

# **Detector tensión alterna con un PIC**

**qlfecv**  
**José Ant<sup>o</sup> Soler**

[qlfecv@ya.com](mailto:qlfecv@ya.com)

## **Resumen**

Este proyecto consiste en la detección de tensión alterna con un PIC de la familia 12F. Se realizara una explicación de la técnica usada para medir la tensión.

Sus características principales son:

- Tensión entrada 0-650Vpp
- Controlada por microprocesador.
- Señalización con 2 LEDs
- Auto alimentada con la tensión a medir

## 1. Introducción

El mayor problema que presenta este proyecto es como conseguir que al PIC le llegué una tensión proporcional a la tensión a medir pero que su rango este entre 0 y 5 V.

Para ello haremos una breve explicación de la técnica usada.

### 1.1. Circuito reductor tensión

Para el cálculo del reductor utilizaremos las leyes de Kirchhoff y el circuito básico es el que se muestra en la figura 1.

EL circuito consta de 3 resistencias conectadas de la siguiente manera:

- R1 entre Vin y Vout
- R2 entre Vr y Vout
- R3 entre Vout y masa

Siendo Vin la tensión que deseamos medir, Vout la tensión que se aplicara al PIC y Vr la tensión de referencia para la conversión del PIC.

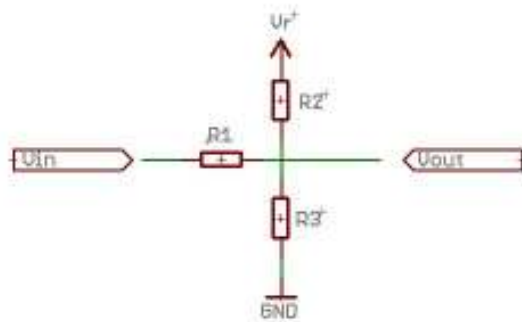


Figura 1. Circuito básico

### 1.2 Ecuaciones aplicadas

Partimos del principio de que la tensión Vout no podrá sobrepasar a Vr, por lo que cuando Vin sea máxima implicara que Vout sea Vr.

Con este razonamiento y aplicando Kirchhoff nos dan las siguientes formulas:

$$\frac{V_r}{R_3} = \frac{V_{in}}{R_1 + R_3} = \frac{V_{in} - V_r}{R_1}$$

Pero tenemos 2 incógnitas a despejar con una sola ecuación, pero podemos fijar una de las incógnitas R3 con un valor fijo de 5KΩ, por ejemplo (más adelante se explica como ajustar ese valor).

De esta forma podemos poner las ecuaciones anteriores de la siguiente manera:

$$R_1 = R_3 \frac{V_{in} - V_r}{V_r}$$

Pero en el circuito hay 3 resistencias y solo conocemos el valor de 2, por lo que en una primera aproximación haremos:

$$R_2 = R_3$$

Con estas formulas podemos pasar a calcular ya nuestro proyecto.

## 2. Cálculo del proyecto

Aplicando las formulas anteriores obtenemos:

$$R_2 = R_3 = 5K\Omega$$

$$R_1 = 5K\Omega \frac{650V - 5V}{5V} = 645K\Omega$$

Me imagino que ahora es cuando nos entra el pánico y no nos atrevemos, ni borrachos, a meterle al PIC 650 V, bueno por lo menos a mi me paso, pero antes comprobaremos lo que hemos calculado es lo que en la practica dará.

Para ello usaremos la simulación del proteus que podemos ver en la figura 2.

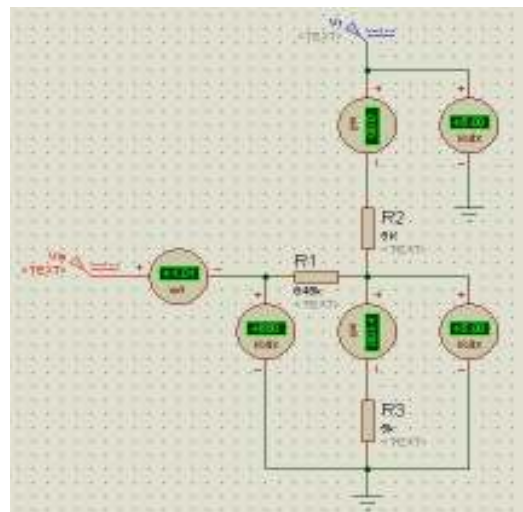
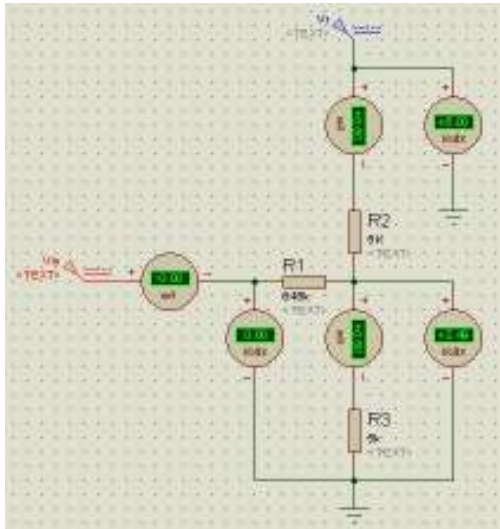
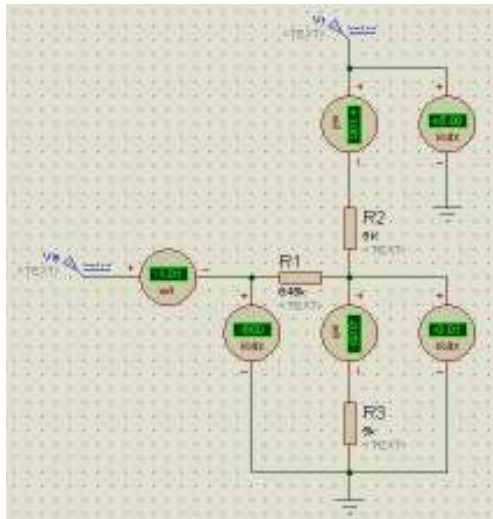


Figura 2. Circuito simulado con Vin=Vmax

Hemos verificado el funcionamiento con la tensión máxima y ahora verificaremos con los otros dos puntos de interés: V=0 y V=-Vin.



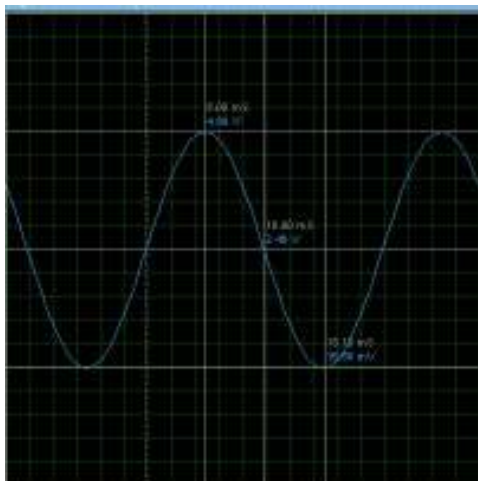
**Figura 3** Circuito simulado con  $V_{in}=0$



**Figura 4** Circuito simulado con  $V_{in}=-V_{max}$

Como podemos observar en todos los casos se cumple la teoría.

Ahora miraremos como se comportamiento con una onda alterna.



**Figura 5** Captura de  $V_{out}$

Como podemos comprobar la teoría se parece a la simulación.

## 2.1 Precisión de la medida

Para este proyecto usaremos el PIC12F683 que dispone de conversores de 10 bits de resolución o lo que es lo mismo tenemos un rango de 0 a 1023.

Con estos datos podemos estimar la resolución con la siguiente formula:

- Lectura de 1023 equivale a 650 V
- Lectura de 512 equivale a 0 V
- Lectura de 0 equivale a -650 V

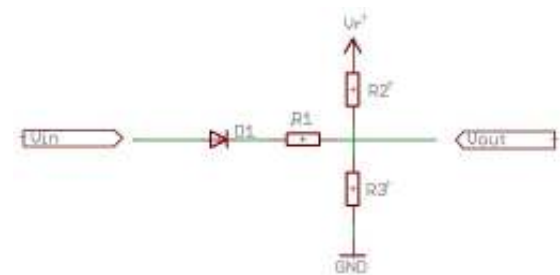
Lo que nos da una resolución de:

$$\frac{1023bits}{1300V_{pp}} = 0.8bit/V$$

Teniendo en cuenta el los valores que nos movemos no esta nada mal esa resolución, pero aplicando una técnica sencilla podemos aumentarla.

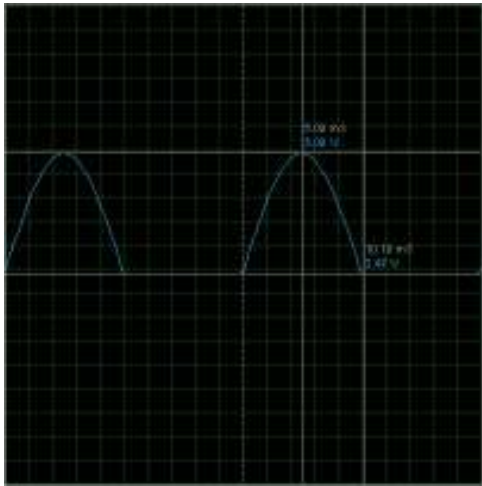
## 2.2 Mejora de la precisión de la medida

Como por definición la onda sinusoidal es simétrica podemos rectificarla añadiendo simplemente un diodo a nuestro circuito como se observa en la figura 6.



**Figura 6**

Con este circuito tendremos una forma de onda que se muestra en la figura 7.



**Figura 7 Onda rectificada**

Con esta modificación no hemos conseguido aumentar la precisión de nuestro sistema, todavía.

Para ello partiremos de la formula que relaciona  $V_{out}$ ,  $R_2$  y  $R_3$ , para  $V_{in}=0V$ .

$$V_{out} = V_r \frac{R_3}{R_2 + R_3}$$

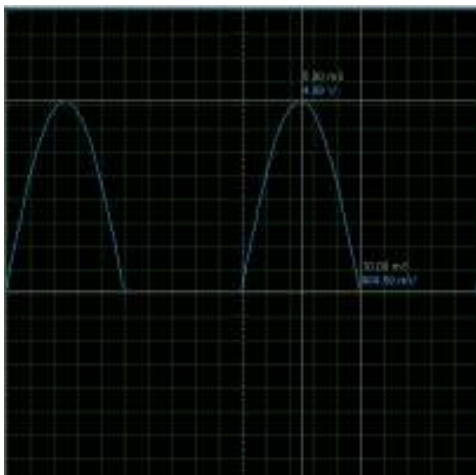
Con los valores actuales de  $R_2$  y  $R_3$  conseguimos valores entre 0 y 5V para el rango de medida, pero al modificar el rango podemos modificar esta relación, de forma que cuando tengamos una  $V_{in}=0V$  obtengamos  $V_{out}=1V$ .  
Simplemente haciendo:

$$R_2 = 4R_3$$

$$V_{out} = V_r \frac{R_3}{R_2 + R_3} = V_r \frac{R_3}{4R_3 + R_3} = V_r \frac{R_3}{5R_3}$$

$$V_{out} = \frac{V_r}{5}$$

Con estos valores volvemos a la simulación y obtenemos los siguientes datos:



**Figura 8 Onda rectificada con  $R_2$  variada**

Ahora la precisión ha mejorado siguiendo la siguiente relación:

- Lectura de 1023 equivale a 650 V
- Lectura de 204 equivale a 0 V

Lo que nos da una resolución de :

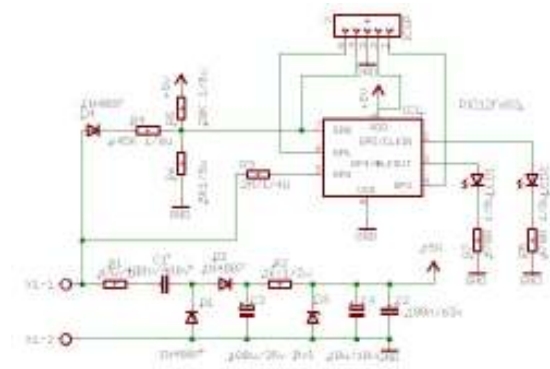
$$\frac{1023 - 204 \text{ bits}}{650 V_p} = 1.26 \text{ bit/V}$$

Podríamos aumentar la precisión más aumentando la relación  $R_2$ - $R_3$  pero para las magnitudes que queremos es más que suficiente.

Ahora ya podemos pasar a la realización práctica.

### 3. Realización practica

Con todos los cálculos realizados montamos el siguiente circuito:



**Figura 9 Esquema eléctrico**

Como se puede observar entramos directamente la tensión al pin GP1 y al pin GP0 a través de nuestra cadena reductora.

### 3.1 Metodología

La onda alterna sinusoidal es de una frecuencia de 50Hz lo que implica que tiene un periodo de 20 ms.

Al haber rectificado la señal nos centraremos en la semionda positiva y el proceso será:

- Detección del paso por cero generando una interrupción.
- Esperar que se alcance el valor de pico máximo (5ms).
- Medir el valor

La detección del paso por cero se basa en la teoría de la AN521 de Microchip.

La rutina de la interrupción será:

```
#int_EXT
void EXT_isr()
{
    // desactivamos la interrupción de paso por cero
    disable_interrupts(INT_EXT);
    // Contador de veces que ejecutamos el timer
    ContadorTimer=0;
    // ponemos el offset al timer para ajustar el tiempo
    set_timer0(DELAY);
    // activamos la interrupción del timer
    enable_interrupts(INT_TIMER0);
}
```

Esta interrupción lanza la interrupción del timer para esperar los 5 ms.

```
#INT_TIMER0
void TIMER_isr(){
    // desactivamos la interrupción, este periodo de tiempo
    // se desprecia
    disable_interrupts(INT_TIMER0);
    // si estamos en el ultimo conteo ajustamos la rutina
    // de tiempo para coincidir con los 5 ms que se alcanza
    // el valor máximo
    If (ContadorTimer==4)
        set_timer0(100);
    else
        set_timer0(DELAY);
    // Contador de veces que hemos ejecutado la rutina
    ContadorTimer++;
    // Verifica si es el momento de medir la tensión máxima
    If (ContadorTimer==6){
        // leemos el valor
        valor=read_adc();
        // restamos el offset que es de 1 volt equivales a
        // 1024/5 = 204, corregimos el error por aquí
        // quitando 3 que son unos 2 volts
        valor-=207;
        // calculamos la constante de conversión que es el
        // valor leído/valor máximo, siendo el valor máximo
        // 1023-el offset sin corrección
        factor=(valor/819.0);
        // damos forma al valor leído, siendo el fondo
        // escala 650v de pico a pico
        tension=(int16)(factor*650.0);
        // calculamos el valor eficaz
        eficaz=(int16)(factor*460.0);
        eficaz-=25;
        // arrancamos la interrupción de paso por cero
        // verificamos el valor con los límites
        if (eficaz>VEMIN && eficaz<VEMAX){
            ledok=1;
            lederror=0;
        }else{
            ledok=0;
            lederror=1;
        }
        // activamos la interrupción de paso por cero
        enable_interrupts(INT_EXT);
    }else
        // activamos la interrupción del timer
        enable_interrupts(INT_TIMER0);
}
```

Y el bucle principal será:

```
void main(){
    // define velocidad del reloj
    setup_oscillator(OSC_8MHZ);
    // inicializa variables
    ledok=0;
    lederror=0;
    // Configuración puertos salida
    // GP0/AN0 => medida de la señal
    // GP1    => Tx rs232
    // GP2    => interrupción paso por cero
    // GP3    => reset
    // GP4    => LED OK
    // GP5    => LED ERROR
    // 1 = input 0 = output
    SET_TRIS_A(0b000101);
    // GP0 como entrada analógica con referencia Vdd-Vss
    setup_adc_ports(sAN0|VSS_VDD);
    // Muestreo del ADC
    setup_adc(ADC_CLOCK_DIV_2);
    // seleccionamos el canal 0 del adc
    set_adc_channel(0);
    // desactivamos las interrupciones que usaremos
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER0);
    // activamos la global
    enable_interrupts(GLOBAL);
    // configuramos el timer 0 para tener un overflow cada
    // 256 us a 4Mhz es decir conteo cada lus = Fosc/4
    setup_timer_0(RTCC_INTERNAL|RTCC_DIV_4);
    // Define flanco de subida (Para la activación de la
    // interrupción)
    ext_int_edge(INT_EXT,L_TO_H);
    // activa la interrupción de paso por cero
    enable_interrupts(INT_EXT);
    // bucle principal
    while (1){
        if (ledok){
            OUTPUT_LOW(LED_ROJO);
            OUTPUT_HIGH(LED_VERDE);
        }
        if (lederror){
            OUTPUT_LOW(LED_VERDE);
            OUTPUT_HIGH(LED_ROJO);
        }
    }
}
```

La definición de variables quedara:

```
#include <12F683.h>

#device adc=10

#FUSES NOWDT           //No Watch Dog Timer
#FUSES INTRC_IO        //Internal RC Osc, no CLKOUT
#FUSES NOCPD           //No EE protection
#FUSES NOPROTECT       //Code not protected from reading
#FUSES NOMCLR          //Master Clear pin enabled
#FUSES NOPUT           //Power Up Timer
#FUSES NOBROWNOUT      //Reset when brownout detected

#define DELAY    6    // avance en la interrupción del timer
#define VEMIN    230-23    // valores de referencia
#define VEMAX    230+23
#define LED_VERDE    PIN_A4
#define LED_ROJO     PIN_A5

/* VARIABLES */

int1 ledok;           // leds
int1 lederror;
//veces que se ejecuta la interrupción del timer
int8 ContadorTimer;
int16 valor, tension, eficaz;    // valores analógicos
float factor;           // factor de conversión
```

## Referencias

Doug Cox. *Microchip AN521 Interfacing to AC Power Lines*.

Nocturno 2.007. *Minifuentes sin transformador*  
[Minifuentes sin transformador](#)

M.E. Van Valkenburg 1.980. *Análisis de Redes*.  
Editorial LIMUSA.

MrCal. *Foro Microchip, A/D Converter*.