Volumen 10
XXVI Verano de la Ciencia
ISSN 2395-9797
www.jovenesenlaciencia.ugto.mx

Implementación de una estación de monitoreo y adquisición de datos a distancia con Arduino y Visual Studio.NET

Juan Antonio Sánchez-Márquez[1], Carlos Arturo Salazar-Parada[2], Melanie Guadalupe Alanís-Serrato[3], Roberto Jafet Pérez-Vázquez[4], Tristán Azael Sánchez-Ramírez[5], Bruno Daniel Pérez-Vázquez[6], Cristina López-Negrete[7]

[1]Escuela del Nivel Medio Superior de Salamanca, Colegio del Nivel Medio Superior UG
[2]Licenciatura en Ingeniería Civil, División de Ingenierías, Campus Guanajuato
[3]Licenciatura en Ingeniería en Agronomía, División de Ciencias de la Vida, Campus Irapuato-Salamanca
[4]Licenciatura en Ingeniería en Comunicaciones y Electrónica, División de Ingenierías, Campus Irapuato-Salamanca
[5]Escuela del Nivel Medio Superior de Guanajuato, Colegio del Nivel Medio Superior UG
[6]Licenciatura en Ingeniería en Sistemas Computacionales, División de Ingenierías, Campus Irapuato-Salamanca
[7]Licenciatura en Ingeniería Química, División de Ciencias Naturales y Exactas, Campus Guanajuato

Resumen

A pesar de que en la actualidad se cuenta con una gran variedad de instrumentos y sensores comerciales que permiten tomar y almacenar datos en diversos procesos; la realidad nos muestra que estos dispositivos no suelen ser económicos y sigue existiendo la necesidad de disponer de aparatos que cuenten con instrumentos y sensores capaces de recopilar y almacenar información a bajo costo, sin que esto implique comprometer la confiabilidad de la información recolectada. Estos dispositivos deben ser accesibles a productores, investigadores o estudiantes con un presupuesto limitado en el monitoreo de sus procesos de investigación y producción; de allíque resulta evidente la necesidad de contar con un sistema de monitoreo, que además actué como un sistema de adquisición de datos (SAD). En este proyecto se desarrollará un sistema de monitoreo y adquisición de datos integrando la plataforma del microcontrolador Arduino con la herramienta de programación VisualBasic.NET. Esta integración nos permitirá contar con interfaces gráficas más atractivas y amigables para los usuarios basadas en la comunicación de la información recolectada por el microcontrolador Arduinoy los entornos construidos en VisualBasic.NET.

Palabras clave: Estación; monitoreo; datos; distancia; Arduino; Visual Studio.NET.

Introducción: Generalidades de Arduino y Visual Studio.NET

Microcontrolador Arduino

Arduino es una plataforma electrónica de código abierto basada en hardware y software gratuitosy fáciles de usar, estas placas pueden leer distintas entradas y convertirlas en una salida, para que sea capaz de llevar a cabo estas acciones se debe utilizar un lenguaje de programación basado enWiring y un Software Arduino basado en Processing. Arduino nació en el Ivrea Interaction Design Institute como una herramienta fácil para la creaciónrápida de prototipos, utilizada a lo largo de los años como el cerebro de miles de proyectos el Arduino comenzó a cambiar para adaptarse a las nuevas necesidades y desafíos, todas las placas Arduino son completamente de código abierto, lo que permite a los usuarios construirlas de forma independiente y eventualmente adaptarlas a sus necesidades particulares.

Arduino también simplifica el proceso de trabajar con microcontroladores, pero ofrece algunas ventajas para profesores, estudiantes y aficionados interesados sobre otros sistemas.

Unas de sus principales características son las siguientes:

- Es bastante a económico a comparación de otros microcontroladores.
- Es multiplataforma y se puede ejecutar en más sistemas operativos además de Windows
- Este tiene un entorno de programación simple y claro, es fácil de usar para principiantes,pero lo suficientemente flexible para que los usuarios avanzados también lo aprovechen.
- Maneja Software de código abierto y extensible.

Autodesk Tinkercad Circuits

TinkerCAD es una colección online que incluye herramientas de software de Autodesk que permite a los usuarios crear modelos 3D, además de circuitos electrónicos. Este software CAD permite crear modelos complejos mediante la combinación de objetos más simples, Thinkercad circuits es una herramienta que dispone de los elementos necesarios para crear y simular sistemas de controlbasados en Arduino. Además, permite la programación online de las placas Arduino del simulador,el resultado final incluso se puede simular y con ello se optimizan los resultados obtenidos, Figura 1.

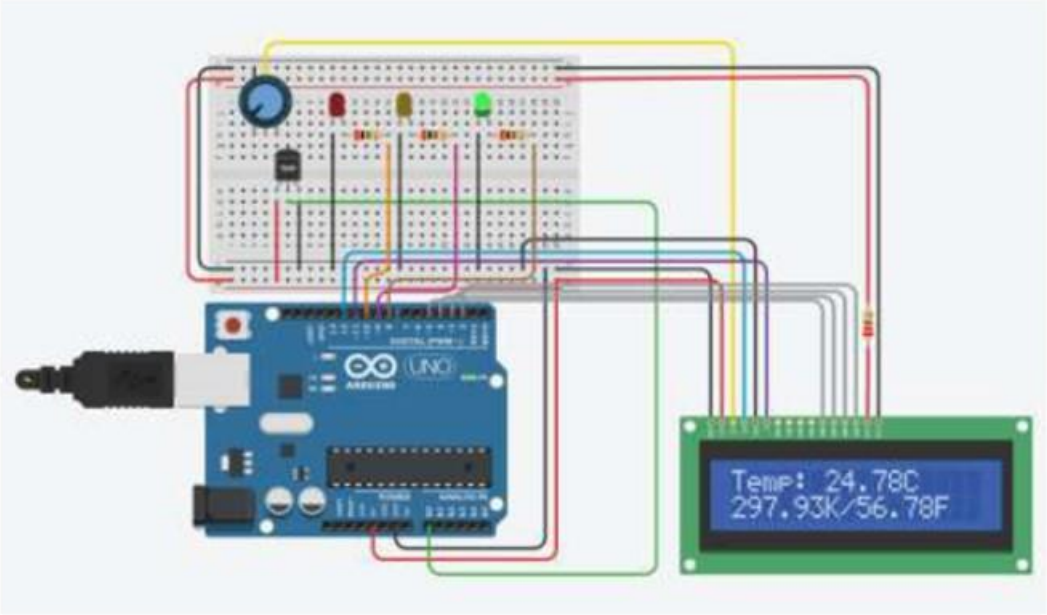

Figura 1. Arreglo electrónico del sensor de inclinación Sw-520d desarrollado en Autodesk Tinkercad Circuits.

Visual Basic.NET

Visual Basic es un lenguaje de programación orientado a objetos desarrollado por Microsoft. El uso de Visual Basic simplifica la creación de aplicaciones .NET además permite crear aplicaciones de línea de negocio (LOB) con un conjunto completo de características bastantes útiles.

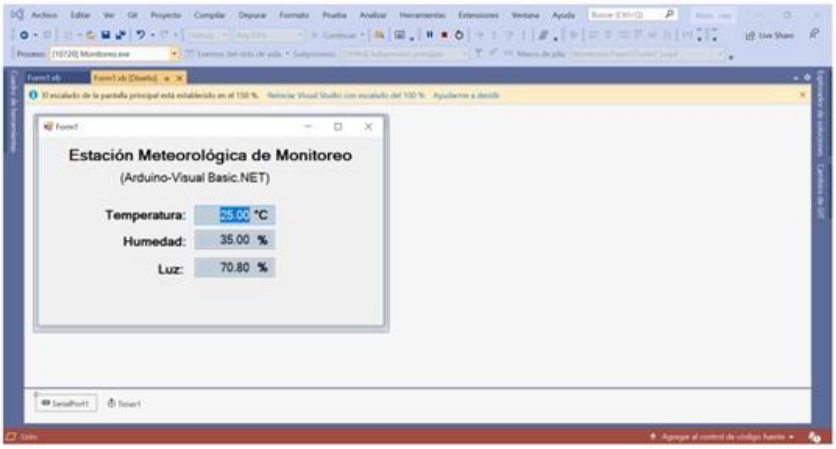

Figura 2. Aplicación de la Estación de Monitoreo desarrollada en Visual Studio.NET.

Materiales y Recursos

Sensores usados con Arduino

a) **Sensor de temperatura LM35:** es un circuito electrónico sensor que puede medir temperatura, proporciona al Arduino un voltaje proporcional a la temperatura que recibe, el LM35proporciona 10mV por cada grado centígrado y tiene un rango desde -55°C a 150°C, Figura 3.

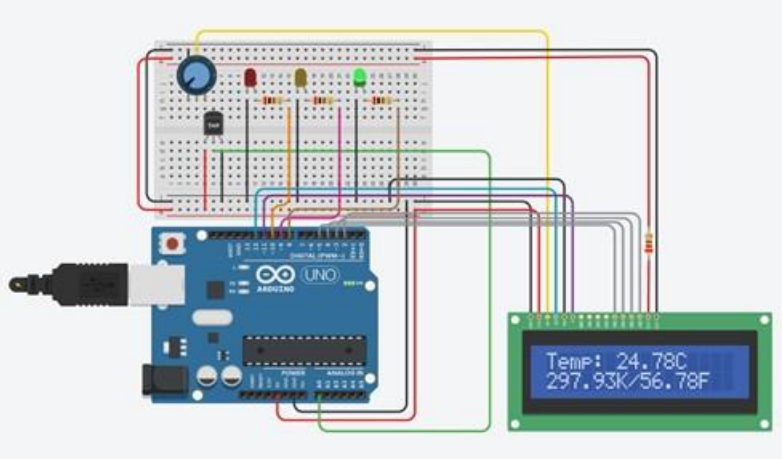

Figura 3. Funcionamiento del Sensor de Temperatura LM35 en el arreglo electrónico montado en Autodesk Tinkercad Circuits.

b) **Sensor ultrasónico HC-SR04:** Es un sensor que utiliza ultrasonido para determinar la distancia de un objeto, esto lo hace por medio de dos transductores un emisor y un receptor piezoeléctrico, el emisor genera 8 pulsos de ultrasonido a 40kHz, el sonido choca con el objeto del cual se requiere medir la distancia a la que se encuentra, y nuestro receptor lo detecta. Los programas utilizan la librería LiquidCrystal para hacer uso de la pantalla LCD, y mandar comandos hacia el dispositivo de una manera más sencilla. La función "ReadUltrasonicDistance" se utiliza para obtener los datos del sensor necesarios para calcular la distancia y la función setup, para declarar puertos como digitales o analógicos, tamaño del LCD, y saber la velocidad de la comunicación de nuestra computadora con el Arduino. Por último, en el main se realizan cálculos para saber la distancia de nuestro sensor hacia el obstáculo detectado, para calcular conversiones y para imprimir datos en la pantalla LCD, Figura 4.

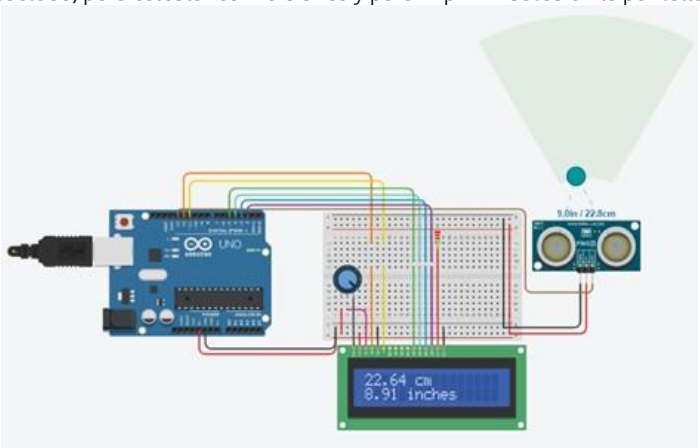

Figura 4. Valores medidos por el Sensor Ultrasónico Hc-sr04 mostrados en la pantalla LCD.

c) **Sensor de Fuerza Resistivo Fsr402:** Este es un componente pasivo que exhibe una disminución de la resistencia cuando se produce un aumento en la fuerza aplicada a la superficie activa de, lo que permite crear un sensor que es capaz de detectar la fuerza o presión. El programa utiliza la librería LiquidCrystal para mostrar los valores medidos por el sensor en la pantalla LCD, Durante el desarrollo del código debemos establecer los pines digitales que le corresponderán a los LEDs, los pines que actuarán como analógicos, y aquellos en los que se conectará la pantalla LCD, Además, es necesario configurar la velocidad de la comunicación serial que tendrá lugar entre nuestra computadora y el microcontrolador Arduino, Figura 5.

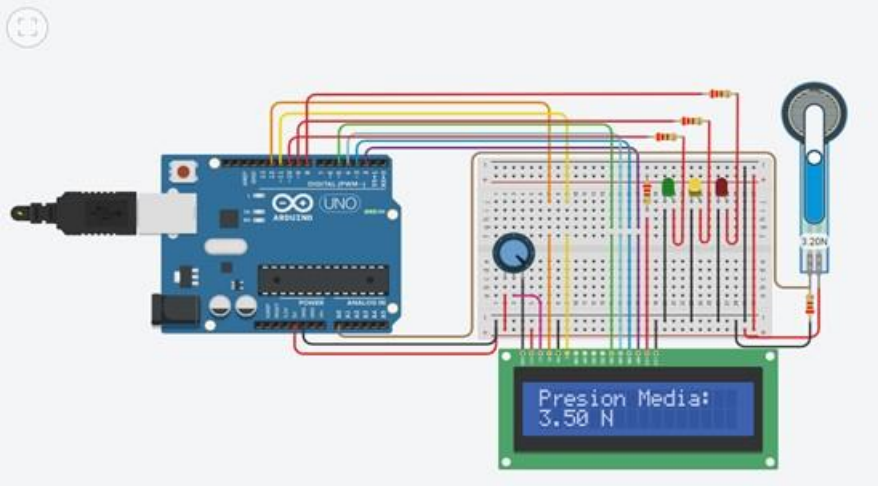

Figura 5. Indicadores de presión visual (LED Amarillo) y digital (Pantalla LCD- Presión Moderada).

d) **Fotorresistencia-LDR**: Es un componente fotoelectrónico cuya resistencia varía en función de la luz que incide en él, de unos pocos Ωs con una luz intensa incide en él y va creciendo fuertemente a medida que esa luz decrece (Se les suele utilizar como sensores de luz). A medida que desplazamos la banda de intensidad luminosa de izquierda a derecha podemos ver como los niveles de intensidad se van activando gradualmente, encendiendo los LEDs indicadores de nivel y mostrando el nivel de intensidad en la pantalla LCD, Figura 6.

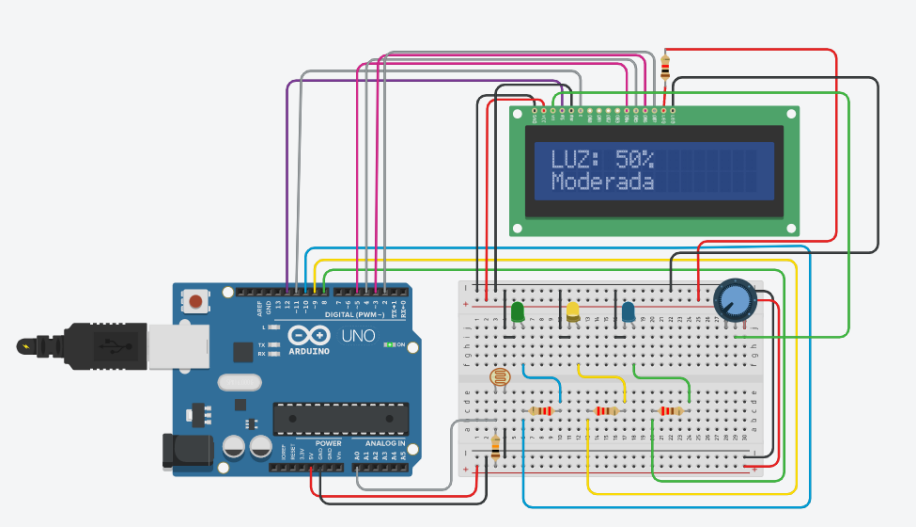

Figura 6. Intensidad de luz moderada medida por el sensor LDR mostrada en la pantalla LCD.

e) **Sensor de inclinación Sw-520d**: Es un dispositivo que proporciona una señal digital en caso de que su inclinación supere un umbral. Este tipo de sensor no permite saber el grado de inclinación del dispositivo, simplemente actúa como un sensor que se cierra a partir de una cierta inclinación. El programa utiliza la librería LiquidCrystal para hacer uso de la pantalla LCD, Al inclinar lo suficiente el dispositivo ambas esferas (Estas se encuentran en un cilindro que está en su interior) constituyen un puente entre ambos contactos, cerrando el circuito. Debido a su principio de funcionamiento, estos sensores resultan sensibles a movimientos bruscos y vibraciones. Cuando nosotros ajustamos la posición del sensor mediante la barra de desplazamiento podemos pasar de un estado "En posición" a un estado "Inclinado" al abrir o cerrar el circuito, Figura 7.

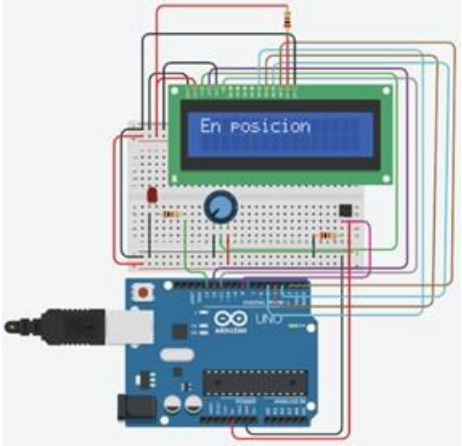

Figura 7. Sensor de inclinación Sw-520d en posición desarrollado en Autodesk Tinkercad Circuits.

Desarrollo del Proyecto

Comunicación Serial Arduino- Visual Studio.NET

El primer paso consiste en desarrollar el código (Sketch) en la placa Arduino para recibir el dato desde Visual Basic.NET. Una vez desarrollado el código descargamos el sketch a la tarjeta Arduino verificando el puerto COM al que se ha conectado la tarjeta. Para realizar una prueba de comunicación y verificar que nuestro programa cargado sea el correcto, se utiliza la consola de comunicación serial del IDE de Arduino, Al escribir 1 o 0 en la consola podemos encender o apagarel LED conectado a la tarjeta, Figura 8.

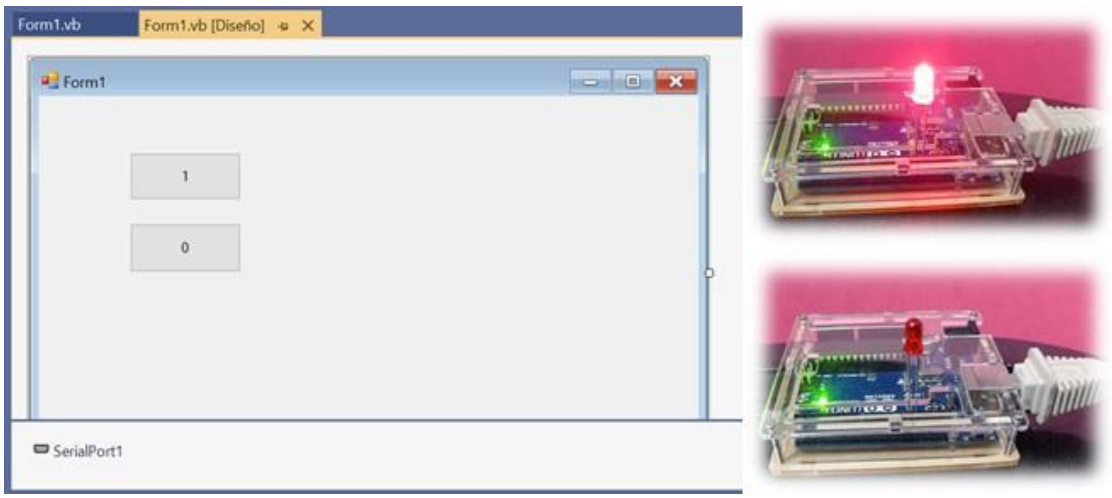

Figura 8. Encendiendo y Apagando el LED de la Tarjeta Arduino con Visual Basic.NET.

Desarrollo de la Estación Meteorológica de Monitoreo con Arduino y Visual Basic.NET

Para la implementación de la Estación de Monitoreo se requiere armar el arreglo electrónico del sensor de Medición de Humedad y Temperatura (DHT11) y de la fotorresistencia incluida en la estación, Figura 9-10.

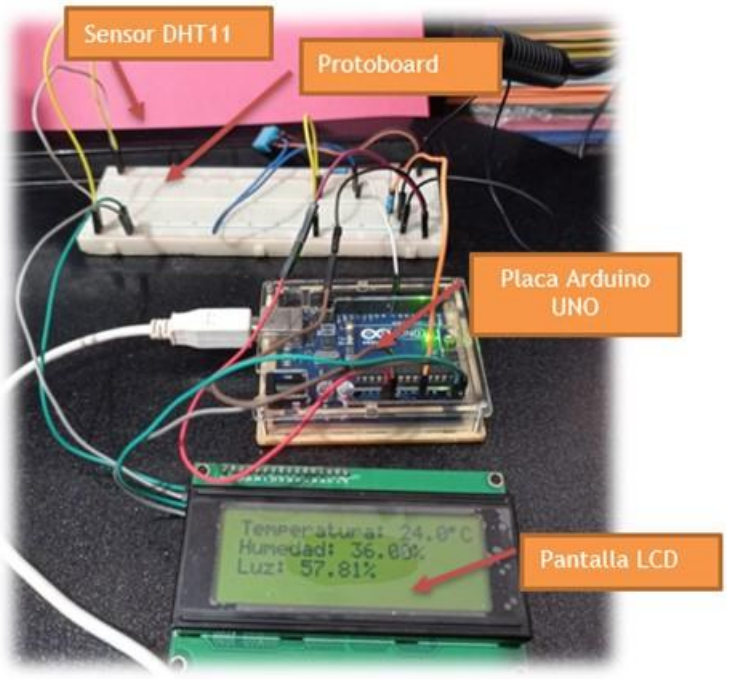

Figura 9. Arreglo Electrónico de la Estación de Monitoreo.

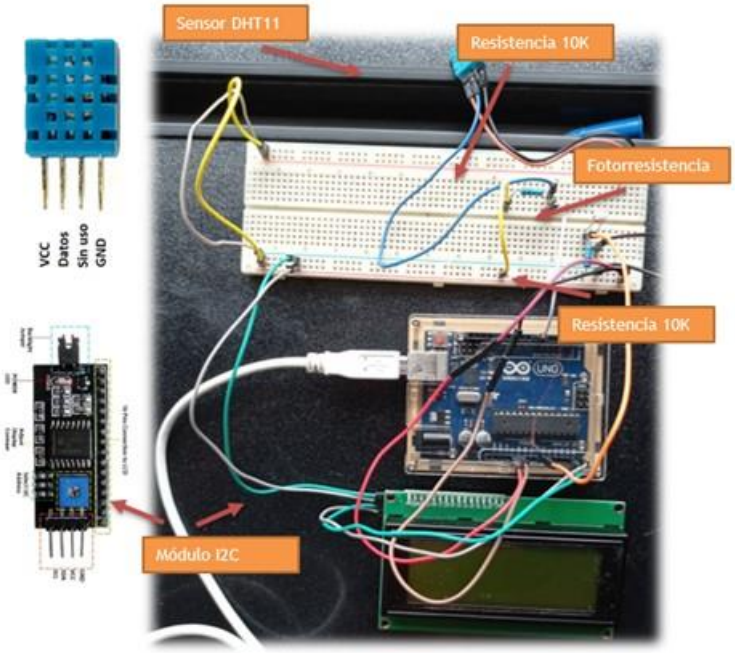

Figura 10. Conexión de Sensores en la Estación de Monitoreo.

Una vez que el hardware del proyecto está totalmente montado, hay que probar los diferentes sensores mediante la construcción del sketch que permita realizar mediciones y que muestre los valores de las variables definidas en la pantalla LCD 20x4. Para ello, se escribe un sketch en el IDE de Arduino que incluya las siguientes librerías: Wire.h, DHT.h y LiquidCrystal_I2C.h. Para instalar las librerías es necesario descargar las carpetas de algún sitio web y copiar éstas en la carpeta Arduino/libraries de nuestro ordenador. El código desarrollado para el arreglo correspondiente a la Estación de Monitoreo se detalla en la Figura 33. El código debe compilarse y descargarse a la placa de Arduino a fin de ponerlo a prueba. También debe abrirse el comunicador serial para visualizar los datos enviados. Es importante verificar que la velocidad de transmisión del puerto serial esté a 9600. En las Figuras 11 se muestra el funcionamiento del código.

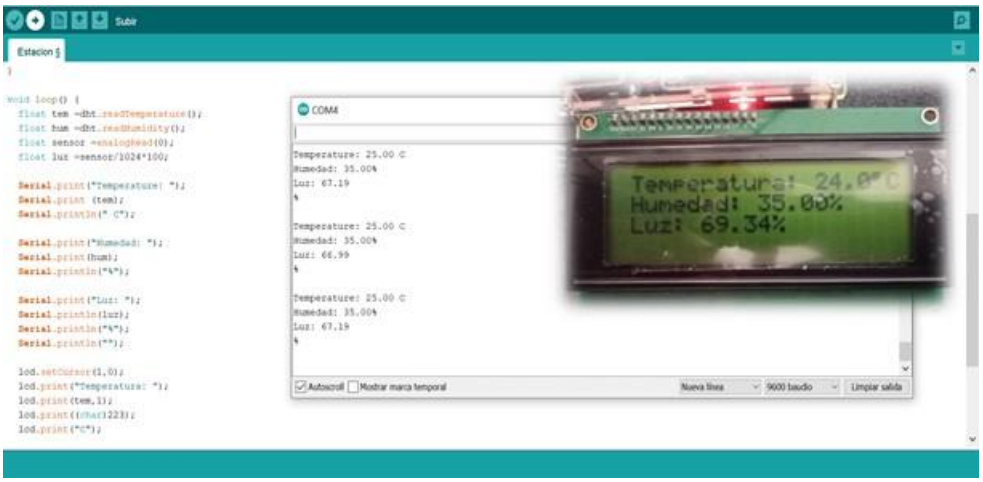

Figura 11. Valores reportados en la pantalla LCD.

Antes de proceder a crear la interfaz, es necesario adecuar el código del arreglo electrónico a fin de que los datos que se envíen al monitor serial se muestren en una sola línea. La interfaz que se desarrollará en Visual Basic.NET requiere que los datos que arroje el controlador se presenten en este formato, Figura 12.

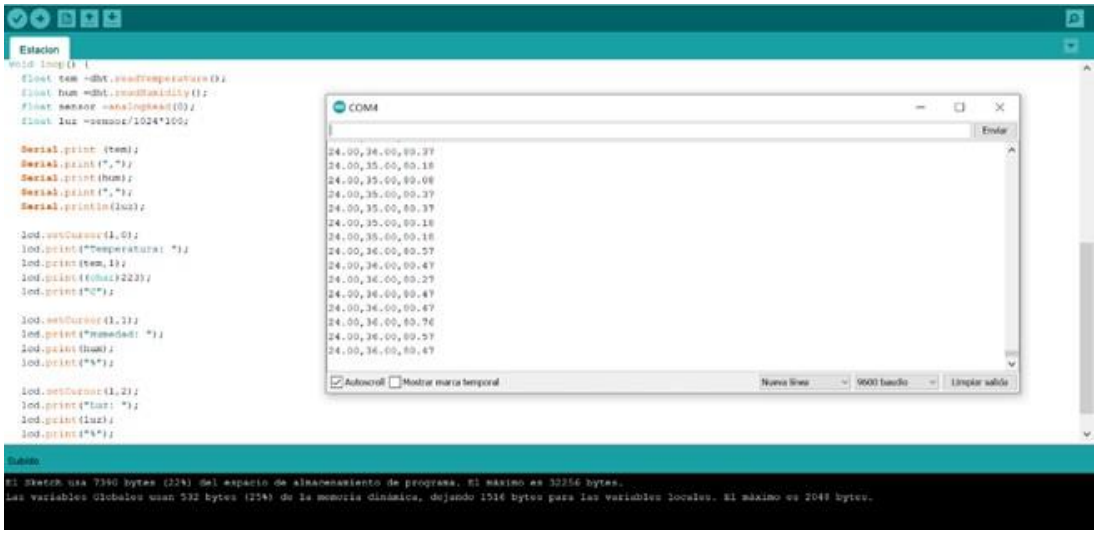

Figura 12. Valores mostrados en el monitor serial en formato de una sola línea.

A continuación, se detalla el código de Arduino desarrollado para la estación de monitoreo.

```
#include <DHT.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

//Sensor DHT11
#define DHTPIN 7
#define DHTTYPE DHT11

LiquidCrystal_I2C lcd(0x27,20,4);

// Instancia del Sensor
DHT dht(DHTPIN,DHTTYPE);

void setup() {
  lcd.begin();
  lcd.backlight();
  lcd.setCursor(1,0);
  lcd.print("Hola!!!");
  lcd.setCursor(1,1);
  lcd.print("Inicializando...");
  Serial.begin(9600);
  dht.begin();
  delay(2000);
  lcd.clear();
}

void loop() {
  float tem =dht.readTemperature();
  float hum =dht.readHumidity();
  float sensor =analogRead(0);
  float luz =sensor/1024*100;

  Serial.print("Temperature: ");
  Serial.print (tem);
  Serial.println(" C");

  Serial.print("Humedad: ");
  Serial.print(hum);
  Serial.println("%");

  Serial.print("Luz: ");
  Serial.println(luz);
  Serial.println("%");
  Serial.println("");

  lcd.setCursor(1,0);
  lcd.print("Temperatura: ");
  lcd.print(tem,1);
  lcd.print((char)223);
  lcd.print("C");

  lcd.setCursor(1,1);
  lcd.print("Humedad: ");
  lcd.print(hum);
  lcd.print("%");

  lcd.setCursor(1,2);
  lcd.print("Luz: ");
  lcd.print(luz);
  lcd.print("%");
  delay(700);
}
```

Volumen 10
XXVI Verano de la Ciencia
ISSN 2395-9797
www.jovenesenlaciencia.ugto.mx

Desarrollo de la Interfaz de Monitoreo con Visual Basic.NET

Para crear una interfase de monitoreo es necesario crear una aplicación en Visual Basic.NET 2019. Una vez abierto el programa, debemos hacer clic en crear un proyecto, elegir Aplicación de Windows Forms (.NET Framework) y darle Nombre. Es importante elegir un Framework del 4 en adelante. Una vez creado el proyecto procedemos a incluir el control SerialPort del entorno de Visual Basic.Net el cual permite establecer una comunicación serial con cualquier dispositivo, Figura 36. Una vez que el control SerialPort está en el formulario, e posible ver las propiedades del control. Se requiere que dichas propiedades queden configuradas de la siguiente manera: BaudRate= 9600, Databits= 8, Parity= None, PortName= COM4 (debe coincidir con el puerto al que está conectado el Arduino). Una vez configurado el SerialPort procedemos a incluir el control Timer, el cual, permitirá establecer un tiempo de lectura para que el búfer no se sature y, en consecuencia, el envío y recepción de los datos estén sincronizados. En las propiedades del objeto Timer se deben realizar los siguientes ajustes: Propiedad Interval (especificar el valor de 2000 o, mínimo, 1900 milisegundos). No se recomienda que sea menor, ya que los cambios en los valores serán muy rápidos y no se podrán visualizar correctamente. En la Figura 13 se señalan los controles adiciones que se agregan al formulario, los cuales son esencialmente: Labels, Textbox, SerialPort y Timer, con los cuales se desarrollará la aplicación.

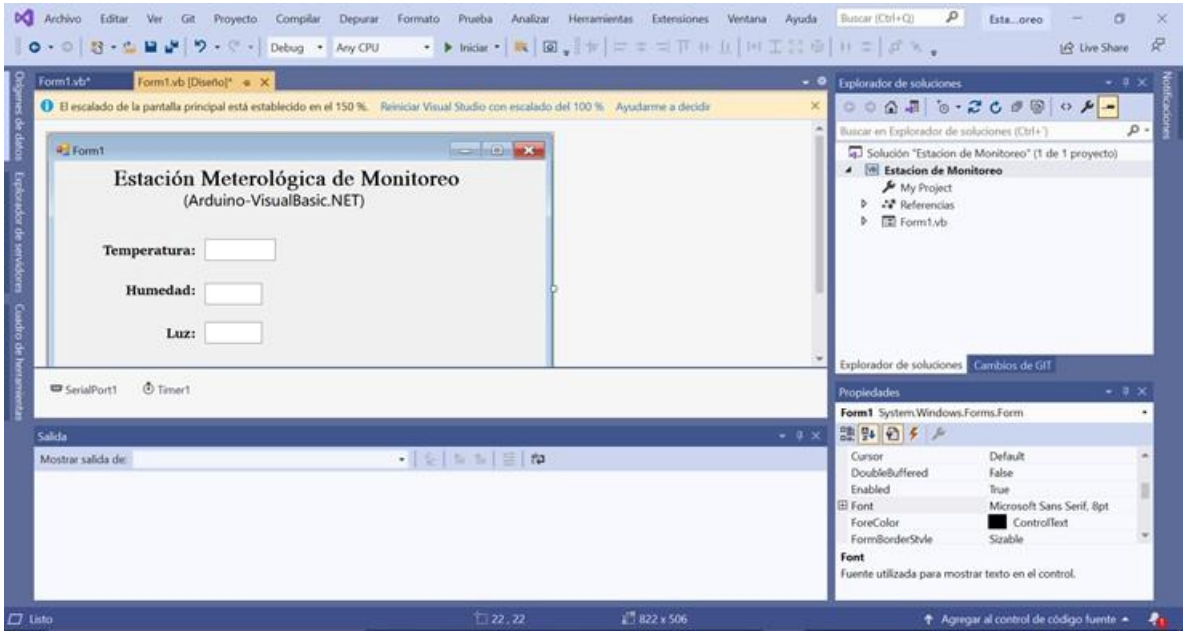

Figura 13. Controles incluidos en la aplicación de la Estación de Monitoreo.

Para leer los datos enviados desde la tarjeta Arduino en el tiempo establecido, hay que abrir el puerto y habilitar el control Timer incluyendo las siguientes líneas de código, Figura 14. Una vez escrito el código del formulario y del Timer podremos visualizar los valores reportados por el Arduino en la interfaz desarrollada en Visual Basic.NET. En esta interfaz tendremos los valores de temperatura, humedad y luz en tiempo real tal y como se muestra en la Figura 15. Estos valores coinciden con los mostrados en la pantalla LCD del arreglo electrónico.

Volumen 10
XXVI Verano de la Ciencia
ISSN 2395-9797
www.jovenesenlaciencia.ugto.mx

```
1 referencia
Public Class Form1
    Dim cadena As String

    0 referencias
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        SerialPort1.Open()
        Timer1.Enabled = True
    End Sub

    0 referencias
    Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick
        Dim cadena1 As String
        Dim cadena2 As String
        Dim cadena3 As String

        cadena = SerialPort1.ReadExisting()

        cadena1 = Mid(cadena, 1, 5)
        cadena2 = Mid(cadena, 7, 5)
        cadena3 = Mid(cadena, 13, 5)

        txttem.Text = cadena1.Trim
        txthum.Text = cadena2.Trim
        txtluz.Text = cadena3.Trim


    End Sub
End Class
```

Figura 13. Código del formulario y del Control Timer.

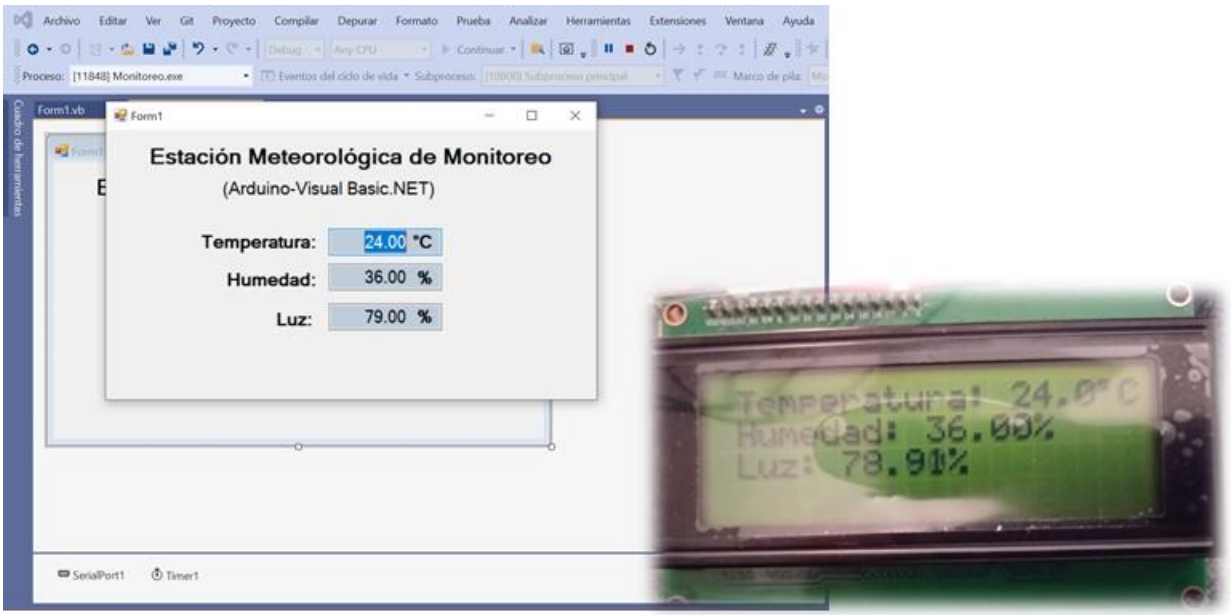

Figura 14. Estación de Monitoreo desarrollada con Arduino y Visual Basic.NET

Registro y Respaldo de Datos (Modulo SD)

Al arreglo electrónico anterior se le adicionó un módulo microSD que servirá de respaldo de la estación de registro de datos para guardar los datos localmente, Figura 15. Esto se hace con el fin de tener almacenado el momento en el que ocurrió un suceso, se activó un evento dentro del proceso o si un valor registrado sobrepasa un límite programado. La incorporación del Módulo MicroSD demanda del uso de las librerías SD.h y SPI.h incluidas ya en el IDE de Arduino. Por lo que no se requiere su descarga de fuentes externas.

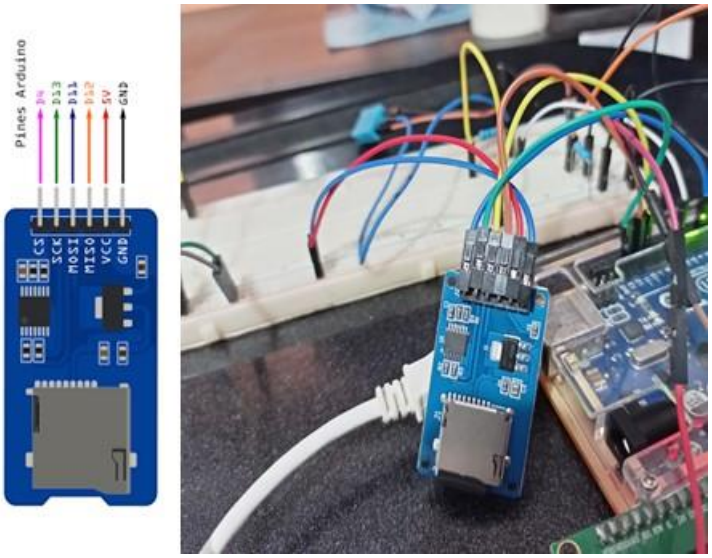

Figura 15. Modulo Micro SD de la Tarjeta de Arduino agregado a la Estación de Monitoreo.

El funcionamiento del módulo puede ser apreciado en las Figuras 16. Podemos ver en estas imágenes como la tarjeta MicroSD se inicializa correctamente y comienza a crear el archivo data.txt, registrando los valores de tempertura y humedad propocionados por el sensor DHT11. Cada segundo se realiza el registro de un nuevo valor en el archivo data.txt.

Figura 16. Funcionamiento Sketch desarrollado para la incorporación del Módulo MicroSD.

A continuación, se presenta el Sketch desarrollado en el IDE de Arduino a fin de realizar el respaldo local de los datos.

```
//Librerías del programa
#include <DHT.h>
#include <SD.h>
#include <SPI.h>

#define DHTPIN 7
#define DHTTYPE DHT11

const int chipSelect =4; // Pin de selección de la memoria.
DHT dht(DHTPIN, DHTTYPE);

void setup() {
 Serial.begin(9600);
 // Se verifica que la memoria se haya detectado.
 Serial.println("Inicializando Tarjeta SD...");
  if (!SD.begin(chipSelect)) {
    Serial.println("Tarjeta no presente");
    return;
  }
  Serial.println("Memoria SD Inicializada.");
  dht.begin();
}


void loop() {
 float h = dht.readHumidity();
 float t = dht.readTemperature();

 String temp = String((int) t);
 String hum = String((int) h);
 String valor = temp + "," + hum; // Se concantenan los valores.

 File dataFile = SD.open("data.txt",FILE_WRITE);  // Se crea el archivo.

//Se escribe el archivo, se cierra y se muestra el valor guardado en el monitor serial.
 if(dataFile) {
  dataFile.println(valor),
  dataFile.close();
  Serial.println(valor);
 }
 else {
  Serial.println("Error al abrir data.txt");
 }
 delay(1000); // Cada segundo se repite el proceso de guardado.

}
```

Sistema de Monitoreo y Adquisición a través del Módulo Bluetooth HC-05

Al arreglo electrónico desarrollado para el monitoreo y adquisición de datos se le adicionó un módulo Bluetooth HC-05 que servirá para realizar la transferencia y lectura de datos a distancia mediante tecnología Bluetooth, Figura 17. Esto se hace con el fin de poder enviar los datos desde Arduino hasta un dispositivo a distancia teniendo como intermediario de la comunicación al sensor HC-05.

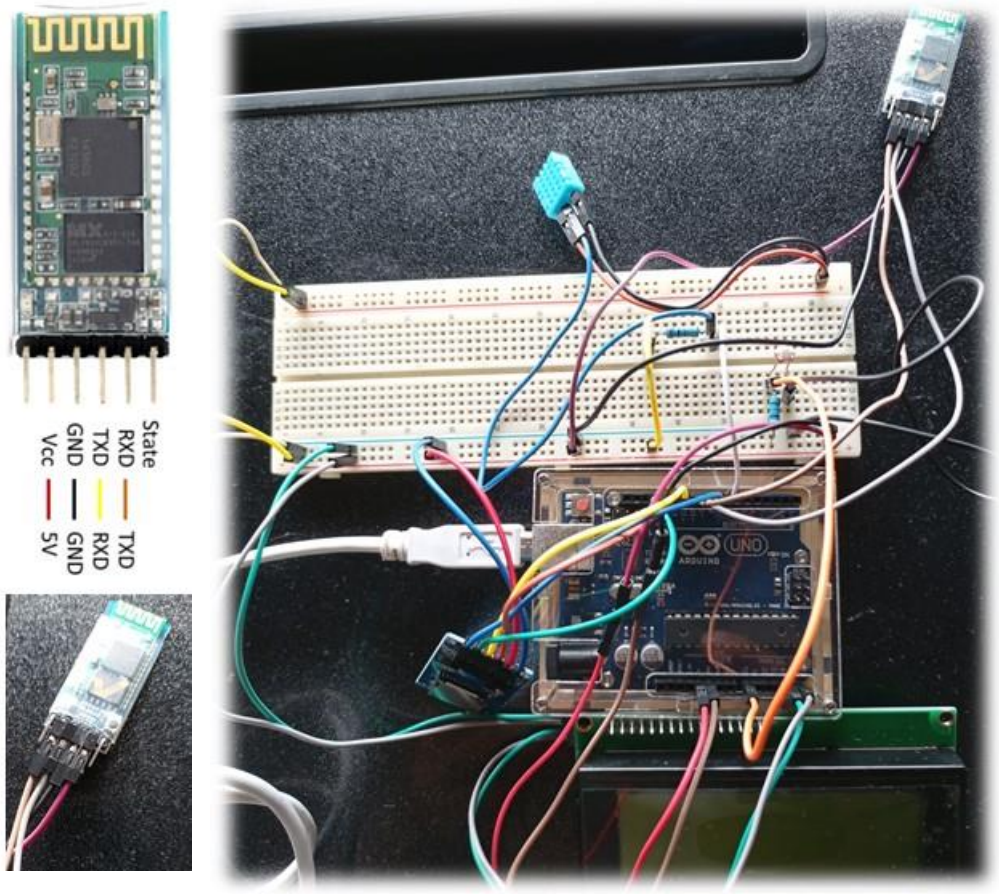

Figura 17. Arreglo Electrónico de la Estación de Monitoreo y Adquisición de Datos con un módulo Bluetooth HC-05.

Una vez montado el Módulo Bluetooth HC-05 procedemos a desarrollar el Sketch que nos permita configurar el Módulo, Figura 18. Una vez desarrollado y cargado el Sketch procedemos a configurar nuestro Módulo Bluetooth HC-05 como esclavo. Cuando está configurado de esta forma, se comporta similar a un HC-06, esperando a que un dispositivo bluetooth maestro se conecte a este. Generalmente este modo se utiliza cuando el módulo se conecta con una PC o Celular, pues éstos se comportan como dispositivos maestros. Una vez configurado el Módulo Bluetooth vinculamos éste con la PC o con el dispositivo móvil en el que recibiremos los datos suministrados por el sensor. Agregamos el dispositivo con el nombre dado en el paso anterior "Estacion" y con clave previamente definida "1212", Figura 18-20. Una vez vinculado el Módulo Bluetooth las posibilidades del uso de la estación se monitoreo se amplían enormemente toda vez que la estación puede vincularse con aplicaciones desarrolladas para dispositivos móviles en las versiones de Android o IOS; o bien puede vincularse con directamente con formularios de Visual Studio.NET. Para esto basta con configurar el parámetro PortName para que coincida con el puerto del módulo Bluetooth designado para el módulo BT.

```
#include <SoftwareSerial.h>   // Incluimos la librería  SoftwareSerial
SoftwareSerial BT(9,10);    // Definimos los pines RX y TX del Arduino conectados al Bluetooth

void setup()
{
  BT.begin(9600);       // Inicializamos el puerto serie BT (Para Modo AT 2)
  Serial.begin(9600);   // Inicializamos  el puerto serie
}

void loop()
{
  if(BT.available())    // Si llega un dato por el puerto BT se envía al monitor serial
  {
    Serial.write(BT.read());
  }

  if(Serial.available())  // Si llega un dato por el monitor serial se envía al puerto BT
  {
     BT.write(Serial.read());
  }
}
```

Figura 18. Sketch desarrollado en el IDE para establecer comunicación entre la PC y el módulo Bluetooth a través de un microcontrolador Arduino.

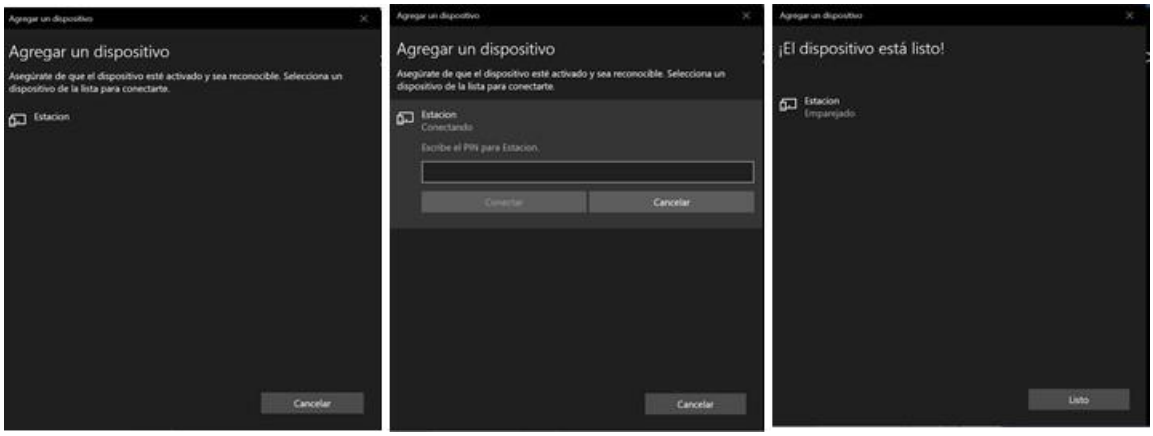

Figura 19. Vinculación del Módulo Bluetooth con la PC.

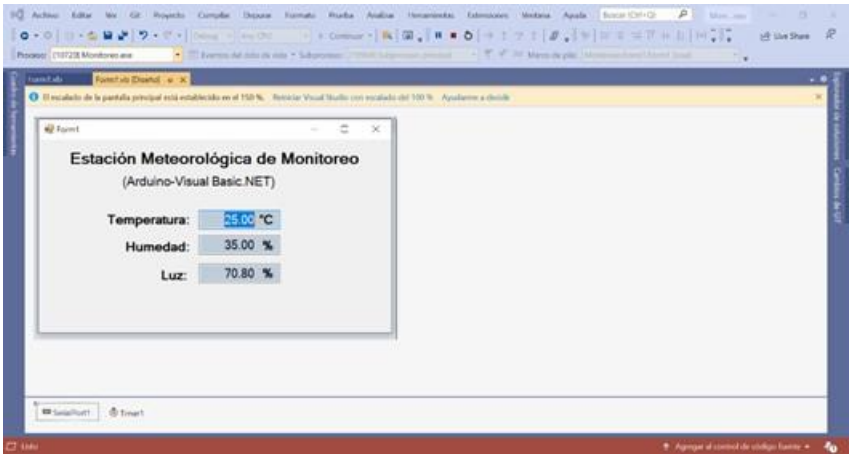

Figura 20. Ajuste del control SerialPort para establecer conexión con el módulo BT.

Conclusiones

Existen plataformas de hardware libre, como lo es el Arduino, una placa con un controlador y un entorno de desarrollo, su sencillez y bajo costo nos permiten hacer múltiples diseños y múltiples tipos de uso de éste. Recurrimos al uso del Arduino puesto que permite realizar muchas tareas, ahorrando elementos electrónicos y en sustitución, el uso de un lenguaje de programación, lo que hace que sea de muy fácil uso, con la combinación de la plataforma de Tinkercad fue más fácil trabajar a distancia, realizando las combinaciones adecuadas de cada sensor, investigaciones en la toma de mediciones por medio de sensores, siendo de código abierto, permitiendo la evolución constante y enfoques múltiples en pro de solucionar problemas diversamente planteados pero para que funcione se necesita tener un programa listo con los comandos o mejor dicho instrucciones que debe seguir el Arduino para que cumpla con su función correctamente, este programa escrito con un lenguaje especifico y se deberá pasar a la placa Arduino mediante los diversos puentes que esta placa posee. Existen diversos sensores que se pueden conectar a los puertos de la placa, en este artículo se mencionaron los sensores de temperatura, sensor de luz, de sonido, de fuerza y también el de inclinación, todos mandando información que pueda leer el Arduino y sea capaz de manifestarlo en un dispositivo externo, la facilidad que tiene están placa para ser capaz de leer todos esos datos y convertirlos en algo que podamos leer nosotros es lo que hace popular al Arduino, ya que no se trata de una placa simple que solamente puede hacer acciones simples sino que tiene la flexibilidad de utilizarse para programas complejos, con todo esto podríamos decir que la placa Arduino nos va a permitir hacer infinidad de proyectos y aprender sobre programación en el camino.

Referencias

(1) Extraído a partir de: https://www.arduino.cc/en/Guide/Introduction

(2) Extraído a partir de: https://hetpro-store.com/TUTORIALES/lm35/

(3) Extraído a partir de: https://naylampmechatronics.com/sensores-proximidad/10-sensor-ultrasonido-hc-sr04.html

(4) Extraído a partir de: https://sandorobotics.com/producto/hs1149/

(5) Extraído a partir de: https://prometec.mx/producto/ldr-sensor-de-luz/

(6) Extraído a partir de: https://www.luisllamas.es/medir-inclinacion-con-arduino-y-sensor-tilt-sw-520d/

(7) Extraído a partir de: https://formacion.intef.es/catalogo/mod/book/view.php?id=69&chapterid=361