Instituto Politécnico Nacional

Escuela Superior de Cómputo

UA. Instrumentación

‘Proyecto Final’

Miembros del Equipo 7:

Márquez León Jorge Luis - 2013090484

Montoya Uribe Miguel Ángel - 2019630229

Rivera Pérez Ricardo - 2019630289

Grupo:

3CV3

Profesora:

Rosario Rocha Bernabé

Fecha de realización:

20/01/2021

Contenido

Introducción 3
Sistema virtual de medición 3
Comunicación serial 4
Estándar RS232 4
Interfaz de usuario 4
Interfaz de hardware 4
Interfaz de software 4
Interfaz de software-hardware 4
Variables físicas del sistema 5
Sensor de Temperatura 6
Sensor AD592 6
Características 6
Simulaciones 7
Sensor de Humedad 8
Sensor HCH-1000 8
Características 8
Simulaciones 9
Sensor ultrasónico 10
Sensor SRF04 10
Características 10
Simulaciones 11
Diseño de Interfaz Gráfica 11
Código del Microcontrolador 17
Código de la Interfaz Gráfica de Usuario 19
Conclusiones 22
Referencias 23

Introducción

El presente trabajo va referido para generar un Invernadero Automatizado, con el fin de aplicar los conocimientos adquiridos en la unidad de aprendizaje de Instrumentación, considerando el uso de sensores analógicos, su posterior acondicionamiento, y digitalización mediante un microcontrolador, enviando los datos obtenidos hacia una GUI (Interfaz Gráfica de Usuario).

El tema de este proyecto radica en el manejo racional de los factores climáticos de forma conjunta para el cultivo de tomates.

Sistema virtual de medición

El concepto de instrumentación virtual nace a partir del uso del computador personal como "instrumento" de medición de señales tales como temperatura, presión, caudal, entre otras. Es decir, el PC comienza a ser utilizado para realizar mediciones de fenómenos físicos representados en señales de corriente y/o voltaje normalmente con una gran precisión y número de muestras por segundo.

Sin embargo, el concepto de "instrumentación virtual" va más allá de la simple medición de corriente o voltaje. También involucra el procesamiento, análisis, almacenamiento, distribución y despliegue de los datos e información relacionados con la medición de una o varias señales específicas. Con éstas, mediante software que permitan la implementación de algoritmos de control, es factible integrar y controlar complicados procesos. Es decir, el instrumento virtual no se conforma con la adquisición de la señal, sino que también involucra la interfaz hombre-máquina, las funciones de análisis y procesamiento de señales.

En sus orígenes, estos equipos, compuestos de una tarjeta de adquisición de datos con acondicionamiento de señales (PCMCIA, ISA, XT, PCI, etc.) y el software apropiado estaban orientados a laboratorios, donde sus prestaciones eran muy requeridas por la gran precisión y capacidad de adecuar sus capacidades y cálculos acorde al proceso que se estaba analizando. Con el tiempo, éstas cada vez más robustas soluciones y prestaciones de PC industriales permiten encontrar aplicaciones en la industria, en sistemas de robótica y otras tantas aplicaciones donde la capacidad de cálculo hace que una instrumentación tradicional no sea capaz de responder al requerimiento del proceso.

La flexibilidad, el bajo costo de mantenimiento, la reusabilidad, la personalización de cada instrumento, la rápida incorporación de nuevas tecnologías, el bajo costo por función y por canal son algunos de los beneficios que ofrece la instrumentación virtual.

La instrumentación virtual puede también ser implementada en equipos móviles (laptops), equipos distribuidos en campo (RS-485), equipos a distancia (conectados vía radio, Internet, etc.) o equipos industriales (NEMA 4X, etc.). Existe una tarjeta de adquisición de datos para casi cualquier bus o canal de comunicación en PC (ISA, PCI, USB, serial RS-232/485, paralelo EPP, PCMCIA, CompactPCI, PCI, etc.) y un driver para casi cualquier sistema operativo (WIN 3.1/95/NT, DOS, Unix, MAC OS, etc.).

Comunicación serial

Consiste en el envío de un bit de información de forma secuencial a un ritmo acordado entre el emisor y el receptor. Se pueden emplear varios tipos de comunicación serie los más conocidos se designan como Simplex y Full Dúplex.

El método Simplex implementa la transmisión de datos unidireccional. En este esquema, solo el origen o el destino están activos en un momento dado. Si la fuente está enviando datos, el receptor no tiene más remedio que aceptar la transmisión. El modo Simplex se usa para transmitir señales de televisión o radio.

Modo Full Dúplex es la forma de comunicación serie más utilizada en el mundo. El origen y el destino están activos y pueden enviar y recibir datos simultáneamente. El teléfono inteligente es un excelente ejemplo del modo full dúplex en acción.

La comunicación serial ha seguido los estándares de comunicación RS232 y fue implementada en diferentes puertos como el “puerto serial” o el “USB”.

Estándar RS232

Es una norma o estándar mundial que rige los parámetros de uno de los modos de comunicación serial. Por medio de este protocolo se estandarizan las velocidades de transferencia de datos, la forma de control que utiliza dicha transferencia, los niveles de voltajes utilizados, el tipo de cable permitido, las distancias entre equipos, los conectores, entre otros.

A nivel de software, la configuración principal que se debe dar a una conexión a través de puertos seriales. RS-232 es básicamente la selección de la velocidad en baudios (1200, 2400, 4800, 9600.), la verificación de datos o paridad (parida par o paridad impar o sin paridad), los bits de parada luego de cada dato (1 o 2), y la cantidad de bits por dato (7 u 8) que se utiliza para cada símbolo o carácter enviado.

Interfaz de usuario

La interfaz de usuario (UI) es el conjunto de los controles y canales sensoriales mediante los cuales un usuario puede comunicarse con una máquina. Por ejemplo, en una computadora, la pantalla, el teclado y las bocinas son parte de la interfaz de usuario porque la utilidad de todas ellas es hacer que entre o salga información del equipo.

Existen 3 tipos de interfaz de usuario según su diseño y propósito:

Interfaz de hardware

Engloba todos aquellos elementos que permiten ingresar, procesar y entregar datos, como los famosos teclados y “ratones”, así como las pantallas.

Interfaz de software

Es aquella que brinda información sobre los procesos y herramientas de control, la cual puede ser observada fácilmente por el usuario en la pantalla de su dispositivo.

Interfaz de software-hardware

Dentro de la interfaz del usuario, esta es la que sirve de puente entre la máquina y las personas.

Genera un círculo virtuoso en el cual el dispositivo comprende las instrucciones del individuo y, a su vez, este entiende la información del aparato, a través de la traducción del código binario a elementos simples.

Bajo este esquema, podemos afirmar que existen interfaces de línea de comandos, gráficas de usuario y natural de usuario.

La interfaz de línea de comandos es alfanumérica, es decir, solo presenta texto, mientras que las interfaces de gráficas del usuario representan visualmente los elementos de control y medida para optimizar la comunicación con el usuario y, en general, la experiencia de este.

En cuanto a la interfaz natural de usuario, son aquellas que conectan de manera directa al hombre y la máquina.

Variables físicas del sistema

Es el grupo de elementos que sirven para medir, sobrevivir, convertir, transmitir, controlar o registrar variables de un proceso con el fin de optimizar los recursos utilizados en este. Es el conocimiento de la correcta aplicación de los equipos encaminados para apoyar al usuario en la medición, regulación, observación, transformación, ofrecer seguridad, etc., de una variable dada en un proceso productivo.

Un sistema de instrumentación es una estructura compleja que agrupa un conjunto de instrumentos, un dispositivo o sistema en el que se mide, unas conexiones entre estos elementos y, por último, y no menos importante, unos programas que se encargan de automatizar el proceso y de garantizar la repetibilidad de las medidas.

El elemento clave fundamental de un sistema de instrumentación, es el elemento sensor. La función del sensor es percibir y convertir la entrada (variable física) percibida por el sensor, en una variable de la señal de salida.

El sensor es un elemento físico que emplea algún fenómeno natural por medio del cual censar la variable a ser medida. El transductor, convierte esta información censada en una señal detectable, la cual puede ser eléctrica, mecánica, óptica, u otra. El objetivo es convertir la información censada en una forma que pueda ser fácilmente cuantificada.

Las variables para medir o controlar pueden ser:

- Caudal
- Presión
- Temperatura
- Nivel
- Velocidad
- Peso
- Humedad
- Punto de rocío
- pH
- Conductividad Eléctrica
- Redox

Sensor de Temperatura

Sensor AD592

El AD592 es un transductor de temperatura de circuito integrado monolítico de dos terminales que proporciona una corriente de salida proporcional a la temperatura absoluta. Para una amplia gama de suministros voltajes el transductor actúa como una temperatura de alta impedancia fuente de corriente dependiente de 1µA/K. El diseño mejorado y el recorte de obleas láser de las resistencias de película delgada del IC permiten que el AD592 alcance niveles de precisión absoluta y errores de no linealidad que antes eran inalcanzables a un precio comparable.

El AD592 puede emplearse en aplicaciones entre –25 ° C y + 105 ° C donde se utilizan actualmente sensores de temperatura convencionales (es decir, termistor, RTD, termopar, diodo).

El bajo costo inherente de un circuito integrado monolítico en un paquete de plástico, combinado con un bajo recuento total de piezas en cualquier aplicación, hacen del AD592 el transductor de temperatura más rentable disponible actualmente. Con el AD592 no se requieren circuitos de linealización costosos, referencias de voltaje de precisión, componentes de puente, circuitos de medición de resistencia y compensación de unión fría.

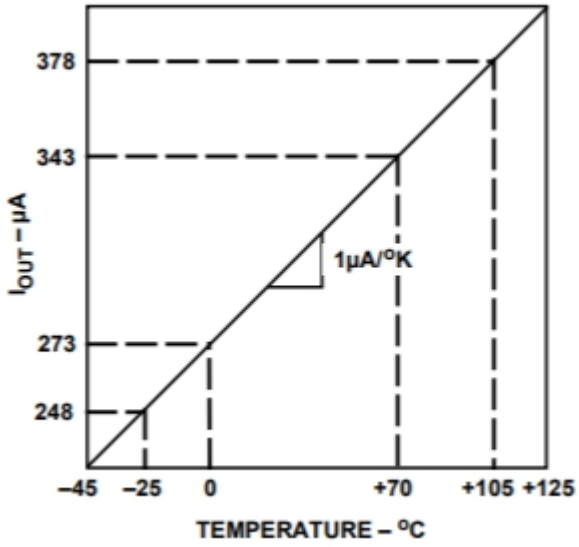

Ilustración 1 Comportamiento del AD592

Características

1.- Con un solo suministro (4 V a 30 V), el AD592 ofrece una precisión de medición de temperatura de 0.5 ° C.

2.- Un amplio rango de temperatura de funcionamiento (–25 ° C a + 105 ° C) y una salida altamente lineal hacen del AD592 un sustituto ideal para tecnologías de sensores más antiguas y limitadas (es decir, termistores, RTD, diodos, termopares).

3.- El AD592 es eléctricamente resistente; las irregularidades y variaciones de suministro o voltajes inversos de hasta 20 V no dañarán el dispositivo.

4.- Debido a que el AD592 es una fuente de corriente dependiente de la temperatura, es inmune a la captación de ruido de voltaje y las caídas de infrarrojos en los cables de señal cuando se usa de forma remota.

5.- La alta impedancia de salida del AD592 proporciona un rechazo superior a 0,5 ° C / V de la deriva y la ondulación del voltaje de suministro.

6.- El recorte de obleas con láser y las pruebas de temperatura garantizan que las unidades AD592 sean fácilmente intercambiables.

7.- La precisión inicial del sistema no se degradará significativamente con el tiempo. El AD592 ha demostrado ventajas de repetibilidad y rendimiento a largo plazo inherentes al diseño y construcción de circuitos integrados.

Simulaciones

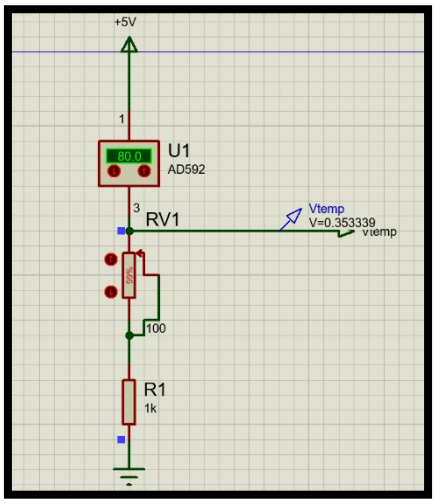

Ilustración 2. Acondicionamiento para el sensor de temperatura

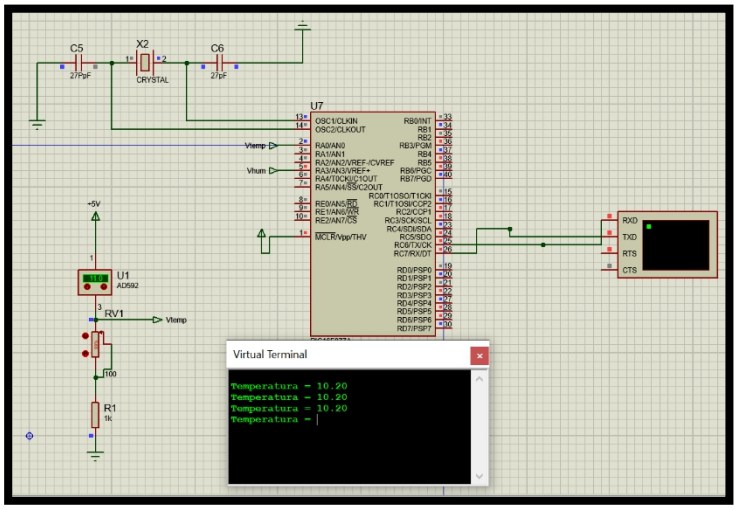

Ilustración 3. Microcontrolador con el sensor de Temperatura

Sensor de Humedad

El sensor de humedad es un aparato de lectura utilizado en espacios interiores para controlar la humedad del aire y la temperatura. Las magnitudes medidas por el sensor de humedad se transforman en una señal eléctrica normalizada, cuya intensidad suele estar comprendida entre 4 y 20 mA. Un material semiconductor es el encargado de determinar con precisión los valores de humedad y temperatura que corresponden con la señal emitida.

Sensor HCH-1000

El sensor de humedad de la serie HCH-1000 es un sensor capacitivo de polímero diseñado para medir la humedad relativa. El sensor convierte el valor de humedad en capacitancia, consta de un electrodo superior de rejilla, una capa de poliimida, y un electrodo inferior. El electrodo superior de rejilla en la parte inferior. El electrodo proporciona una sensibilidad mejorada en comparación con una estructura estándar.

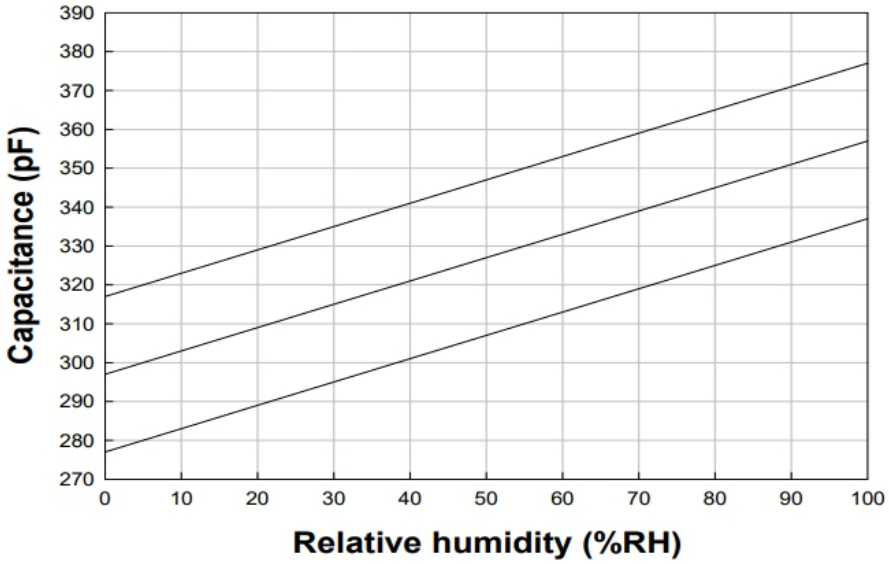

Ilustración 4. Comportamiento de HCH-1000

Características

- Ofrece una mayor resistencia contra la contaminación.
- Dependencia a la temperatura reducida.
- Fabricado a partir de semiconductores.
- Poca histéresis y gran durabilidad.
- Sensibilidad y precisión mejorados proporcionando una respuesta rápida.

Simulaciones

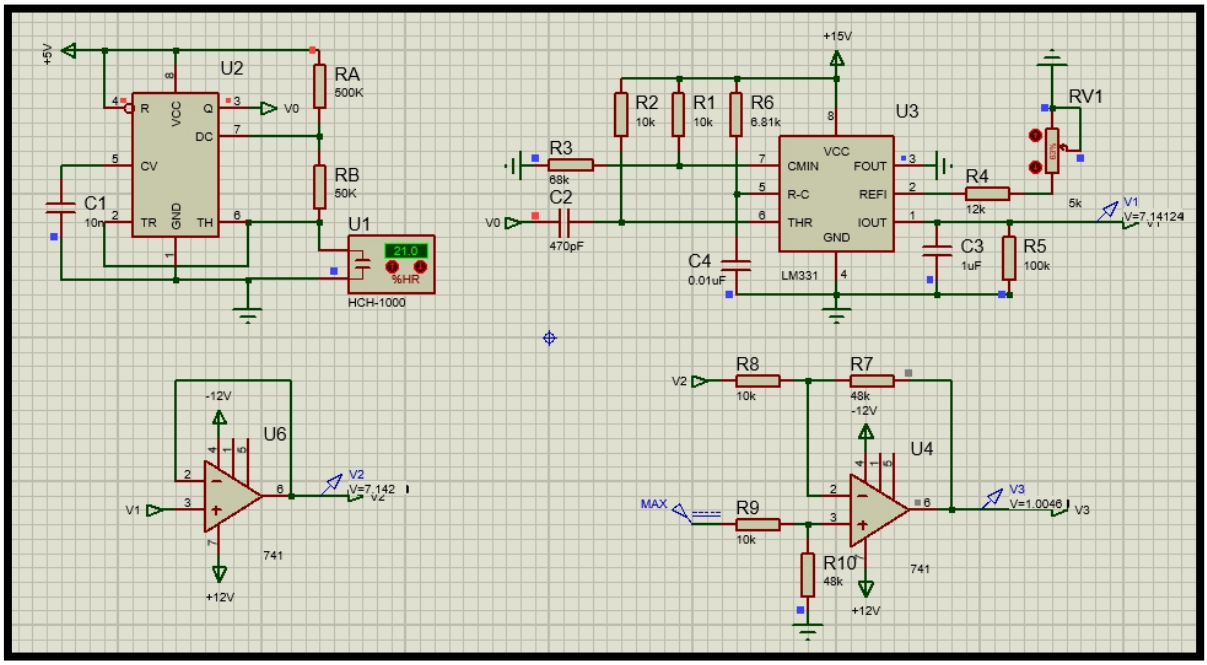

Ilustración 5. Acondicionamiento para el sensor de Humedad

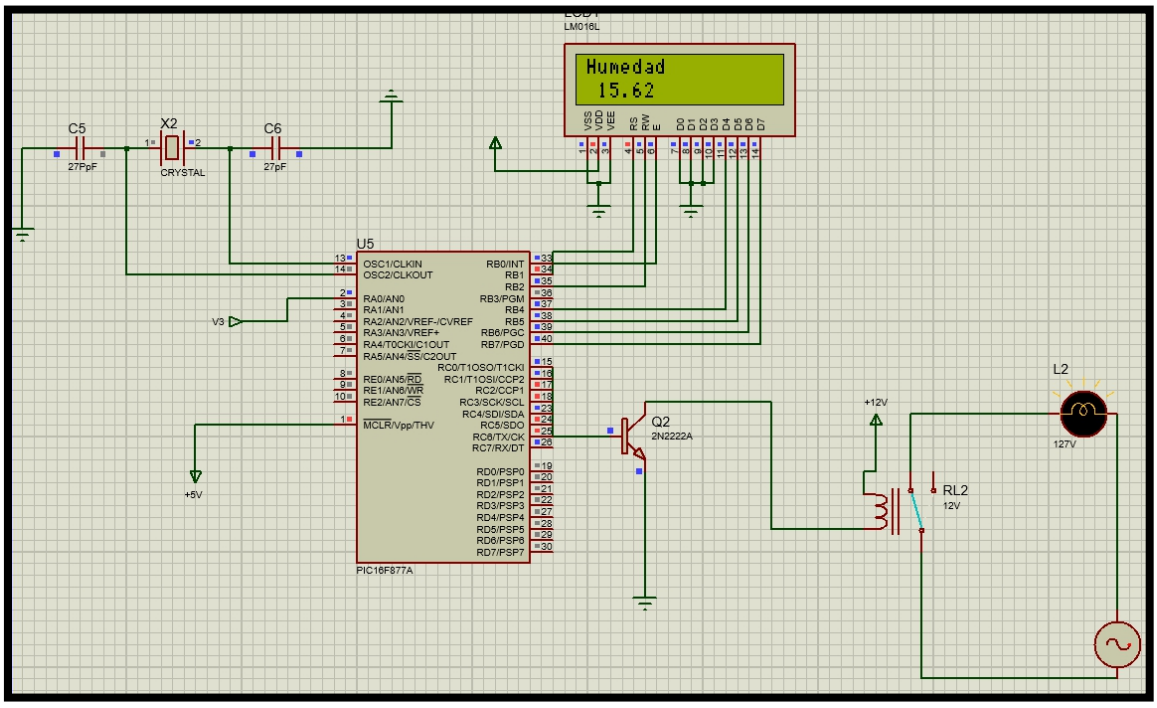

Ilustración 6. Microcontrolador para el sensor de Humedad

Sensor ultrasónico

Sensor SRF04

SRF04 es un sensor de distancias por ultrasonidos capaz de detectar objetos y calcular la distancia a la que se encuentra en un rango de 3 a 300 cm. Trabajan solamente donde hay presencia de aire (no pueden trabajar en el vacío, necesitan un medio de propagación).

El sensor SRF04 funciona emitiendo impulsos de ultrasonidos inaudibles para el oído humano. Los impulsos emitidos viajan a la velocidad del sonido hasta alcanzar un objeto, entonces el sonido es reflejado y captado de nuevo por el receptor de ultrasonidos.

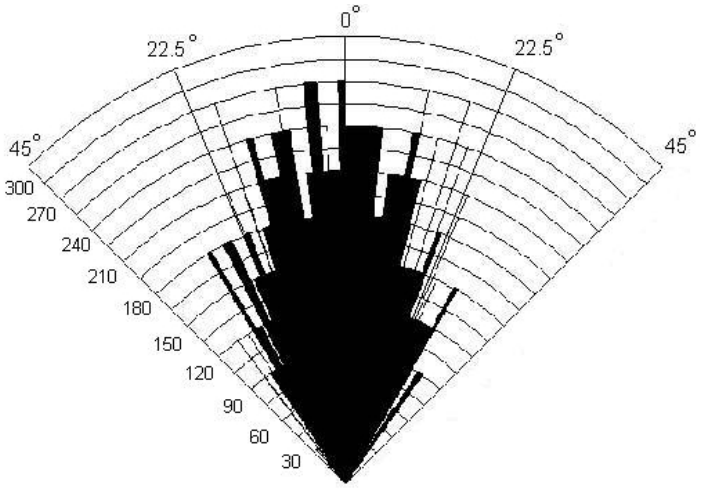

Ilustración 7 Rango efectivo del SRF04

Algunas de sus aplicaciones son:

- Navegación con robots.
- Automatización de fábricas.
- Detección de nivel del agua.

Características

- Detecta objetos en distancias entre los 3 y 300 cm.
- Cuenta con emisor de 40Khz.
- Tiene un voltaje de operación de 5V.
- Ángulo de apertura de 15°.
- Corriente de trabajo de 15 mA.

Simulaciones

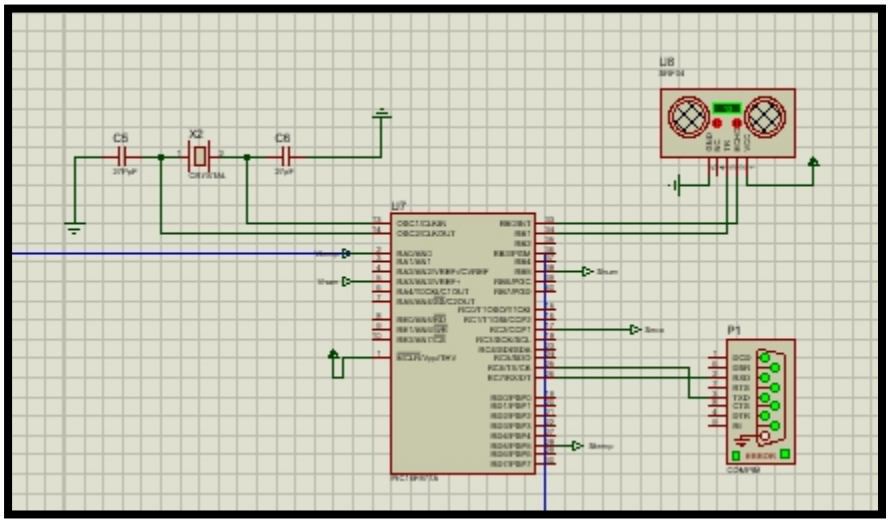

Ilustración 8 Microcontrolador para el sensor ultrasonido

Diseño de Interfaz Gráfica

El diseño de la Interfaz que se muestra al usuario se puede observar en la Ilustración 9, cabe destacar que la interfaz se encuentra desconectada/apagada para recibir señal del puerto COM del circuito.

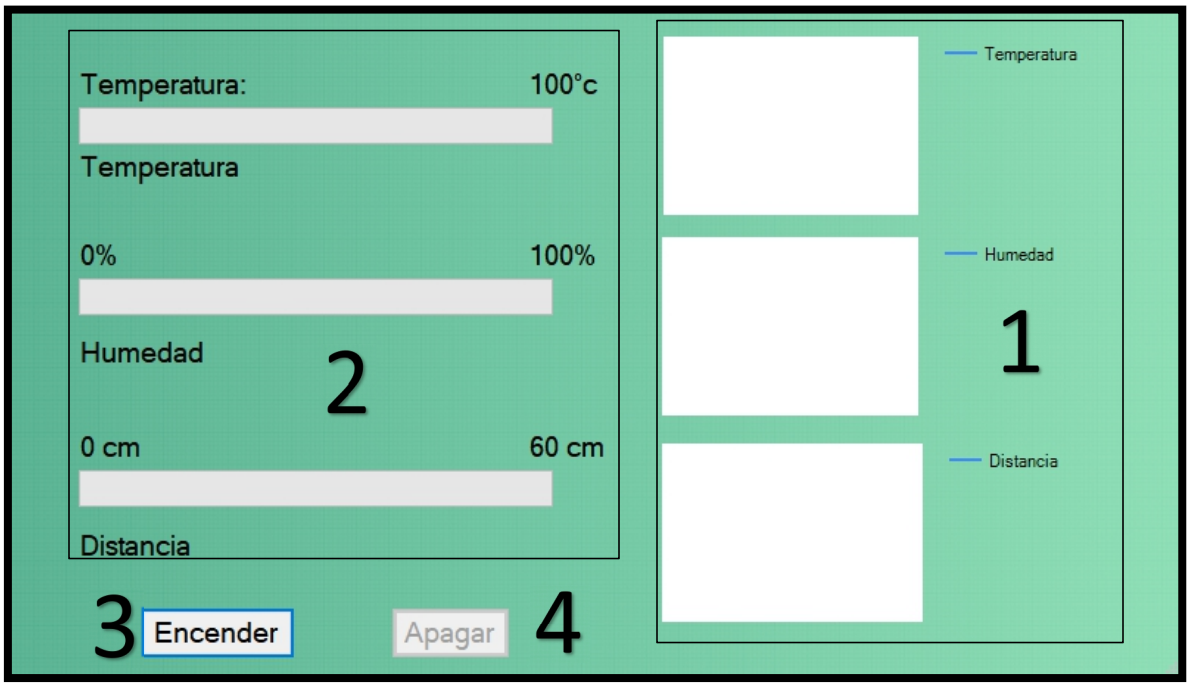

Ilustración 9. Vista de la Interfaz de Usuario, apagada

Los componentes que encontramos en la ilustración 9 son:

1. Graficas de línea para las variables de Temperatura, Humedad y Distancia.

2. Barras de cargas de progreso para las variables de Temperatura, Humedad y Distancia.
3. Botón de Encendido, para empezar a captar datos del circuito.
4. Botón de Apagado, para dejar de captar datos del circuito.

Cuando se oprime el botón de encendido, la interfaz gráfica de usuario empieza a recibir los datos desde el circuito, esta comunicación se logra a través del puerto serial del modulo COMPIM, que se conecta en el pin 26 (RX-Receptor), y el Pin 25(TX-Transmisor) del microcontrolador PIC16F877A.

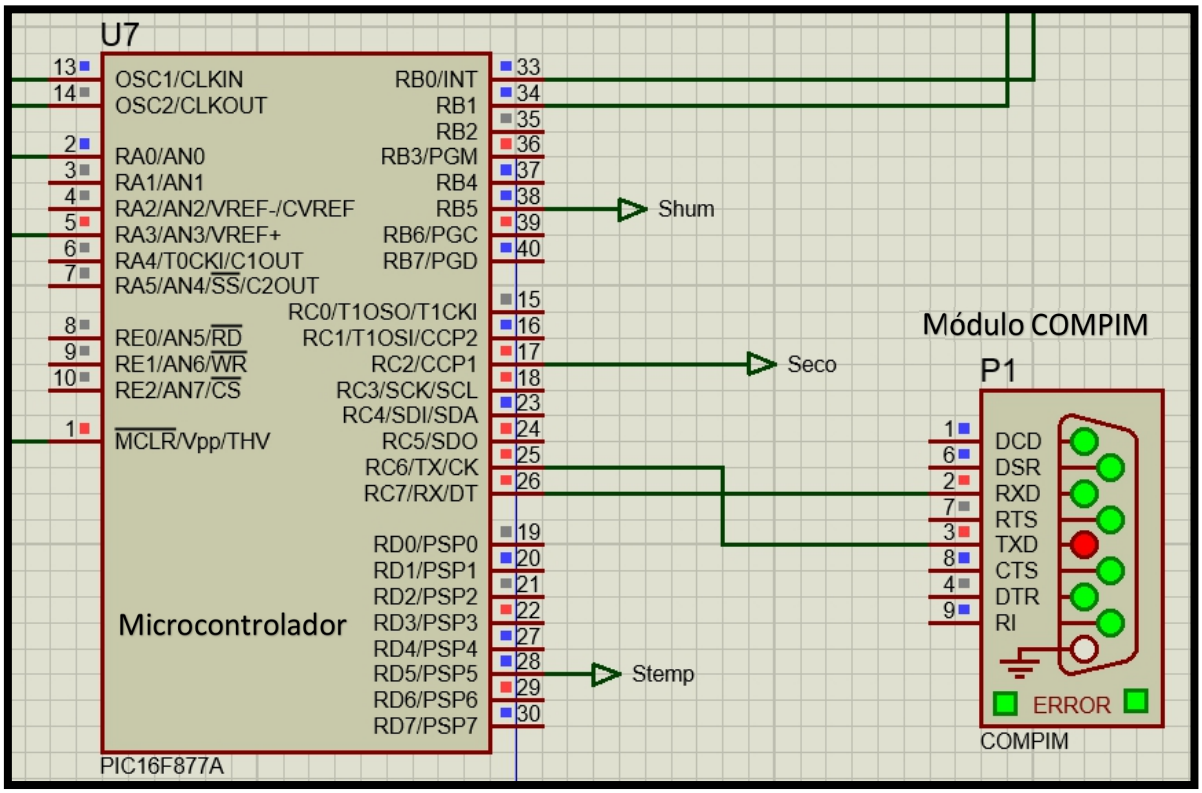

Ilustración 10.Componente COMPIM

Este adaptador se debe configurar según el protocolo RS-232 y debe concordar con lo programado en el PIC, en este caso, la línea de programación del PIC es

```
1 #use rs232 (baud=9600, parity=N, xmit=pin_c6, rcv=pin_c7, bits=8)
```

La anterior directiva al preprocesador entre sus parámetros podemos encontrar los siguientes:

- **BAUD** con este parámetro establecemos la velocidad en baudios a la que queremos que se transmitan los datos por el puerto serie, 9600 es lo normal.
- **BITS** número de bits que utilizaremos en la transmisión, el estándar establece que pueden ser 8 o 9, para la comunicación con microcontroladores con 8 son suficientes.
- **PARITY** nos permite utilizar un bit de paridad para la comprobación de errores, está opción la dejamos a No.
- **XMIT** está opción nos configura porque patilla del PIC saldrán los datos, está opción junto con la siguiente sí que la tendremos que cambiar a nuestras necesidades.

- **RCV** nos configura porque patilla del PIC se recibirán los datos. En el ejemplo, los datos se transmiten por el PIN 25 y se reciben por PIN 26

  Por lo que el componente COMPIM se configuró de la siguiente manera:

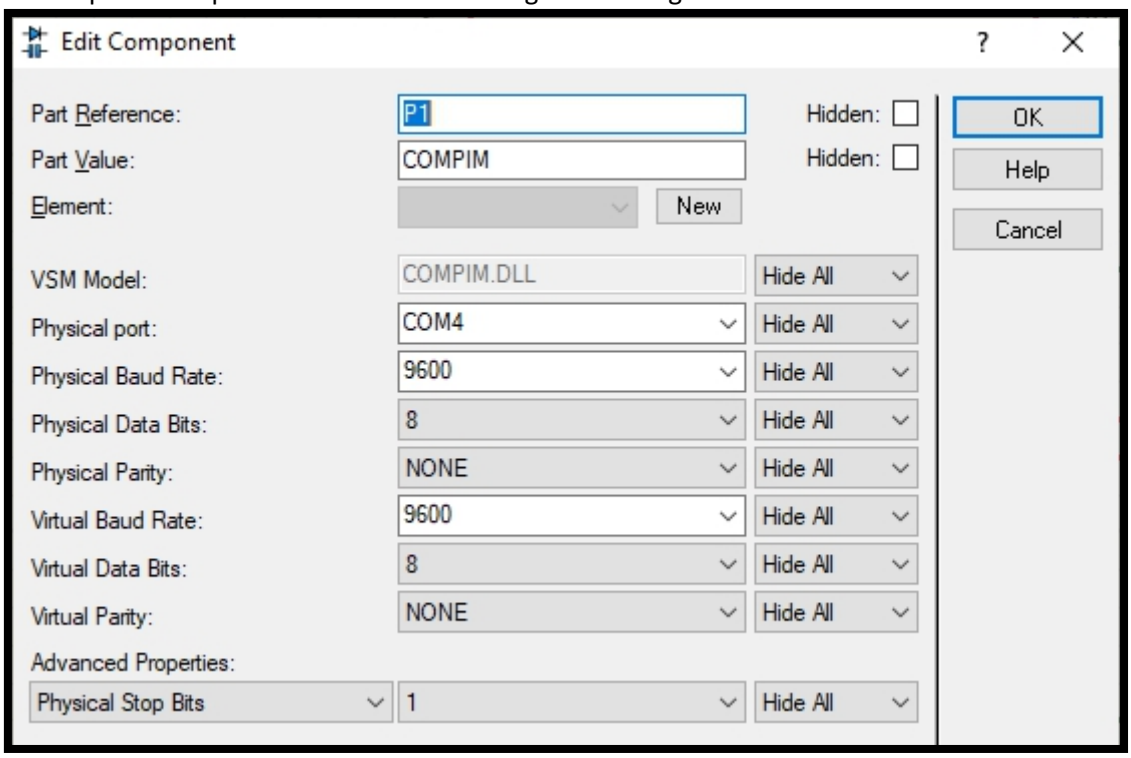

Ilustración 11. Configuración de COMPIM

En la Ilustración 12 y 13 mostramos la exitosa comunicación con el circuito a través del módulo COMPIM, en adición se muestra un cambio en el sensor de temperatura de 44 a 71 grados.

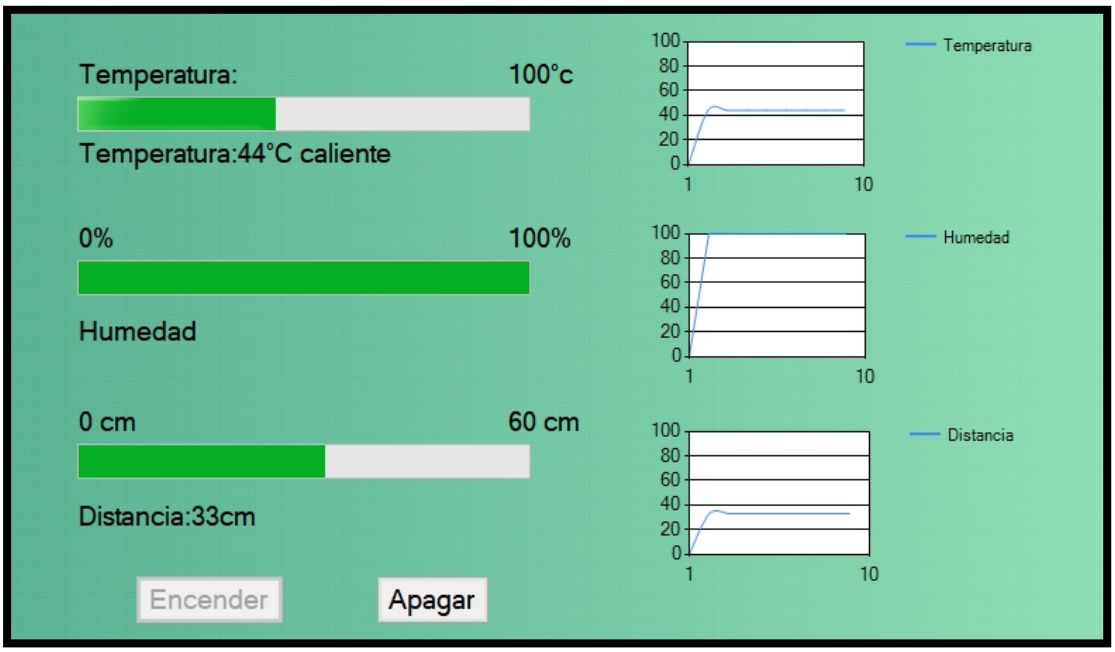

Ilustración 12. Cambio del sensor de Temperatura 44 °C a 71 °C

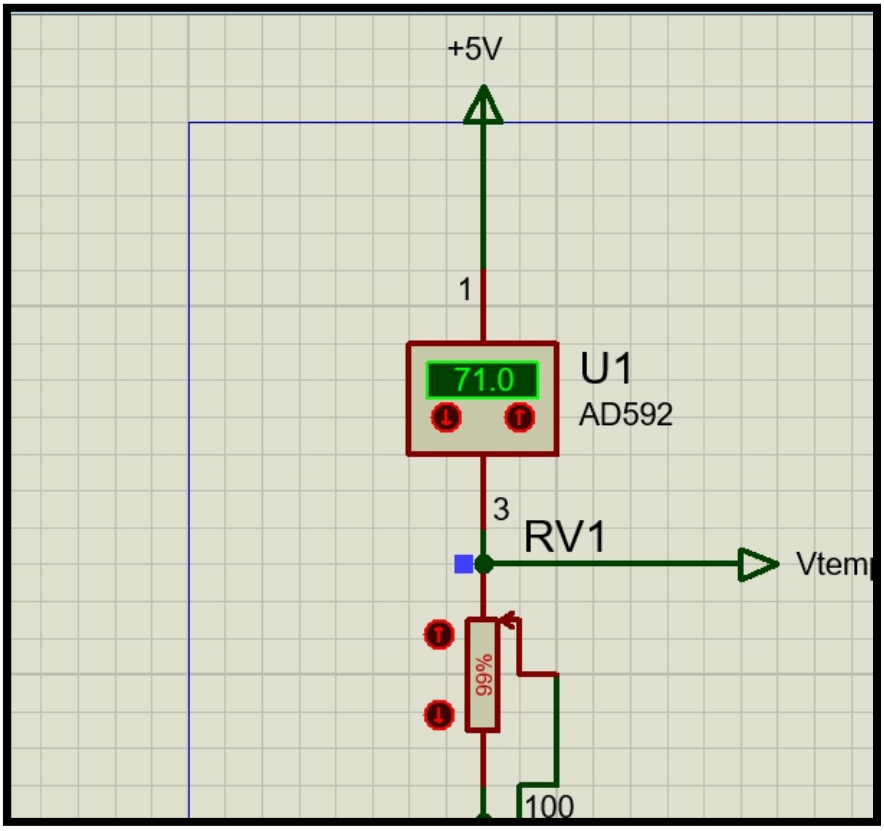

Ilustración 13. Sensor AD592 registrando una temperatura de 71°C

En la siguiente Ilustración 14 y 15, mostramos a un cambio en el sensor de humedad de 95 a 23% de humedad relativa.

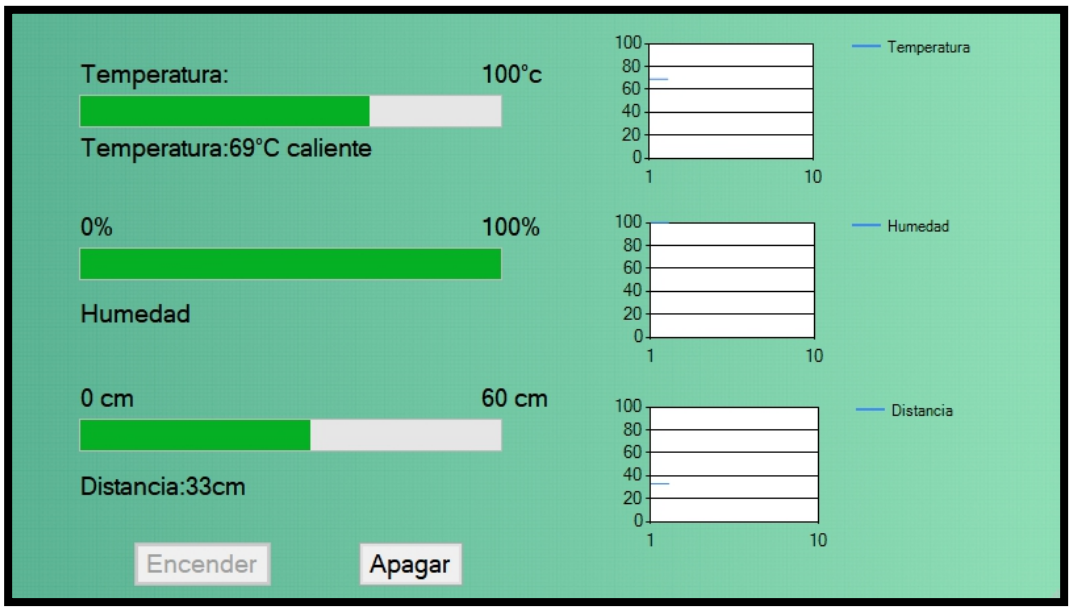

Ilustración 14. Cambio de Humedad de 95 a 23% humedad Relativa

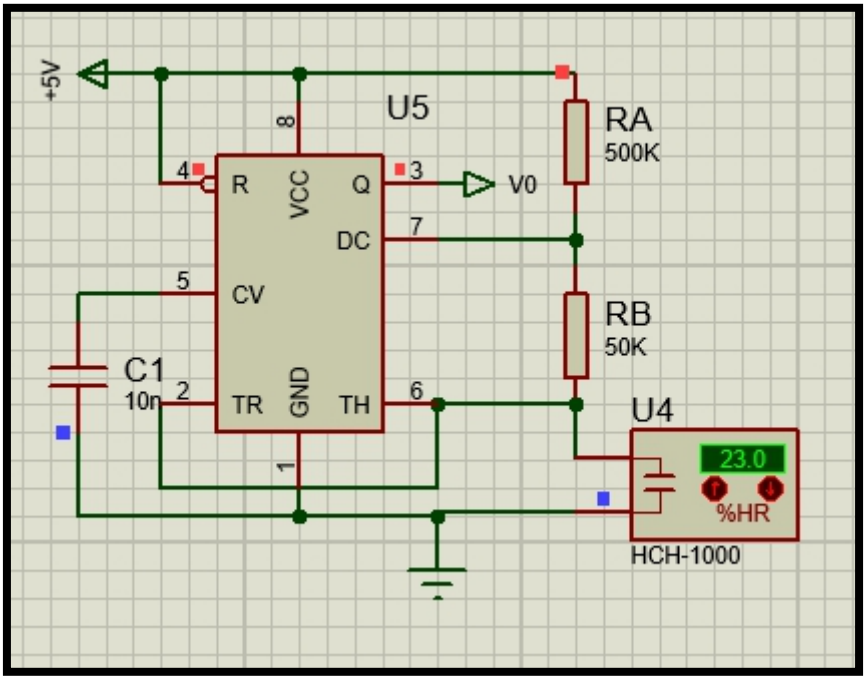

Ilustración 15. HCH-1000 Registrando una Humedad Relativa del 23 %

En la ilustración 16, 17 y 18 se presenta un cambio en el sensor de distancia de 33 cm a 55 cm

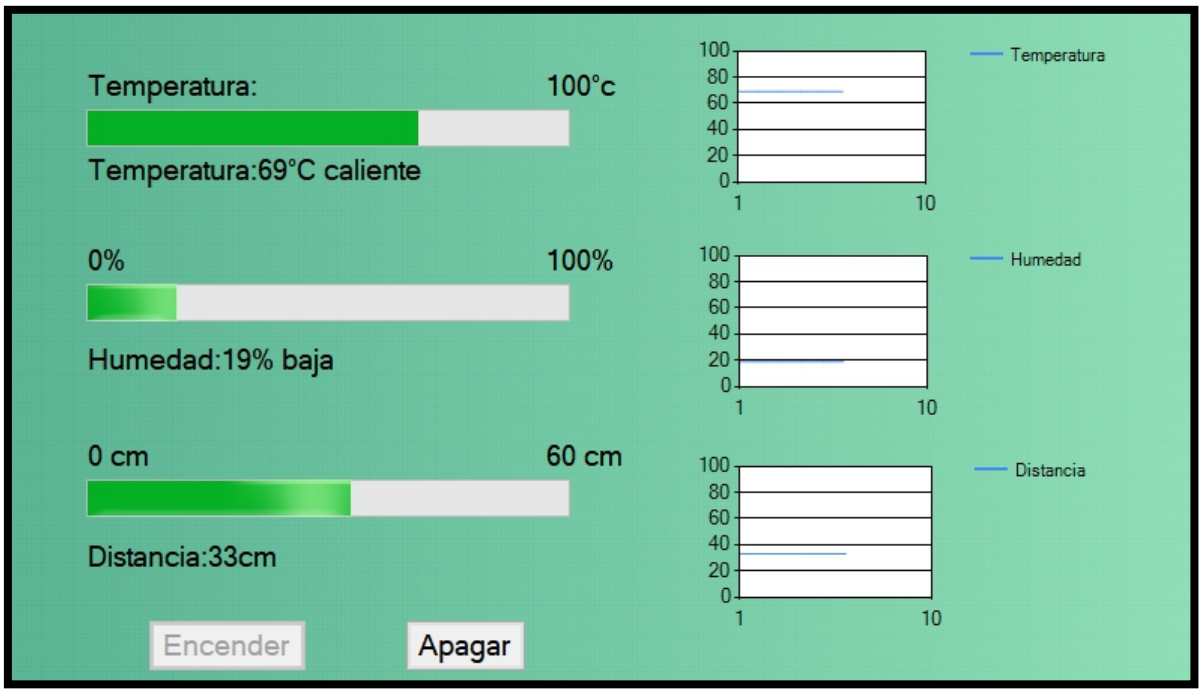

Ilustración 16. Captando una distancia de 33cm en el sensor del ultrasonido.

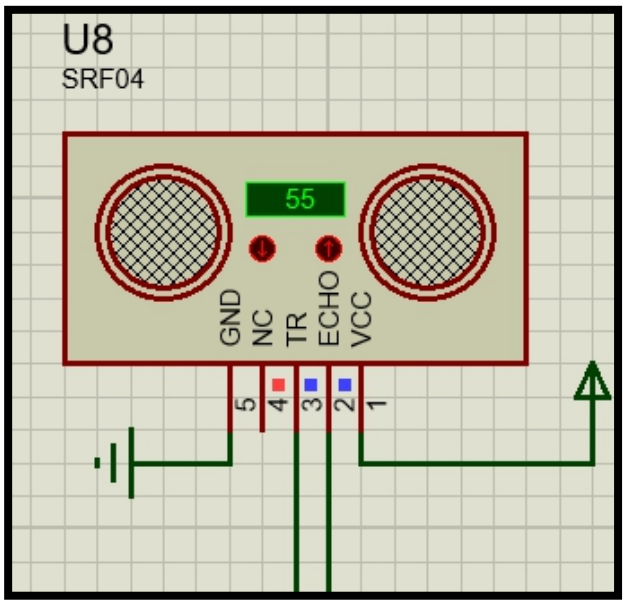

Ilustración 17. Sensor SRF04 captando una distancia de 55 cm

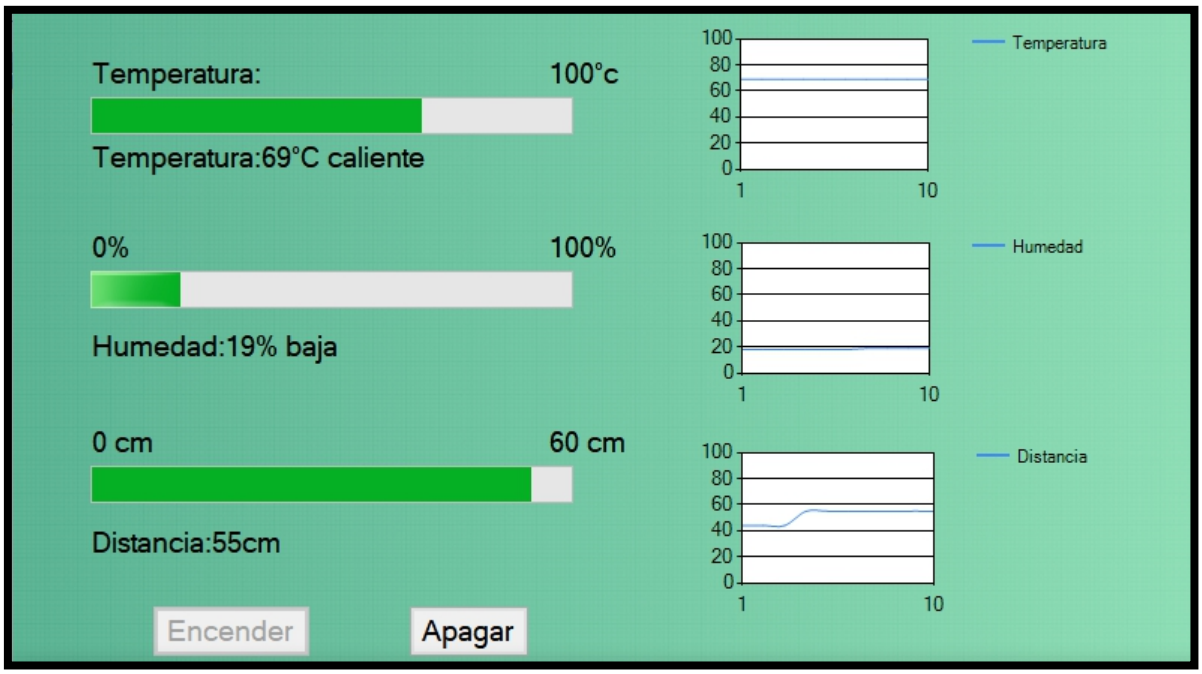

Ilustración 18. Distancia de 55cm registrada

Código del Microcontrolador

```
#include <16f877a.h>
#device adc=10
#include <stdio.h>
#fuses XT,NOWDT,NOPROTECT
#use delay(clock=4Mhz)
#use rs232 (baud=9600, parity=N, xmit=pin_c6, rcv=pin_c7, bits=8)
#define trigger pin_B1
#define echo pin_B0

void temperatura(){
   int16 valor_t;
   float tempx,temperatura;
   set_adc_channel(0);
   delay_us(20);
   valor_t=read_adc();
   tempx=(5.0*valor_t/1024.0);
   temperatura=(tempx-0.273)*(1000);
 printf("t\n");
   printf("%0.2f\r\n",temperatura);
   //delay_ms(500);
   if(temperatura<20){
      output_high(pin_D5);
   }
   else{
      output_low(pin_D5);
```

```
26    }
27 }
28 void humedad() {
29   float valor_d;
30   float humedad;
31   set_adc_channel(3);
32   delay_us(20);
33   valor_d = read_adc();
34   //valor_d=Get_Lectura(1);
35   humedad = ((5.0 * valor_d / 1024.0) * 1.0 / 0.05);
36    printf("h\n");
37   printf("%0.2f\r\n", humedad );
38   if (humedad < 60) {
39     output_high(pin_B5);
40
41   } else {
42     output_low(pin_B5);
43   }
44 }
45 void eco(){
46 int16 duty=0;
47 int Timer2,Poscaler;
48 float distancia, tiempo;
49 //Generar una señal cuadrada de 1khz
50 Timer2=249; // Se carga Timer2 con 249
51 //Pre-escalador=4, solo puede tomar valores de 1,4,16
52 Poscaler=1;
53 setup_timer_2(t2_div_by_4,Timer2,Poscaler);
54 setup_ccp1(ccp_pwm); //se configura el modulo CCP! en modo ccp_pwm
55 setup_timer_1(T1_internal|T1_div_by_1);
56 output_high(trigger);
57 delay_us(10);
58 output_low(trigger);
59
60 while(!input(echo));
61 set_timer1(0);
62
63 while(input(echo));
64 tiempo = get_timer1();
65 distancia=(tiempo/2)*(0.0343);
66
67 duty=(distancia*16.66);
68
69 set_pwm1_duty(duty); //10 bits=1023 y 8bits=255
70 delay_us(20);
71 printf("d\n");
72 printf("%0.2f\r\n",distancia);
73 }
74 void main() {
75    setup_adc_ports(AN0_AN1_AN3);
76    setup_adc(ADC_CLOCK_INTERNAL);
77   Set_tris_b(5);
```

```
78   Set_tris_c(1);
79   Set_tris_d(5);
80   for (;;) {
81     humedad();
82     temperatura();
83     eco();
84   }
85 }
```

Código de la Interfaz Gráfica de Usuario

```
 1 using System;
 2 using System.Windows.Forms;
 3
 4 namespace serial
 5 {
 6     public partial class Form1 : Form
 7     {
 8         int flag = 0, timer = 0,
 9 temperatura,humedad,distancia,flag2=0;
10         string lectura;
11         public Form1()
12         {
13             InitializeComponent();
14             button2.Enabled = false;
15             label1.Text = "Temperatura:";
16             try { serialPort1.Open(); }
17             catch (Exception msg)
18             {
19                 MessageBox.Show(msg.ToString());
20             }
21         }
22
23         private void chart3_Click(object sender, EventArgs e)
24         {
25
26         }
27
28         private void button1_Click(object sender, EventArgs e)
29         {
30             flag = 1;
31             button1.Enabled = false;
32             button2.Enabled = true;
33             label1.Text = "Temperatura:";
34             chart1.Series[0].Points.Clear();
35             progressBar1.Value = 0;
36             chart2.Series[0].Points.Clear();
37             progressBar2.Value = 0;
38             chart3.Series[0].Points.Clear();
39             progressBar3.Value = 0;
40             timer = 0;
```

```
41          }
42
43          private void progressBar2_Click(object sender, EventArgs e)
44          {
45              progressBar2.Value = humedad;
46              if (humedad <= 60)
47              {
48                  label4.Text = "Humedad:" + humedad.ToString() + "%
49  baja";
50              }
51              if (humedad > 60 && humedad < 80)
52              {
53                  label4.Text = "Humedad:" + humedad.ToString() + "%
54  normal";
55              }
56              if (temperatura >= 80)
57              {
58                  label4.Text = "Humedad:" + humedad.ToString() + "%
59  alta";
60              }
61          }
62
63          private void progressBar3_Click(object sender, EventArgs e)
64          {
65              progressBar3.Value = distancia;
66              label7.Text = "Distancia:" + distancia.ToString() + "cm";
67          }
68
69          private void chart2_Click(object sender, EventArgs e)
70          {
71
72          }
73
74          private void button2_Click(object sender, EventArgs e)
75          {
76              flag = 2;
77              button1.Enabled = true;
78              button2.Enabled = false;
79              label1.Text = "Temperatura:";
80              label4.Text = "Humedad:";
81              label7.Text = "Distancia:";
82          }
83
84          private void progressBar1_Click(object sender, EventArgs e)
85          {
86              progressBar1.Value = temperatura;
87              if (temperatura <= 20)
88              {
89                  label3.Text = "Temperatura:" +
90  temperatura.ToString()+"°C frio";
91              }
92              if (temperatura > 20&& temperatura<30)
```

```
            {
                label3.Text = "Temperatura:" + temperatura.ToString()
+ "°C bien";
            }
            if (temperatura >= 30)
            {
                label3.Text = "Temperatura:" + temperatura.ToString()
+ "°C caliente";
            }
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            if (flag == 1)
            {
                timer++;
                if (timer == 10)
                {
                    timer = 0;
                    chart1.Series[0].Points.Clear();
                    chart2.Series[0].Points.Clear();
                    chart3.Series[0].Points.Clear();
                }


chart1.Series["Temperatura"].Points.AddY(temperatura);


                    chart2.Series["Humedad"].Points.AddY(humedad);


                    chart3.Series["Distancia"].Points.AddY(distancia);


            }
        }

        private void serialPort1_DataReceived(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
        {
            if (flag == 1)
            {
                lectura = serialPort1.ReadLine();
                if (lectura == "h") {
                    lectura = serialPort1.ReadLine();
                    double hum = Convert.ToDouble(lectura);
                    humedad = Convert.ToInt16(hum);
                    Invoke(new EventHandler(progressBar2_Click));
                }
                if (lectura == "t")
                {
```

```
145                     lectura = serialPort1.ReadLine();
146                     double temp = Convert.ToDouble(lectura);
147                     temperatura = Convert.ToInt16(temp);
148                     Invoke(new EventHandler(progressBar1_Click));
149                 }
150                 if (lectura == "d")
151                 {
152                     lectura = serialPort1.ReadLine();
153                     double dist = Convert.ToDouble(lectura);
154                     distancia = Convert.ToInt16(dist);
155                     Invoke(new EventHandler(progressBar3_Click));
156
                }
            }
        }
        private void Proyecto(object sender, EventArgs e)
        {
            //Esta es una funcion externa a los componentes con invoke
la llamampps
        }
    }
}
```

Conclusiones

Con esta práctica implementamos todos los conocimientos adquiridos sobre la configuración de un programa en c para un microcontrolador, desde el manejo de la información provenientes por las entradas , hasta su posterior salida con el estándar rs232 , sobre todo se dio una vistazo de mayor profundidad a la comunicación serial, la cual fue de vital importancia para conectar dos aplicaciones como lo fueron al simulación en proteus y la interfaz que sirvió para representa los datos de una manera más amigable para el usuario mediante la presentación de elementos de interfaz gráfica como lo son barras de carga y gráficas que van variando en el tiempo con los valores que va recibiendo, esto se programó usando el lenguaje de programación visual c# el cual ya contaba con elementos que facilitaron la conexión, como un objeto llamado SerialPort, al cual solo se debió configurar algunas características de la comunicación, como su configuración de baudios, de bits, paridad , el puerto usado, etc.

Rivera Pérez Ricardo

La mayor parte de microcontroladores de gama media y alta incluyen un componente de comunicación USART (Universal Synchronous Asynchronous Receiver Transmitter), que permiten la comunicación entre microcontroladores, o bien entre un microcontrolador y una computadora, en este proyecto usamos una comunicación serial que transmite los bits de información bit a bit, es decir envió de datos de forma secuencial.

Para el desarrollo de la interfaz gráfica usamos un nuevo lenguaje de programación C# (C Sharp), ya que encontramos vasta documentación para generar una comunicación serial, primeramente, intentamos prográmalo en Java, pero necesitábamos el uso de librerías ajenas a este lenguaje, por lo cual desistimos.

Empezamos a buscar alternativas para la correcta realización del proyecto lo que nos llevo a programar en este nuevo lenguaje y ampliar nuestros horizontes, sin olvidar que tuvimos una previa etapa de acondicionamiento del circuito.

Sin duda ha sido una experiencia fenomenal aprender como se comunican los sensores de medición ya sean de temperatura, humedad, ultrasonido, u de otro tipo, con una computadora, así poder generar sistemas de información que permitan ayudar al ser humano.

Márquez León Jorge Luis.

Al desarrollar el proyecto se implementaron los conocimientos adquiridos a lo largo del semestre, ya sea para implementar un sensor, con su respectiva fase de acondicionamiento, para después juntar todos los sensores y desarrollar un sistema que sea capaz de mostrarle al usuario los datos en forma de barras de carga, así como mostrar el valor numérico. Esto se logra usando un microcontrolador para el manejo adecuando de los valores y desarrollo de operaciones, hasta la salida estándar rs232.

Para el desarrollo de la interfaz gráfica se había optado por desarrollarla en el lenguaje de programación Java, sin embargo ya que esté no cuenta con bibliotecas nativas para la comunicación serial nos era más complejo implementar, principalmente por la escasa documentación que había en Internet, por eso se decidió cambiar a un lenguaje que contará con una mayor documentación de la comunicación serial, lo que permitió desarrollar la interfaz más rápido, además de ampliar nuestro conocimiento en lenguajes de programación.

Montoya Uribe Miguel Angel

Referencias

[1] C.González, «¿Qué es la Instrumentación Virtual?,» Revista ElectroIndustria, 2006. [En línea]. Available: http://www.emb.cl/electroindustria/articulo.mvc?xid=471&ni=que-es-la-instrumentacion-virtual.. [Último acceso: 21 Enero 2021].

[2] J. A. Corrales, «Interfaz de usuario o UI: ¿qué es y cuáles son sus características?,» rockcontent, 2 Agosto 2019. [En línea]. Available: https://rockcontent.com/es/blog/interfaz-de-usuario/. [Último acceso: 21 Enero 2021].

[3] «Instrumentación Industrial,» Wikipedia, 8 Noviembre 2020. [En línea]. Available: https://es.wikipedia.org/wiki/Instrumentaci%C3%B3n_industrial. [Último acceso: 21 Enero 2021].

[4] «Estándar RS232,» Cidecame, [En línea]. Available: http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro27/342_estndar_rs232.html. [Último

acceso: 20 Enero 2021].

[5] O. Weis, «Comunicación Serie,» Eltima Publishing, 4 Febrero 2020. [En línea]. Available: https://www.serial-port-monitor.org/es/articles/serial-communication/. [Último acceso: 20 Enero 2021].

[6] «HCH-1000,» Mouser Electronics, [En línea]. Available: https://www.mouser.mx/Electronic-Components/. [Último acceso: 20 Diciembre 2020].

[7] «Sensor Ultrasonido,» Todo tipo de Sensores, 19 Noviembre 2019. [En línea]. Available: https://sensores.top/sensor-ultrasonido. [Último acceso: 7 Enero 2021].