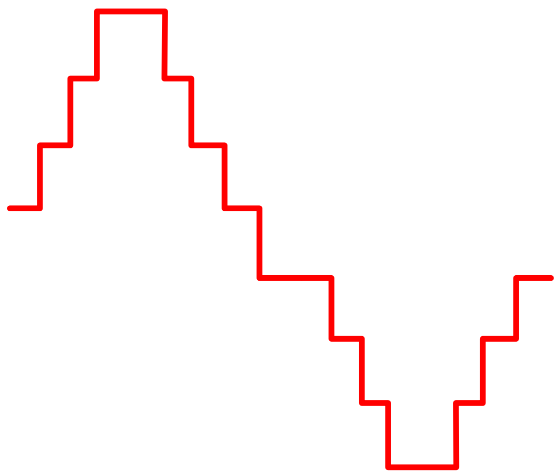
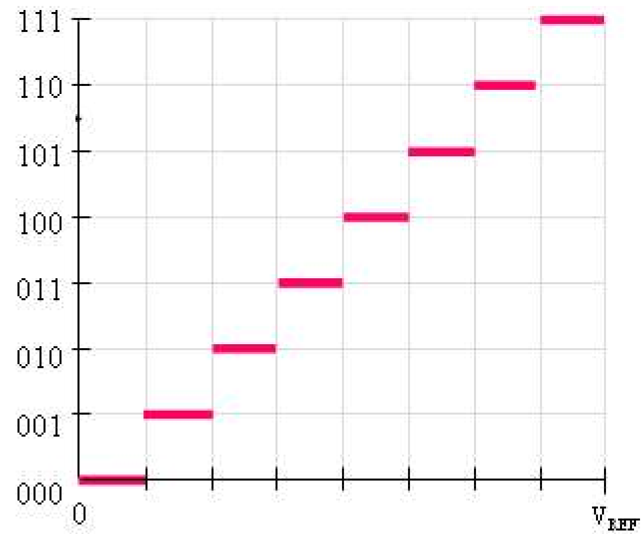


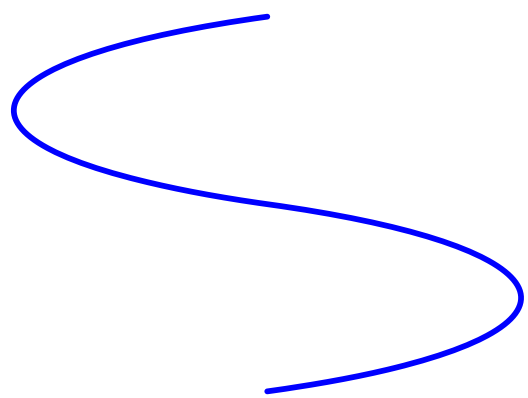
Figure 12.3 Output characteristic of an ideal n-bit A/D

- The area between dotted line and staircase is called the **quantization error**.
- The **resolution** of this A/D converter is $V_{DD}/2^n$.
- Average conversion error is $V_{DD}/2^{n+1}$.
- A real A/D converter has nonlinearity.

The A/D converter



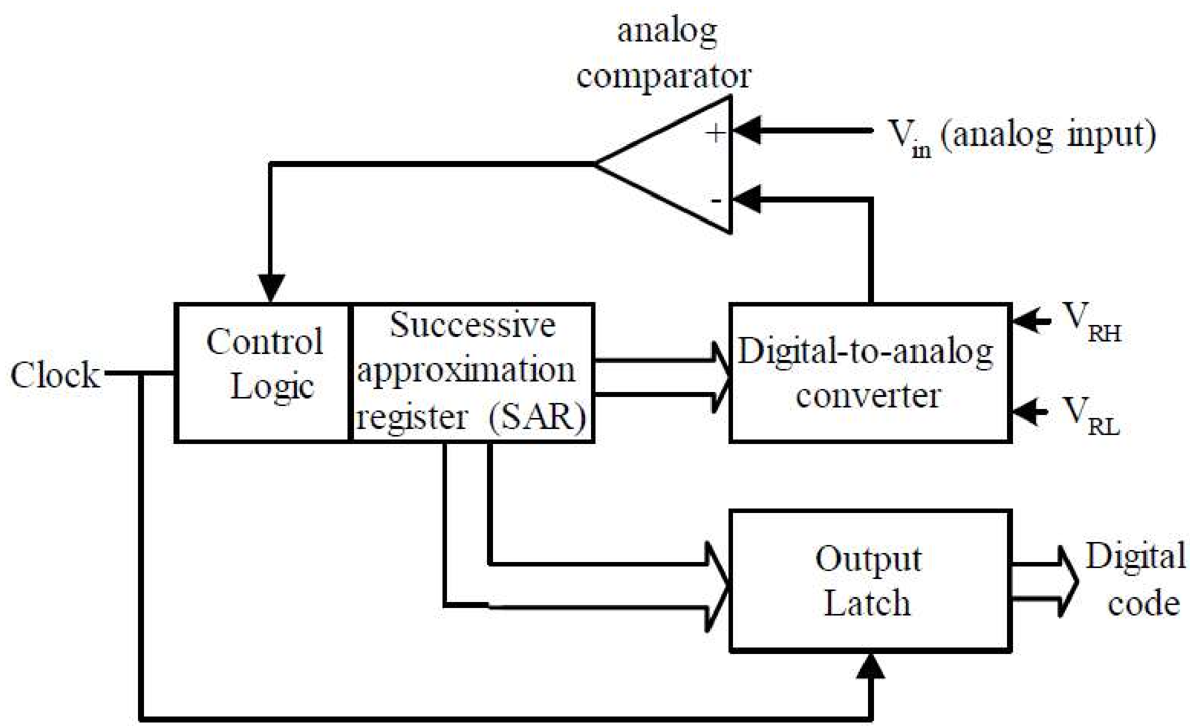




Types of A/D converters

- Flash
- Slope and double slope
- Sigma-delta
- Successive approximations (SA)

The SA converter



The optimal voltage for conversion

- The ADC requires a negative (Vref-) and positive (Veref+) references in order to execute the conversion.
- Typically, the Vref- is connected to ground when working with uni-polar signals, so the conversion starts at 0V

The optimal voltage for conversion

- In the past it was common to use positive and negative references since analog signals are bipolar in many cases, but nowadays is more customary to use the signal conditioning to convert the signal in unipolar

The optimal voltage for conversion

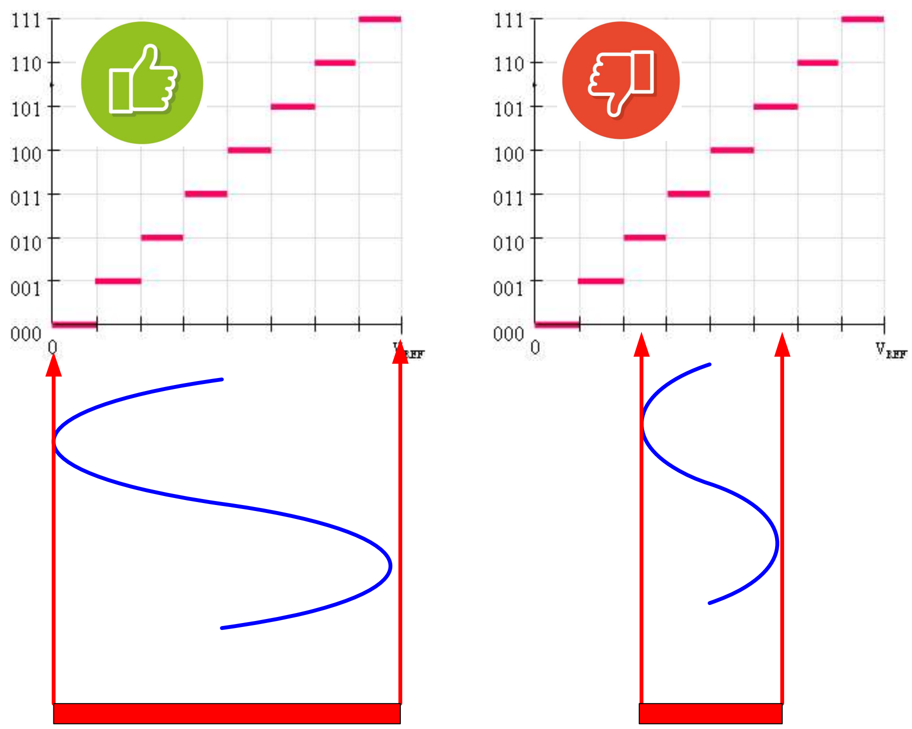
- In general ADC converters are considered to be ratiometric devices give the following:
  - A 0V voltage is converted in to a digital code 0x000
  - A voltage with the value of VDD o Vref+ is converted to the digital code $2^n - 1$
  - A voltage with the value of K*VDD sera will be converted to the code K* $(2^n - 1)$

If the VREF- is not 0, the converted value will be

$$CODE = \frac{(Vin - VREF^{-}) \times (2^{n} - 1)}{(VREF^{+} - VREF^{-})}$$

The optimal voltage for conversion

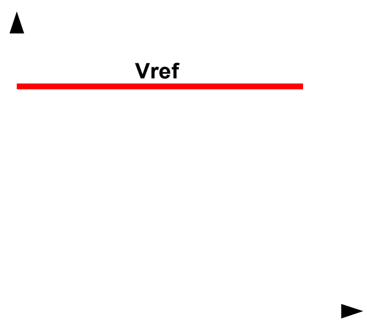
- It will be obvious that the ADC will be more precise if the signal convers the entire range of operation



Voltage scalers

- If the signal from the transducer goes from 0 to V1, but V1 value is lower tan Vref, we use a voltage scaler (amplifier) to make an optimal use of the ADC input range

$$V_{OUT} \div V_{IN} = 1 + R_2/R_1$$



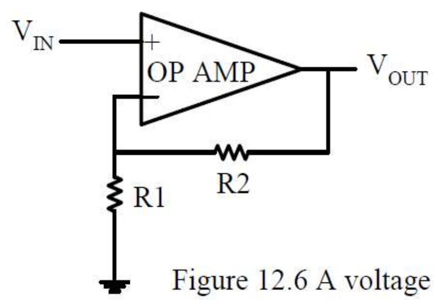
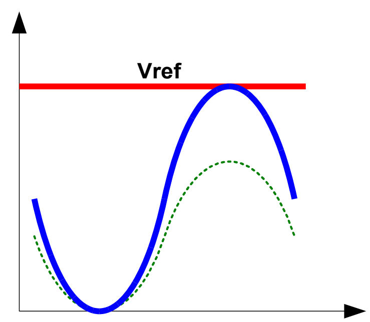


Figure 12.6 A voltage scaler



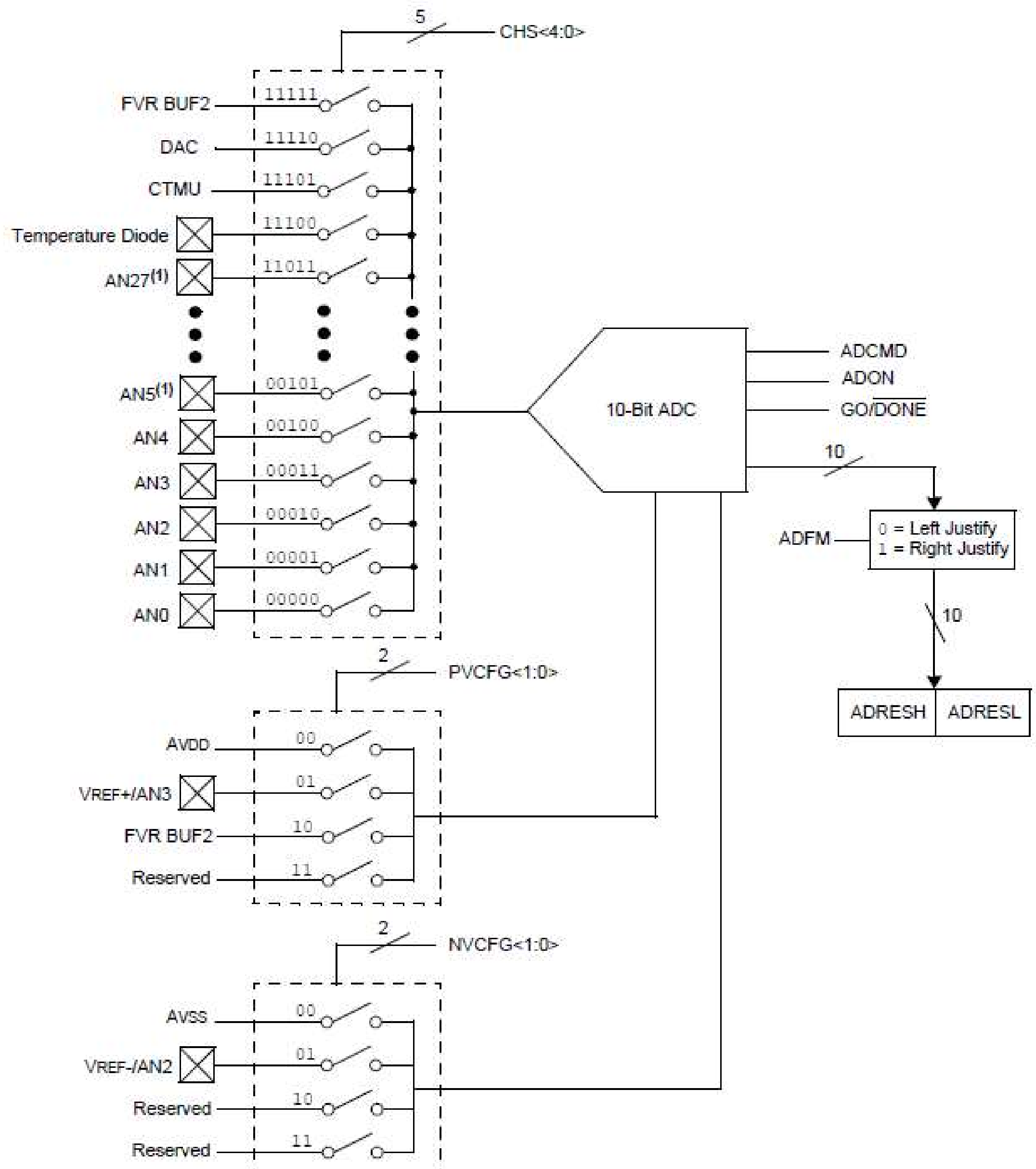
Voltage shifters

- Some transducers can have the outputs with a range from V1 to V2 where V1 is different than Vref- and V2 is also different from Vref+

*V1 could have negative values and V2 higher than Vref+

The ADC of the PIC18(L)F2X/45K50

- Its a SAR with a resolution of 10 bits
- Multiple inputs using an analog mux
- Reference values can be internal or externally supplied by the user in certain pins.
- The end of a conversion can trigger an interrupt


CHS<4:0>
5
FVR BUF2
DAC
CTMU
Temperature Diode
AN27(1)
AN5(1)
AN4
AN3
AN2
AN1
AN0
11111
11110
11101
11100
11011
00101
00100
00011
00010
00001
00000
10-Bit ADC
ADCMD
ADON
GO/DONE
10
ADFM
0 = Left Justify
1 = Right Justify
10
ADRESH
ADRESL
PVCFG<1:0>
2
AVDD
VREF+/AN3
FVR BUF2
Reserved
00
01
10
11
NVCFG<1:0>
2
AVSS
VREF-/AN2
Reserved
Reserved
00
01
10
11

ADC configuration basics

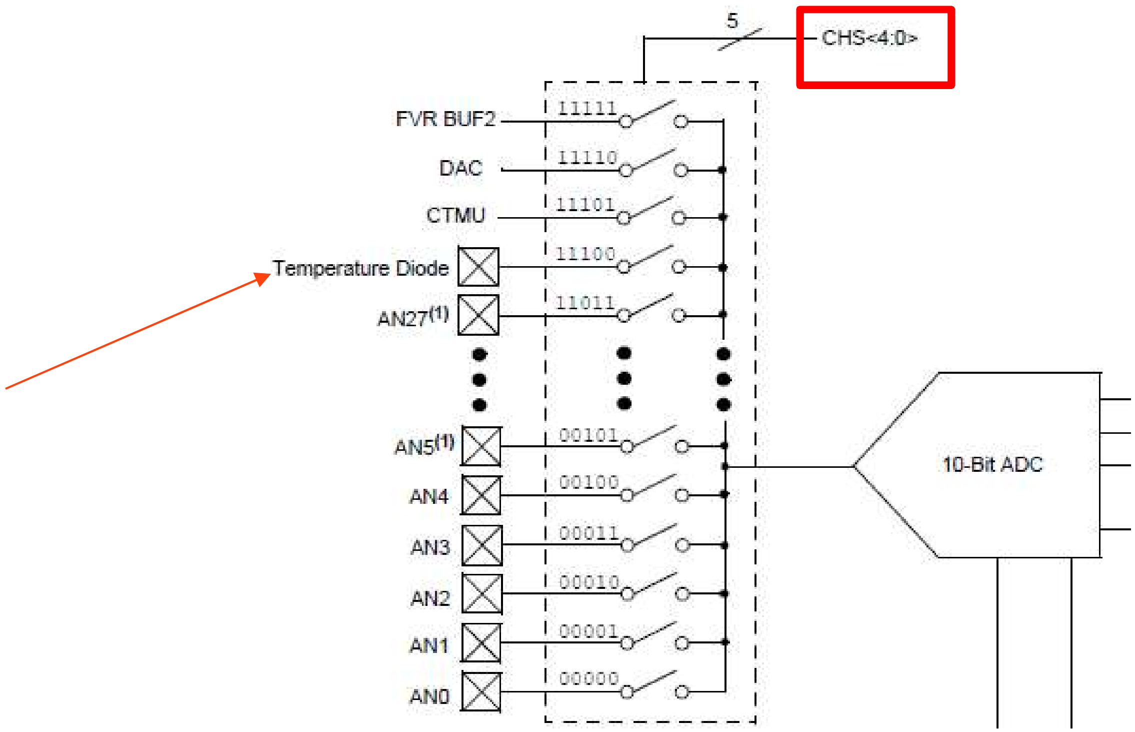
- To configure the ADC you must consider the following:
  - Port configuration to analog
  - Channel selection to direct to a pin
  - Reference voltage selection
  - Conversion time base selection
  - Interrupt control (if required)
  - Result format type

Port selection

- Registers ANSELx and TRISx configure the inputs of the ADC
- The port pin must be configured as input using TRISx (set to 1)
- The port pin must be configured as analog ANSELx (set to 1)

Channel selection

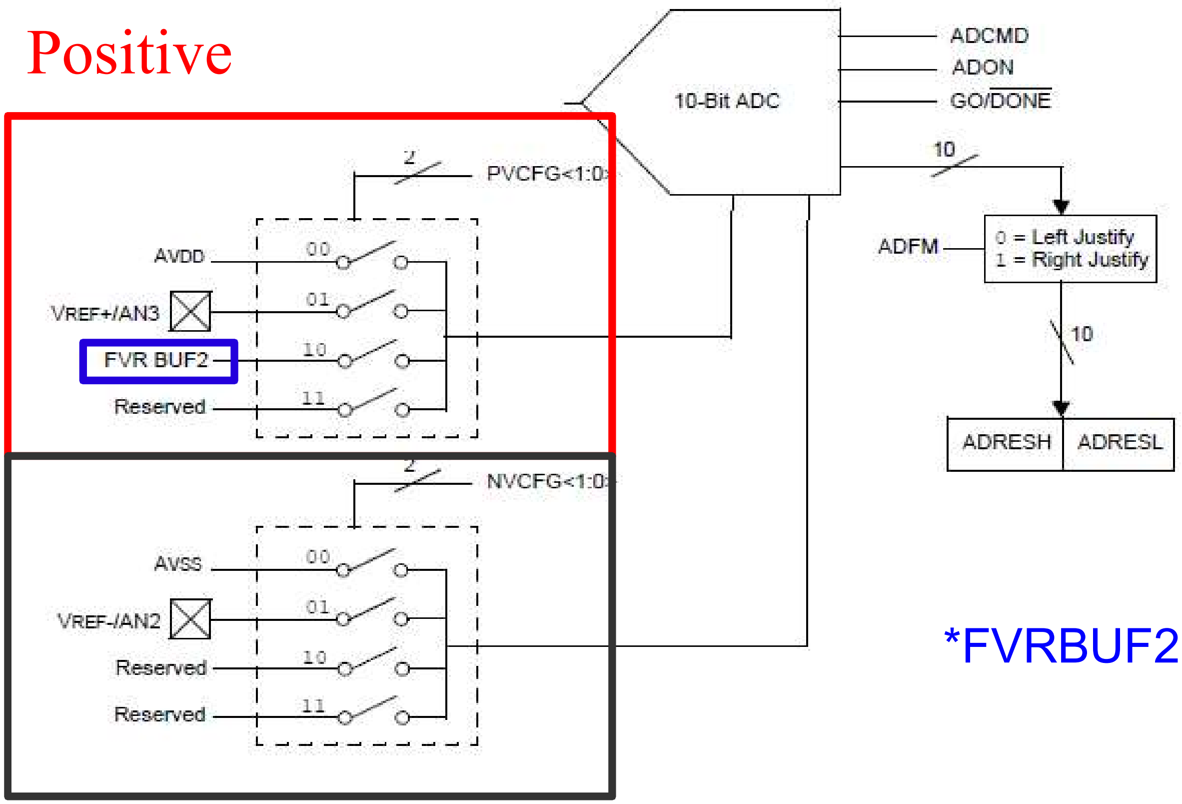
- The CHS of register ADCCON0 tell which channel (pin) input will be directed to the ADC
- If you are digitizing two several inputs, consider that you must have a time guard to change between channels



Reference voltage selection

- Bits PVCFG[1:0] and NVCFG[1:0] in the ADCON1 register define the source for the negative and positive reference voltages

Positive

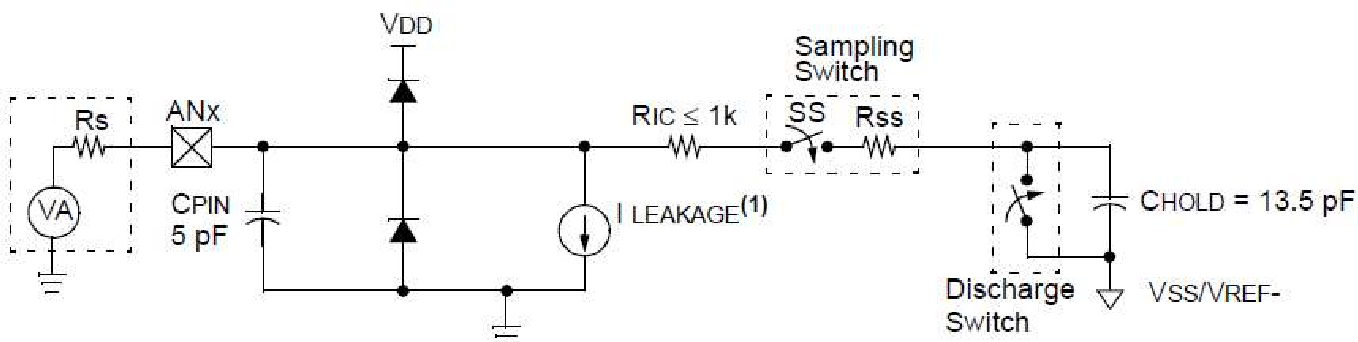


*FVRBUF2 (fixed internal)

Negative

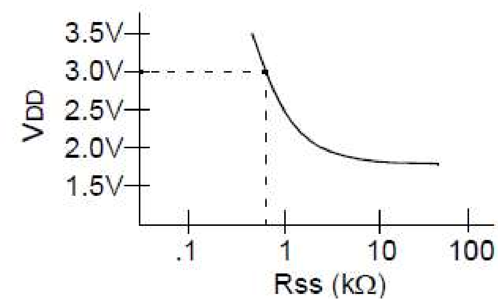
Acquisition time

- The circuit model of an analog input up to the ADC is as follows:



Legend:

CPIN	= Input Capacitance
I LEAKAGE	= Leakage current at the pin due to various junctions
RIC	= Interconnect Resistance
SS	= Sampling Switch
CHOLD	= Sample/Hold Capacitance



The Acquisition time

- In order to maintain a stable voltage during the conversion, a “picture” of the amplitude is take and is stored on the CHOLD capacitor (sample-hold)
- Since there is a chain of resistor in the input path, the CHOLD capacitor will take some time to charge and we will have a dependence of RS, RIC and RSS and this will change depending on your design
- In order to have the true value of the input voltage, we must give enough time to the SS switch if not we will have an error

The Acquisition time

- Register ADCON2 allows to select the acquisition time that will be used when the conversion is started
- The setting of ACQT[2:0] bits allow to configure the time from 2 to 20 times the duration of the conversion clock period.
- The microcontroller will automatically wait for this time before the conversion is stared by the user.
- The programmer can also control the acquisition timer setting manually setting the ACQT[2:0] = 000b

The conversion clock

- The selection of the ADC conversion clock is made with the ADCS bits in the ADCON2 register and you can have the following options:
  - Fosc/2, Fosc/4,Fosc/8,Fosc/16
  - Fosc/32,Fosc/64
  - FRCC (Dedicated oscillator)
- A complete 10 bits conversion requires 11 cycles of the clock (TAD) to be completed

Conversion clock

TABLE 18-1: ADC CLOCK PERIOD (TAD) vs. DEVICE OPERATING FREQUENCIES

AD Clock Period (TAD)		Device Frequency (FOSC)			
ADC Clock Source	ADCS<2:0>	48 MHz	16 MHz	4 MHz	1 MHz
Fosc/2	000	41.17 ns(2)	125 ns(2)	500 ns(2)	2.0 μs
Fosc/4	100	83.3 ns(2)	250 ns(2)	1.0 μs	4.0 μs(3)
Fosc/8	001	166.7 ns(2)	500 ns(2)	2.0 μs	8.0 μs(3)
Fosc/16	101	333.3 ns(2)	1.0 μs	4.0 μs(3)	16.0 μs(3)
Fosc/32	010	666.7 ns(2)	2.0 μs	8.0 μs(3)	32.0 μs(3)
Fosc/64	110	1.3 μs	4.0 μs(3)	16.0 μs(3)	64.0 μs(3)
F_{RC}	011	1-4 μs(1,4)	1-4 μs(1,4)	1-4 μs(1,4)	1-4 μs(1,4)

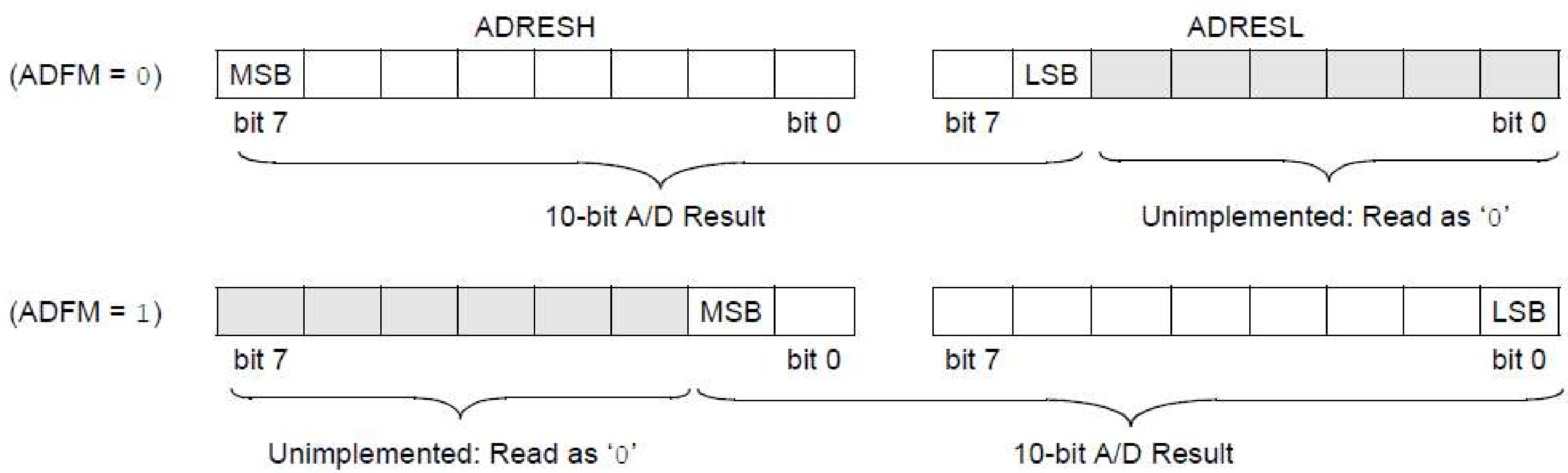
TAD min value is 1usec

Interrupts

- The ADC can generate an interrupt at the end of the conversion
- The interrupt is enabled setting the ADIE bit on register PIE1
- At the end of the conversion, the PIR1 is turned on after each conversion but it must be cleared by firmware.

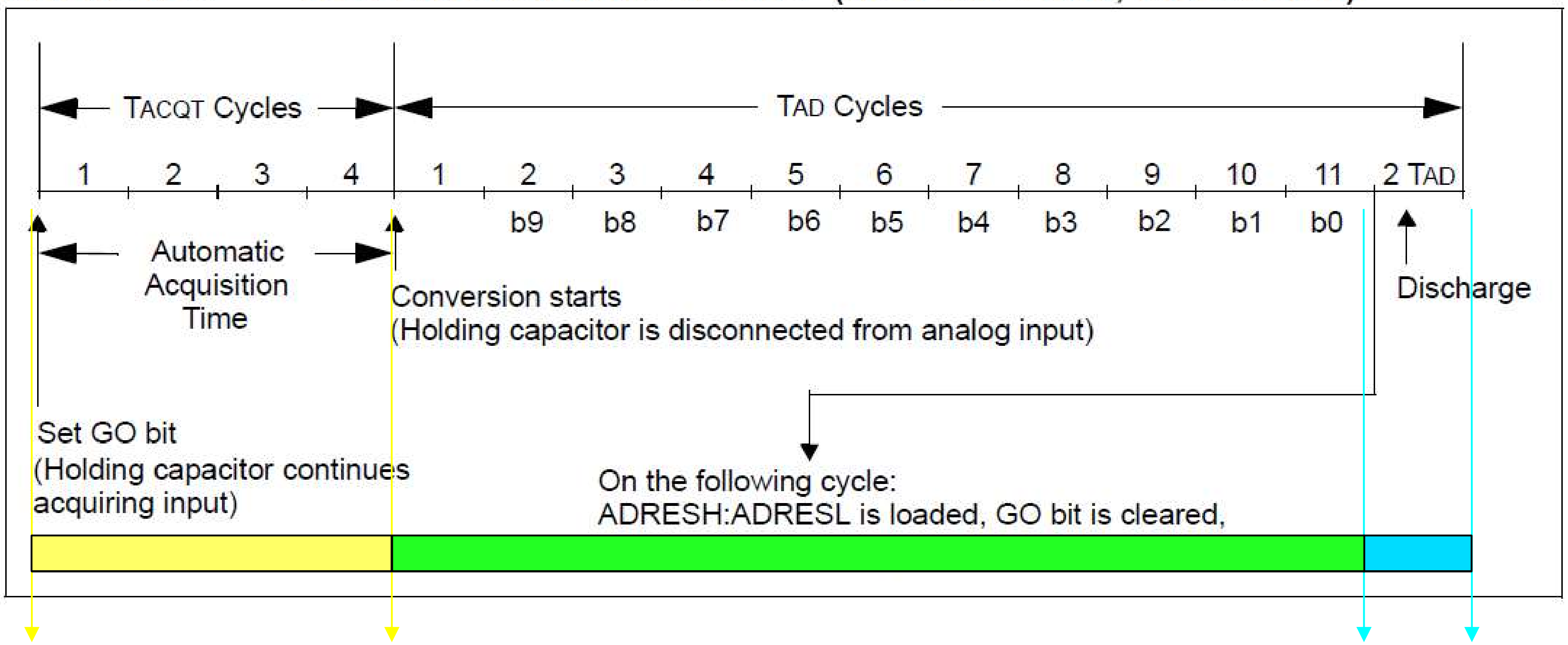
Result format

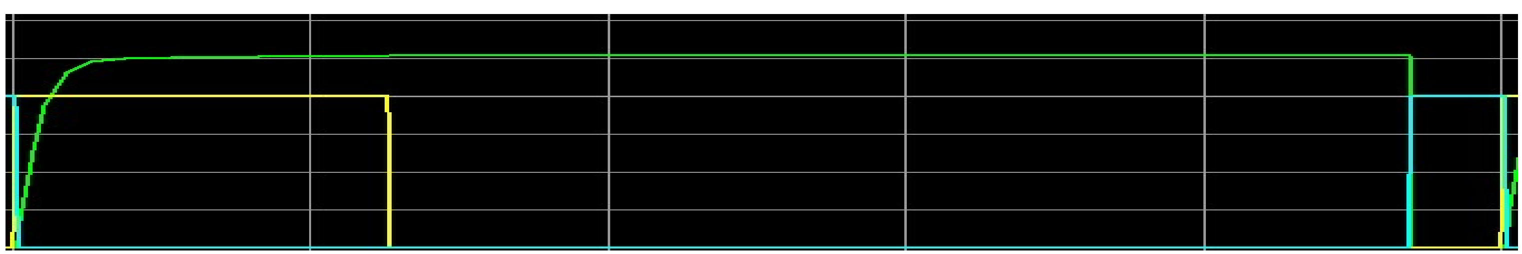
- The conversion result can be provided in two formats ; left or right justified. This is defied in the ADCON2 register



Operation of the ADC

FIGURE 17-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)





Delay between conversions

- When a conversion ends (indicated by GO/DONE or the interrupt flag) you must wait at least 2TAD before starting the next conversion

Procedure to make an AD conversion:

1 Configure the input port
- Se pin(s) at input (TRIS)
- Set pin(s) as analog (ANSEL)

2 Configure the ADC peripheral
- Turn on the ADC
- Select the input channel
- Configure the reference
- Select the result output format
- Select the acquisition time
- Select the conversion clock

Procedure to make an AD conversion:

3 Configure the interrupt (Optional)
- Clear the ADC interrupt flag
- Enable the ADC interrupt
- Enable the global interrupt

4 Wait the acquisition time (if manual)

5 Start the conversion setting the flag GO/DONE to one

Procedure to make an AD conversion:

6 Wait for the ADC to finish using one of the following methods
- Poll GO/DONE bit (it will clear at the end)
- Wait for the interrupt flag to turn on

7 Read the ADC result

8 Clear the interrupt flag (if used)

Registers

REGISTER 18-1: ADCON0: A/D CONTROL REGISTER 0

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CHS<4:0>					GO/$\overline{DONE}$	ADON
bit 7							bit 0

bit 6-2 **CHS<4:0>: Analog Channel Select bits**

00000 = AN0
00001 = AN1
00010 = AN2
00011 = AN3
00100 = AN4
00101 = AN5(1)
00110 = AN6(1)
00111 = AN7(1)
01000 = AN8
01001 = AN9
01010 = AN10
01011 = AN11
01100 = AN12
01101 = AN13
01110 = AN14
01111 = AN15
10000 = AN16
10001 = AN17
10010 = AN18
10011 = AN19
10100 = AN20(1)
10101 = AN21(1)
10110 = AN22(1)
10111 = AN23(1)
11000 = AN24(1)
11001 = AN25(1)
11010 = AN26(1)
11011 = AN27(1)
11100 = Temperature Diode
11101 = CTMU
11110 = DAC
11111 = FVR BUF2 (1.024V/2.048V/4.096V Fixed Voltage Reference)(2)

bit 1 **GO/$\overline{DONE}$:** A/D Conversion Status bit
1 = A/D conversion cycle in progress. Setting this bit starts an A/D conversion cycle.
This bit is automatically cleared by hardware when the A/D conversion has completed.
0 = A/D conversion completed/not in progress

bit 0 **ADON:** ADC Enable bit
1 = ADC is enabled
0 = ADC is disabled and consumes no operating current

Registers

REGISTER 18-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
TRIGSEL	—	—	—	PVCFG<1:0>		NVCFG<1:0>	
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TRIGSEL**: Special Trigger Select bit
1 = Selects the special trigger from CTMU
0 = Selects the special trigger from CCP2

bit 6-4 **Unimplemented:** Read as '0'

bit 3-2 **PVCFG<1:0>:** Positive Voltage Reference Configuration bits
00 = A/D VREF+ connected to internal signal, AVDD
01 = A/D VREF+ connected to external pin, VREF+
10 = A/D VREF+ connected to internal signal, FVR BUF2
11 = Reserved

bit 1-0 **NVCFG<1:0>:** Negative Voltage Reference Configuration bits
00 = A/D VREF- connected to internal signal, AVSS
01 = A/D VREF- connected to external pin, VREF-
10 = Reserved
11 = Reserved

Registers

REGISTER 18-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT<2:0>			ADCS<2:0>		
bit 7							bit 0

bit 7 **ADFM:** A/D Conversion Result Format Select bit

1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as '0'

bit 5-3 **ACQT<2:0>:** A/D Acquisition time select bits. Acquisition time is the duration that the A/D charge holding capacitor remains connected to A/D channel from the instant the GO/$\overline{\text{DONE}}$ bit is set until conversions begins.

000 = 0(1)
001 = 2 TAD
010 = 4 TAD
011 = 6 TAD
100 = 8 TAD
101 = 12 TAD
110 = 16 TAD
111 = 20 TAD

bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits

000 = FOSC/2
001 = FOSC/8
010 = FOSC/32
011 = FRC(1) (clock derived from a dedicated internal oscillator = 600 kHz nominal)
100 = FOSC/4
101 = FOSC/16
110 = FOSC/64
111 = FRC(1) (clock derived from a dedicated internal oscillator = 600 kHz nominal)

Note 1: When the A/D clock source is selected as FRC then the start of conversion is delayed by one instruction cycle after the GO/$\overline{\text{DONE}}$ bit is set to allow the SLEEP instruction to be executed.

Registers

LEFT JUSTIFIED

REGISTER 18-4: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ADRES<9:2>							
bit 7							bit 0

REGISTER 18-5: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ADRES<1:0>		r	r	r	r	r	r
bit 7							bit 0

RIGHT JUSTIFIED

REGISTER 18-6: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 1

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
r	r	r	r	r	r	ADRES<9:8>	
bit 7							bit 0

REGISTER 18-7: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 1

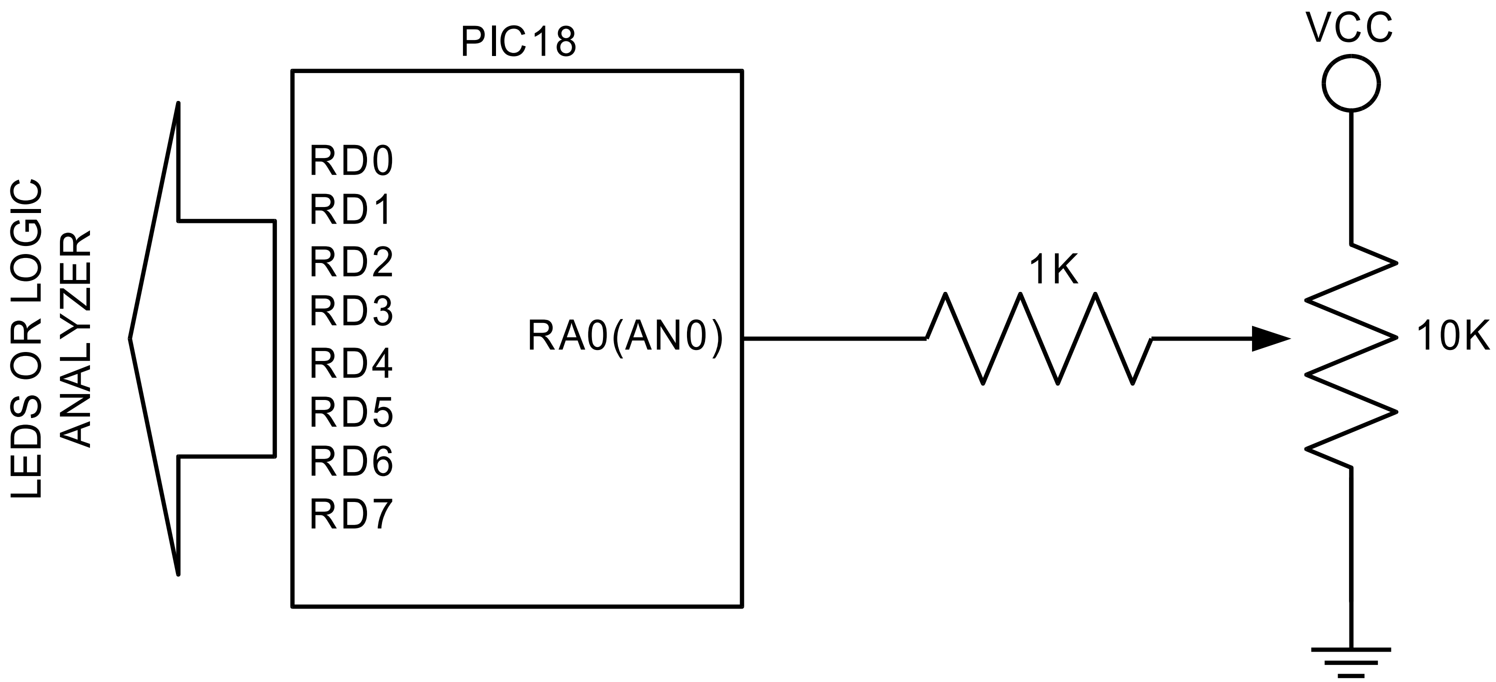
R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ADRES<7:0>							
bit 7							bit 0

Registers related to the ADC

TABLE 18-2: REGISTERS ASSOCIATED WITH A/D OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
ADCON0	—	CHS<4:0>					GO/$\overline{DONE}$	ADON	306
ADCON1	TRIGSEL	—	—	—	PVCFG<1:0>		NVCFG<1:0>		307
ADCON2	ADFM	—	ACQT<2:0>			ADCS<2:0>			308
ADRESH	A/D Result, High Byte								309
ADRESL	A/D Result, Low Byte								309
ANSELA	—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0	154
ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	155
ANSELC	ANSC7	ANSC6	—	—	—	ANSC2	—	—	155
ANSELD[1]	ANSD7	ANSD6	ANSD5	ANSD4	ANSD3	ANSD2	ANSD1	ANSD0	155
ANSELE[1]	—	—	—	—	—	ANSE2	ANSE1	ANSE0	156
CTMUCONH	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	335
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	IOCIE	TMR0IF	INT0IF	IOCIF	120
IPR1	ACTIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	129
PIE1	ACTIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	126
PIR1	ACTIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	123
PMD1	—	MSSPMD	CTMUMD	CMP2MD	CMP1MD	ADCMD	CCP2MD	CCP1MD	65
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	156
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	156
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	156
TRISD[1]	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	156
TRISE	WPUE3	—	—	—	—	TRISE2[1]	TRISE1[1]	TRISE0[1]	156

Example


PIC18
RD0
RD1
RD2
RD3
RD4
RD5
RD6
RD7
RA0(AN0)
LEDS OR LOGIC ANALYZER
1K
VCC
10K

Example

- For the circuit, write a program that does the following:
  - Read the voltage in the potentiometer
  - Discard the least significant bits of the A/D
  - Write the result of the conversion to PORTD
  - Repeat the later procedure for ever

Example

- Assumptions
  - The oscillator frequency is16Mhz
  - The frequency of the conversion clock is Fosc/64
  - Use the most convenient justification for the result
  - Use the analog input AN0
  - Use an acquisition time of 20TAD
  - Use VDD and GND as Vref+/Vref-
  - Do not use interrupts

Example

REGISTER 18-1: ADCON0: A/D CONTROL REGISTER 0

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CHS<4:0>					GO/$\overline{\text{DONE}}$	ADON
bit 7	0	0	0	0	0	0	1 bit 0

bit 6-2 **CHS<4:0>: Analog Channel Select bits**

00000 = AN0
00001 = AN1
00010 = AN2
00011 = AN3
00100 = AN4
00101 = AN5[(1)]
00110 = AN6[(1)]
00111 = AN7[(1)]
01000 = AN8
01001 = AN9
01010 = AN10
01011 = AN11
01100 = AN12
01101 = AN13
01110 = AN14
01111 = AN15
10000 = AN16
10001 = AN17
10010 = AN18
10011 = AN19
10100 = AN20[(1)]
10101 = AN21[(1)]
10110 = AN22[(1)]
10111 = AN23[(1)]
11000 = AN24[(1)]
11001 = AN25[(1)]
11010 = AN26[(1)]
11011 = AN27[(1)]
11100 = Temperature Diode
11101 = CTMU
11110 = DAC
11111 = FVR BUF2 (1.024V/2.048V/4.096V Fixed Voltage Reference)[(2)]

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
1 = A/D conversion cycle in progress. Setting this bit starts an A/D conversion cycle.
This bit is automatically cleared by hardware when the A/D conversion has completed.
0 = A/D conversion completed/not in progress

bit 0 **ADON:** ADC Enable bit
1 = ADC is enabled
0 = ADC is disabled and consumes no operating current

Example

REGISTER 18-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
TRIGSEL	—	—	—	PVCFG<1:0>		NVCFG<1:0>	
bit 7 X	X	X	X	0	0	0	0 bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TRIGSEL**: Special Trigger Select bit
1 = Selects the special trigger from CTMU
0 = Selects the special trigger from CCP2

bit 6-4 **Unimplemented:** Read as '0'

bit 3-2 **PVCFG<1:0>:** Positive Voltage Reference Configuration bits
00 = A/D VREF+ connected to internal signal, AVDD
01 = A/D VREF+ connected to external pin, VREF+
10 = A/D VREF+ connected to internal signal, FVR BUF2
11 = Reserved

bit 1-0 **NVCFG<1:0>:** Negative Voltage Reference Configuration bits
00 = A/D VREF- connected to internal signal, AVSS
01 = A/D VREF- connected to external pin, VREF-
10 = Reserved
11 = Reserved

REGISTER 18-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT<2:0>			ADCS<2:0>		
bit 7 0	X	1	1	1	1	1	0 bit 0

bit 7 **ADFM:** A/D Conversion Result Format Select bit
1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as '0'

bit 5-3 **ACQT<2:0>:** A/D Acquisition time select bits. Acquisition time is the duration that the A/D charge holding capacitor remains connected to A/D channel from the instant the GO/$\overline{\text{DONE}}$ bit is set until conversions begins.
000 = 0(1)
001 = 2 TAD
010 = 4 TAD
011 = 6 TAD
100 = 8 TAD
101 = 12 TAD
110 = 16 TAD
111 = 20 TAD

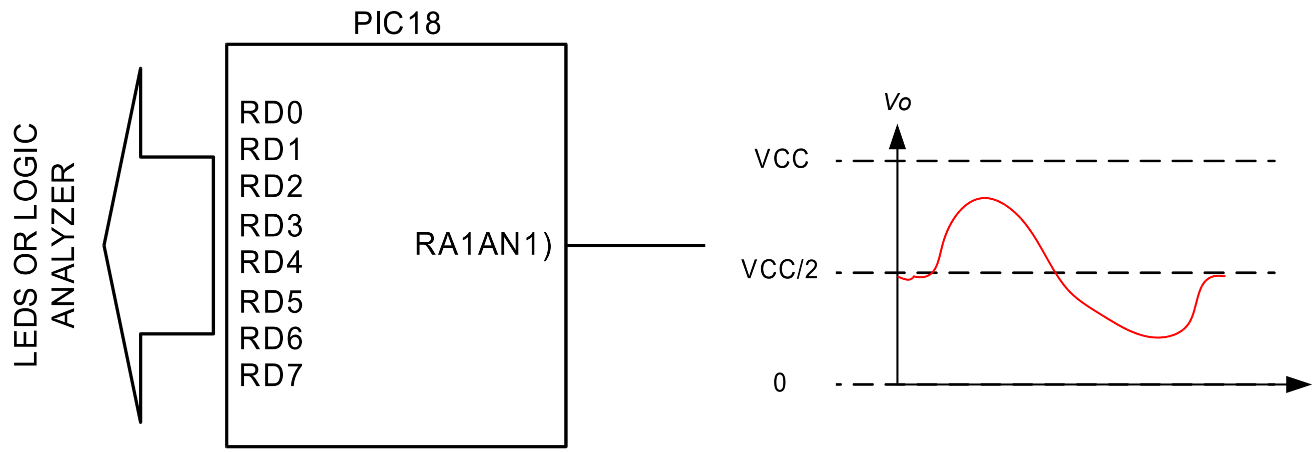
bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits
000 = FOSC/2
001 = FOSC/8
010 = FOSC/32
011 = FRC(1) (clock derived from a dedicated internal oscillator = 600 kHz nominal)
100 = FOSC/4
101 = FOSC/16
110 = FOSC/64
111 = FRC(1) (clock derived from a dedicated internal oscillator = 600 kHz nominal)

Note 1: When the A/D clock source is selected as FRC then the start of conversion is delayed by one instruction cycle after the GO/$\overline{\text{DONE}}$ bit is set to allow the SLEEP instruction to be executed.

```c
#include<xc.h>
void int_oscillator(void);          //Inits oscilator to 16Mhz
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
//+ Main program
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
void main(void){

    int_oscillator();        //Init osc
    //Output to port D
    TRISD = 0b00000000;     // Output
    ANSELD = 0b00000000;    // Digital
    PORTD = 0b00000000;     // Init value

    //Input AN0 is mapped to RA0 so RA must be analog
    TRISAbits.RA0 = 1;      // RA0 Input
    ANSELAbits.ANSA0 = 1;   // Analog

    //Analog converter configuration

    ADCON0 = 0b00000001;    //Select channel AN0 and turn on the ADC
    ADCON1 = 0b00000000;    //Vref+ (VCC), Vref-(GND)
    ADCON2 = 0b00111110;    //Just left, 20TAD, FOSC/4

    //ADC is configured so lets convert
    //
    while(1){
        ADCON0bits.DONE = 1;        //Start conversion
        while(ADCON0bits.DONE==1);  //Wait to end
        PORTD = ADRESH;             //Hign part is copied to the port output
    }
}
```

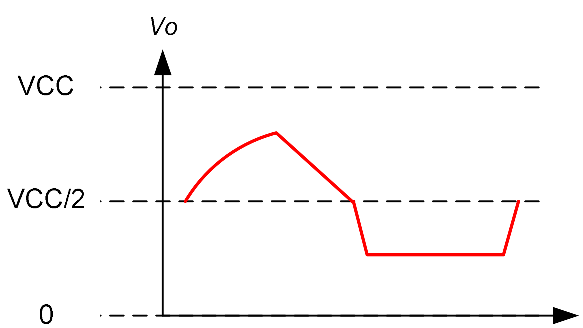
Example

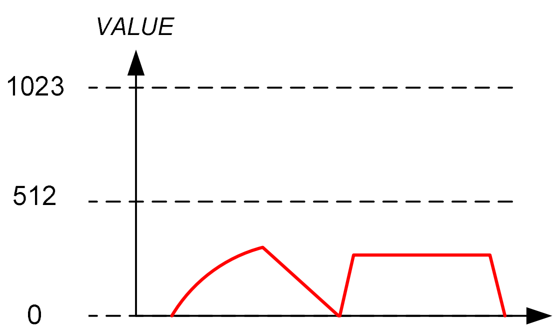
- Write a program that obtains the absolute value of a signal that has an offset at VCC/2
- The circuit and signal are as follow



Example

- Use the same settings for the ADC as the previous example
- Since the signal has an offset, the value that the ADC reads when the signal is "0" (VCC/2) will be around 512
- Since we will obtain the absolute value, we must eliminate the offset from the result

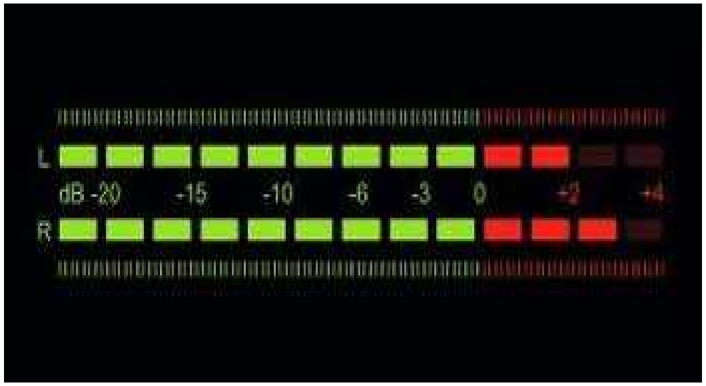




Example

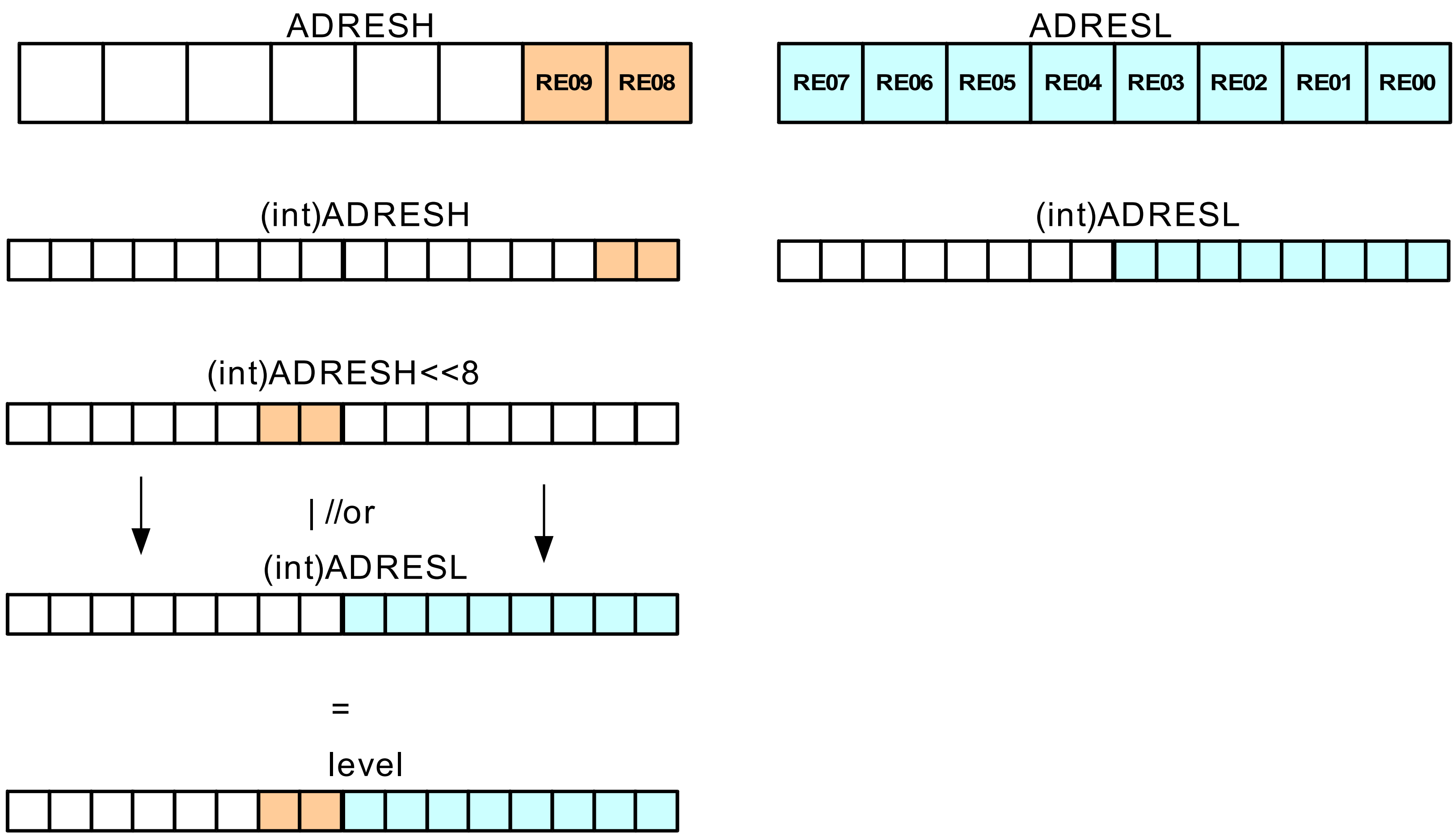
- Show the magnitude of the signal using the LED to mimic a magnitude bar graph using the relation shown in the table

LED	SIGNAL VALUE
D8	456
D7	399
D6	342
D5	285
D4	228
D3	171
D2	114
D1	57
NOTHING	<



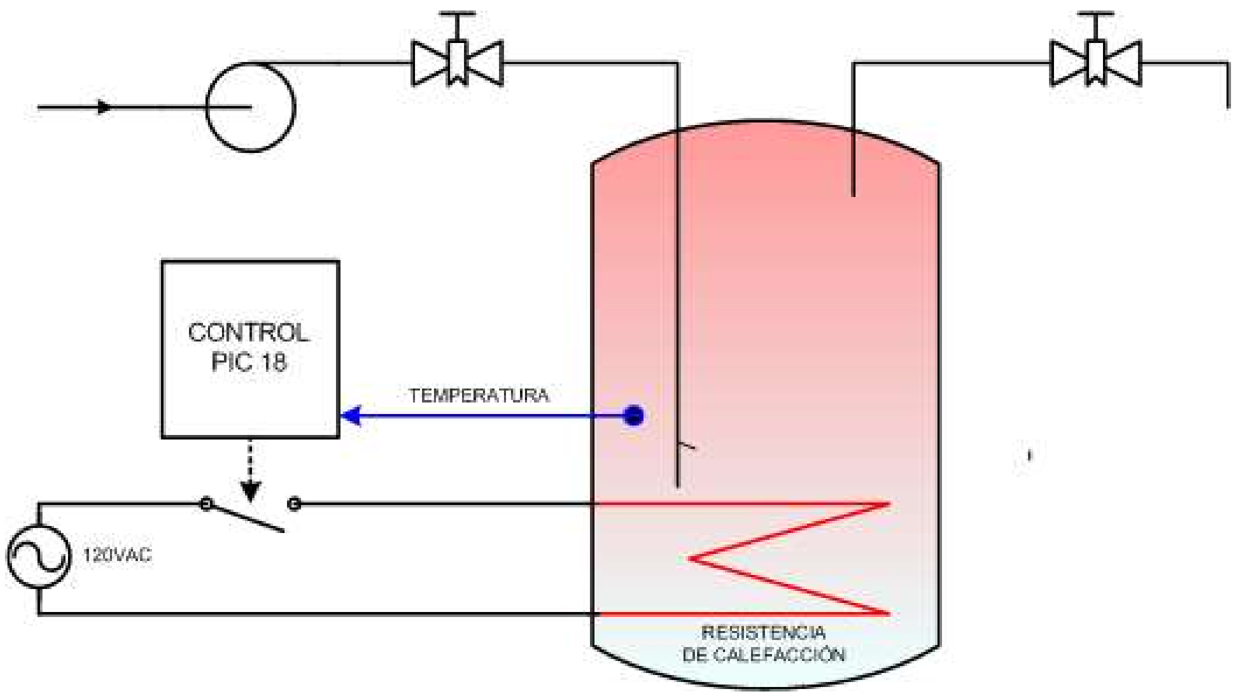
```
#include<xc.h>
void int_oscillator(void);           //Inits oscilator to 16Mhz
void main(void){
int level ;               //Stores the ADC result.
    int_oscillator();  //Init oscillator
    //Output is PORTD
    TRISD = 0b00000000;     // Output
    ANSELD = 0b00000000;    // Digital
    PORTD = 0b00000000;     // Init value
    //Input will be AN1 is assigned RA1
    TRISAbits.RA1 = 1;  // RA0 será entrda
    ANSELAbits.ANSA1 = 1;   // Forzamos a que sea analogica
    //COnfiguramos converditor A/D
    ADCON0 = 0b00000101;    //Select AN1 turn on ADC
    ADCON1 = 0b00000000;    //Vref+ (VCC), Vref-(GND)
    ADCON2 = 0b10111110;    //Right Just, 20TAD, FOSC/64
    while(1){
        ADCON0bits.DONE = 1;        //Init conversion
        while(ADCON0bits.DONE==1);  //Waits
        //Convert the 2 bytes ADRESH:ADRESSL--> to an INT (16 bits)
        level = (int)ADRESH<<8  |  (int)ADRESL;
        level = level - 512;            //Remove offset
        if( level < 0) level = -level;  //Absolute value
        if( level > 57) PORTDbits.RD0 = 1; else PORTDbits.RD0 = 0;
        if( level > 114) PORTDbits.RD1 = 1; else PORTDbits.RD1 = 0;
        if( level > 171) PORTDbits.RD2 = 1; else PORTDbits.RD2 = 0;
        if( level > 228) PORTDbits.RD3 = 1; else PORTDbits.RD3 = 0;
        if( level > 285) PORTDbits.RD4 = 1; else PORTDbits.RD4 = 0;
        if( level > 342) PORTDbits.RD5 = 1; else PORTDbits.RD5 = 0;
        if( level > 399) PORTDbits.RD6 = 1; else PORTDbits.RD6 = 0;
        if( level > 456) PORTDbits.RD7 = 1; else PORTDbits.RD7 = 0;
```

```
level = (int)ADRESH<<8   |   (int)ADRESL;
```



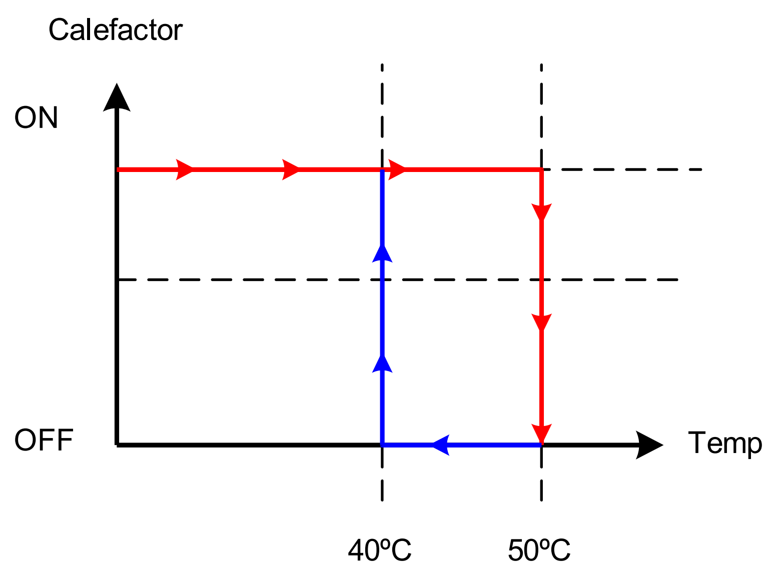
Example

- We want to control the temperature of the following hydraulic system



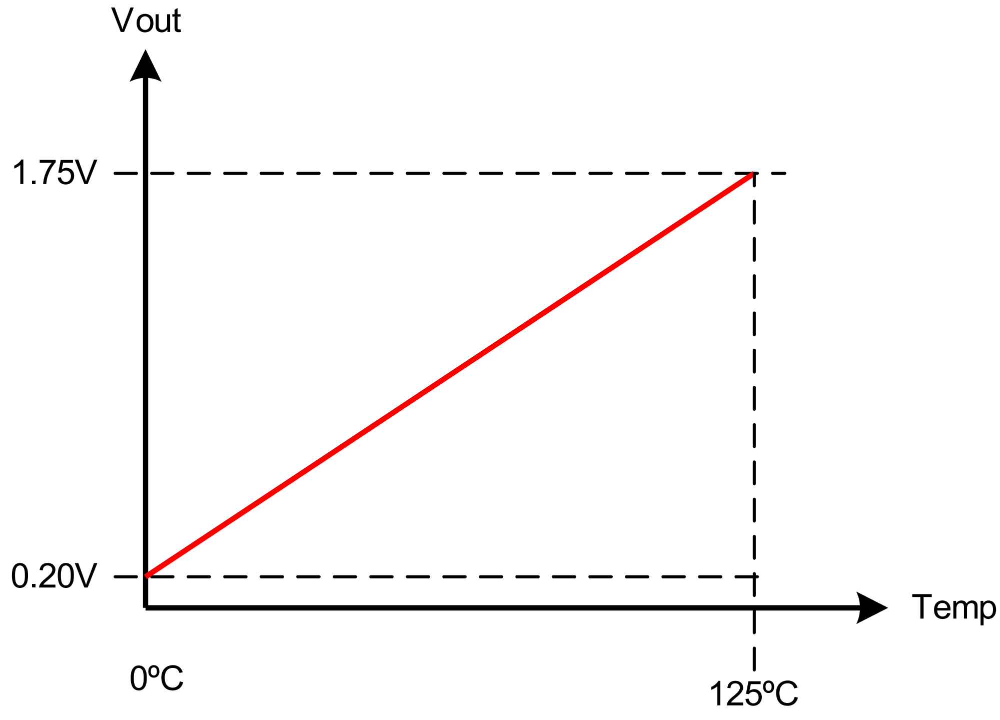
Ejemplo

- The control algorithm is ON-OFF and it must maintain the temperature between 40 y 50C
- The actuator is a resistive heater that is powered by 120VAC, to activate the resistor the controller provides a dry contact.



Example

- The sensor is active, and has the following voltage to temperature curve



Example

- We will use the following external voltage reference:

LM4128/LM4128Q
SOT-23 Precision Micropower Series Voltage Reference

General Description

Ideal for space critical applications, the LM4128 precision voltage reference is available in the SOT-23 surface-mount package. The LM4128's advanced design eliminates the need for an external stabilizing capacitor while ensuring stability with capacitive loads up to 10 µF, thus making the LM4128 easy to use.

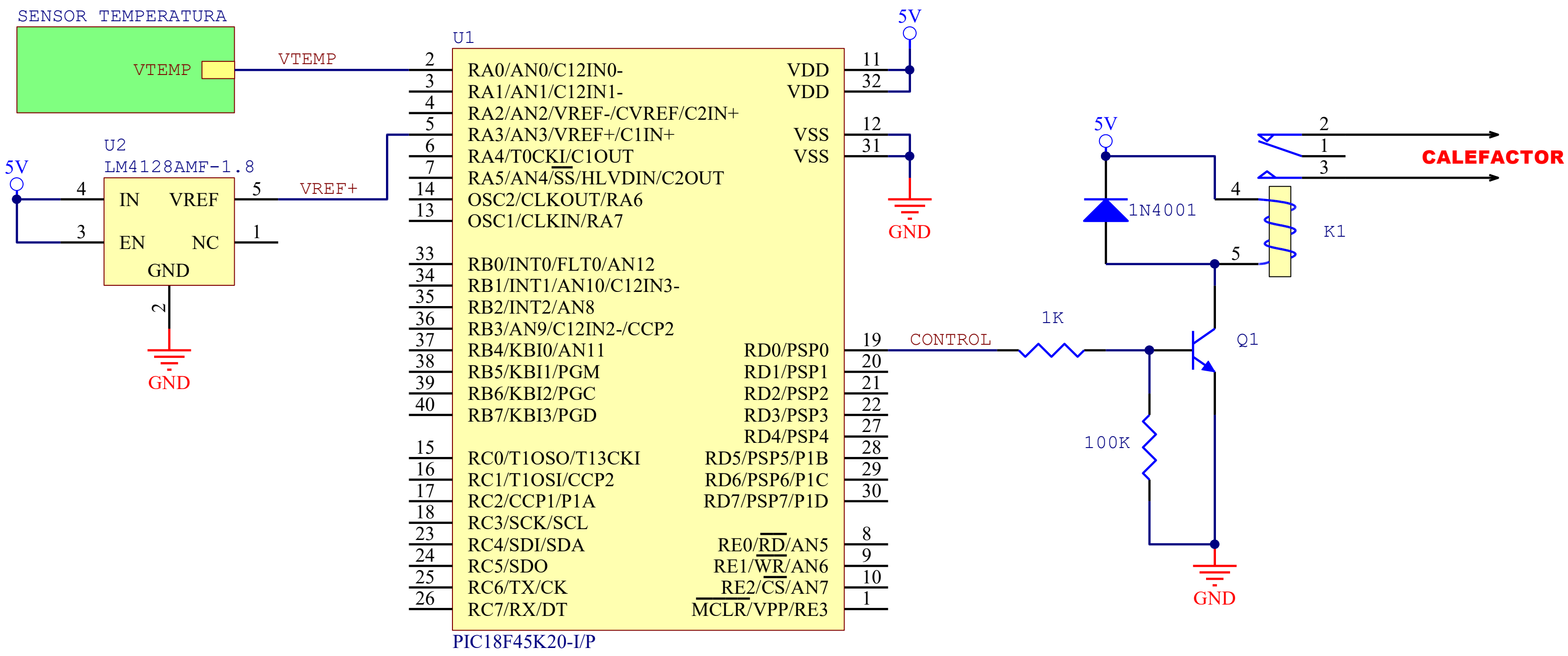
Series references provide lower power consumption than shunt references, since they do not have to idle the maximum possible load current under no load conditions. This advantage, the low quiescent current (60 µA), and the low dropout voltage (400 mV) make the LM4128 ideal for battery-powered solutions.

The LM4128 is available in four grades (A, B, C, and D) for greater flexibility. The best grade devices (A) have an initial accuracy of 0.1% with guaranteed temperature coefficient of 75 ppm/°C or less, while the lowest grade parts (D) have an initial accuracy of 1.0% and a tempco of 100 ppm/°C.

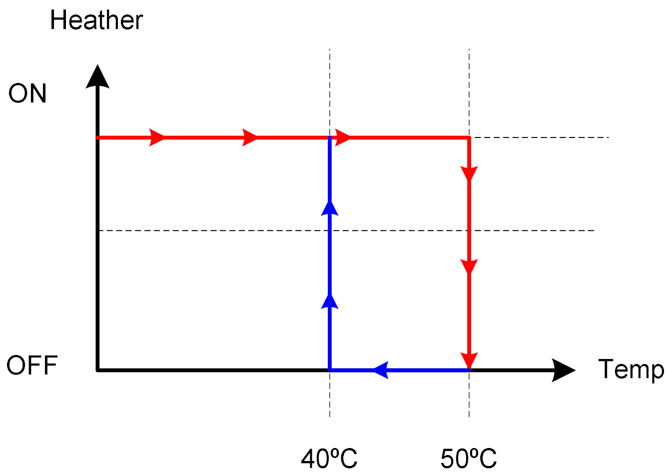
Features

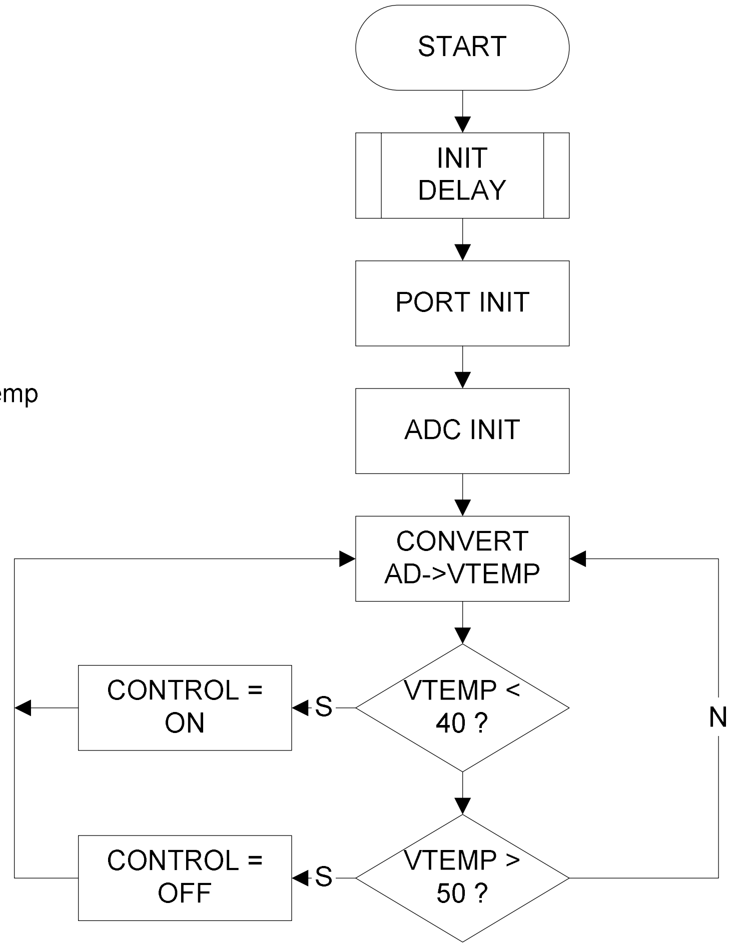
- Output voltage initial accuracy 0.1%
- Low temperature coefficient 75 ppm/°C
- Low Supply Current, 60 µA
- Enable pin allowing a 3 µA shutdown mode
- Up to 20 mA output current
- Voltage options 1.8V, 2.048V, 2.5V, 3.0V, 3.3V, 4.096V
- Custom voltage options available (1.8V to 4.096V)
- V_{IN} range of V_{REF} + 400 mV to 5.5V @10 mA
- Stable with low ESR ceramic capacitors
- SOT23-5 Package
- –40°C to 125°C junction temperature range
- LM4128AQ/BQ/CQ/DQ are AEC-Q100 Grade 1 qualified and are manufactured on an Automotive Grade Flow

Example


SENSOR TEMPERATURA
VTEMP
VTEMP
U1
5V
RA0/AN0/C12IN0-
RA1/AN1/C12IN1-
RA2/AN2/VREF-/CVREF/C2IN+
RA3/AN3/VREF+/C1IN+
RA4/T0CKI/C1OUT
RA5/AN4/SS/HLVDIN/C2OUT
OSC2/CLKOUT/RA6
OSC1/CLKIN/RA7
RB0/INT0/FLT0/AN12
RB1/INT1/AN10/C12IN3-
RB2/INT2/AN8
RB3/AN9/C12IN2-/CCP2
RB4/KBI0/AN11
RB5/KBI1/PGM
RB6/KBI2/PGC
RB7/KBI3/PGD
RC0/T1OSO/T13CKI
RC1/T1OSI/CCP2
RC2/CCP1/P1A
RC3/SCK/SCL
RC4/SDI/SDA
RC5/SDO
RC6/TX/CK
RC7/RX/DT
VDD
VDD
VSS
VSS
RD0/PSP0
RD1/PSP1
RD2/PSP2
RD3/PSP3
RD4/PSP4
RD5/PSP5/P1B
RD6/PSP6/P1C
RD7/PSP7/P1D
RE0/RD/AN5
RE1/WR/AN6
RE2/CS/AN7
MCLR/VPP/RE3
PIC18F45K20-I/P
GND
U2
LM4128AMF-1.8
IN
VREF
EN
NC
GND
VREF+
CONTROL
1K
100K
Q1
1N4001
K1
CALEFACTOR

Example


Heather
ON
OFF
Temp
40ºC
50ºC


START
INIT
DELAY
PORT INIT
ADC INIT
CONVERT
AD->VTEMP
VTEMP <
40 ?
S
CONTROL =
ON
VTEMP >
50 ?
S
CONTROL =
OFF
N

Example

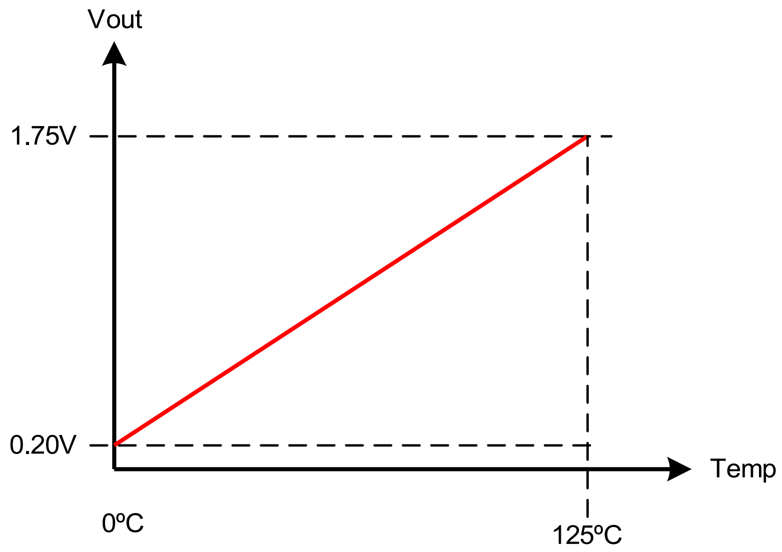
- What is the voltage at 40 y 50ºC ?

$$y = \left[\frac{y2 - y1}{x2 - x1} \right] x + b$$

$$Vout = \left[\frac{1.75 - 0.2}{125 - 0} \right] Temp + b$$

$$Vout = 0.0124 \times Temp + b$$

$$Vout = 0.0124 \times Temp + 0.2$$

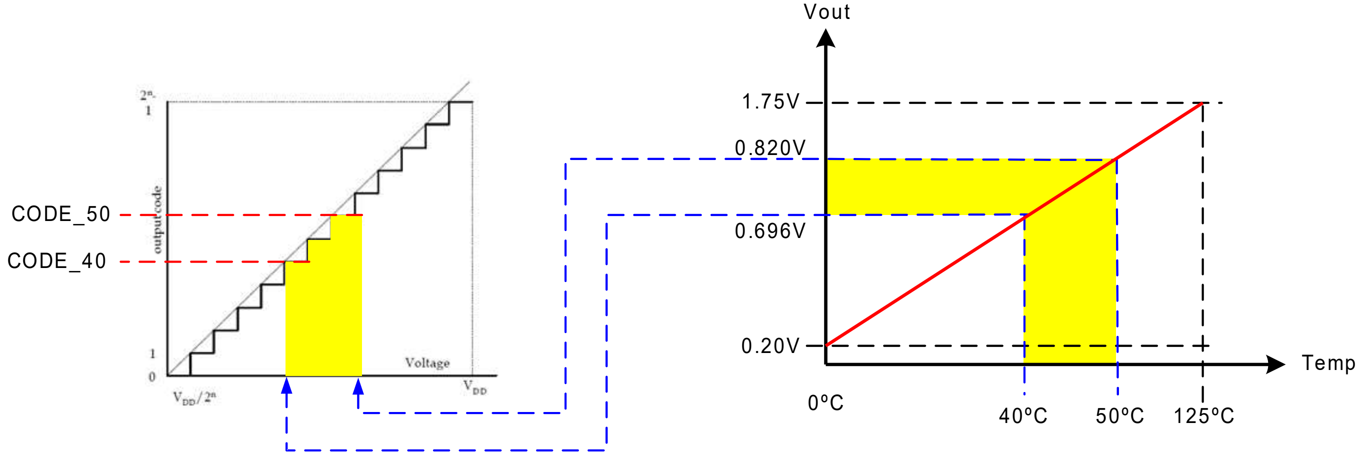


$$Vout(40^{\circ} C) = 0.0124 \times 40 + 0.2 = 0.696V$$

$$Vout(50^{\circ} C) = 0.0124 \times 50 + 0.2 = 0.820V$$

Example

- What numeric code will the voltage values have if the VREF+ is 1.8V ?



$$CODE = \frac{(Vin - VREF^-) \times (2^n - 1)}{(VREF^+ - VREF^-)} = \frac{Vin \times 1023}{1.8} = Vin \times 568.33$$

$$CODE\_40 = 0.696 \times 568.33 = 395.56 \approx 0x18C$$

$$CODE\_50 = 0.8204 \times 568.33 = 466.26 \approx 0x1D2$$

Example

- We will use the following asuptions
  - Conversion clock Fosc/64
  - Result will be right justified
  - Use analog input AN0
  - The acquisition time 20TAD
  - Use Vref+ as external reference
  - Use internal Vref- = 0V
  - Do not use interrupts

```
#include<p18f45k22.h>       //Contiene las definiciones para el procesador especifico
void my_delay(int);                 //Mi rutina general de delay

#define CONTROL  PORTDbits.RD0      //Definimos salida de control
#define CODE_40  396                //Codigo que corresponde a 40°C
#define CODE_50  466                //Codigo que corresponde a 50°C

main(){
unsigned int resultado_adc;         //Almacena resultado
//Inicializamos AN0 (RA0) como entrada y analogico
TRISAbits.RA0 = 1;          //Entrada
ANSELAbits.ANSA0 = 1;       //Analogica
//Inicializamos VREF+ (RA3) como entrada y analogico
TRISAbits.RA3 = 1;          //Entrada
ANSELAbits.ANSA3 = 1;       //Analogica
//Inicializamos RD0 como salida
CONTROL = 0;                //Para ponerlo en estado conocido
TRISDbits.RD0 = 0;          //Salida
//Esperamos un tiempo para que sistema se estabilice (recomendado)
my_delay(200);              //Esperamos 100msec
//Configuracion del convertidor (con referncia externa)
ADCON0 = 0b00000001;    //Seleccionamos AN0 y encendemos ADC
ADCON1 = 0b00000100;    //Vref+ (Externa), Vref-(GND)
ADCON2 = 0b10111110;    //Just derecha, 20TAD, FOSC/4
while(1){
    ADCON0bits.DONE = 1;         //Inicia la conversión
    while(ADCON0bits.DONE ==1); //Espera el fin de conversión
    resultado_adc = (((unsigned int)ADRESH)<<8)|(ADRESL);
    if(resultado_adc < CODE_40) CONTROL = 1;    //Encender
    if(resultado_adc > CODE_50) CONTROL = 0;    //Apagar
}//Del while(1)
} //de main() TEMA_10_PIC_2.C
```