SENSOR DE TEMPERATURA AMBIENTAL CON DISPOSITIVO ARDUINO Y LENGUAJE ENSAMBLADOR

Antonio Guadalupe Cruz Bautista
agcb10ster@gmail.com,
I. T. Galván-García
isis.sist@gmail.com
Centro de Innovación y Desarrollo Tecnológico en Cómputo I.P.N.

Abstract

El objetivo de este trabajo es obtener la temperatura ambiental con un sensor LM35; en esta lectura, la cual es señal analógica, se usa el Arduino como auxiliar para realizar la conversión analógica a digital y posteriormente mandar esa señal digital hacia el puerto paralelo de una computadora. Esta señal se lee por la computadora a través de un programa hecho en lenguaje ensamblador utilizando el debug de Windows el cual está preinstalado en Windows 7 de 32 bits y versiones anteriores, no así en la versión de 64 bits, y muestra la temperatura en la ventana de símbolo de sistema de Windows.

Introducción

La medición de la temperatura ha sido una de las mediciones más utilizadas, desde hogares hasta fábricas cuentan con termómetros o sensores de temperatura para controlar diversos procesos industriales, o para salvaguardar la integridad física. El LM35 es uno de los sensores de temperatura más usados para desarrollos escolares y algunos usos comerciales debido a su bajo costo, y la relación que tiene este costo con su precisión, durabilidad, resistencia y rango de temperatura. Se han hecho una gran variedad de aplicaciones y circuitos utilizando el LM35 como sensor de temperatura, y al utilizarse con Arduino, que es una plataforma libre para la realización de prototipos electrónicos, se obtienen funcionalidades variadas y de potencial para aplicaciones cada vez más amplias. En la Figura 1 se tiene la configuración de los pines del puerto paralelo, y como se aprecia en dicho esquema, los pines del 2 al 9 corresponden a los datos, y por medio de estos se pueden enviar y recibir información a través del puerto paralelo. Se conectan los pines del 18 a 25 a tierra y adicionalmente se hace uso del pin 10 para enviar una señal por medio de un botón cuyo fin es dar por terminado el programa de ensamblador.

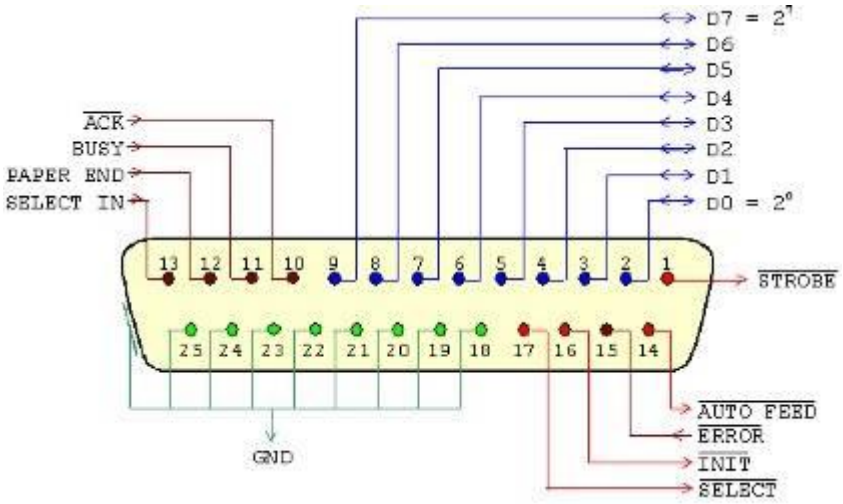

Figura 1. Configuración de los pines del puerto paralelo

Cabe mencionar que los pines de datos del puerto paralelo siempre están configurados por defecto para sólo escritura de datos utilizando la computadora y lo que hay que hacer para que sea posible leer y recibir datos por el puerto paralelo, es establecerlo como bidireccional, esto es, a través del BIOS de la computadora como se muestra en la figura 2, en la cual se puede apreciar la imagen del BIOS de la computadora con la configuración correcta.

Figura 2. El modo por defecto que trae la computadora utilizada es “STD Printer Mode”. Se cambió por “EPP-1.9 and SPP Mode”

Existen 3 modos diferentes en que puede estar configurado un puerto paralelo, estos son SPP (Standard Parallel Port), EPP (Enhanted Parallel Port) o ECP (Extended Capabilities Port). El SPP es el

modo bidireccional ofrecido por IBM alrededor del año 1987, el cual permite al dispositivo recibir y transmitir datos. El EPP fue creado por Intel y permite una transmisión mayor de datos por segundo que el SPP. La transmisión de datos del EPP varía de los 500 kilo-bytes hasta los 2 mega-bytes por segundo. El ECP fue creado principalmente para mejorar la velocidad y funcionalidad para las impresoras.

En la Figura 3 se observa el código en Arduino que toma la entrada del sensor en A0, y la convierte en señal de 8 bits para la salida para el puerto paralelo, y cuenta con una entrada para un botón que funciona para finalizar el programa de ensamblador.

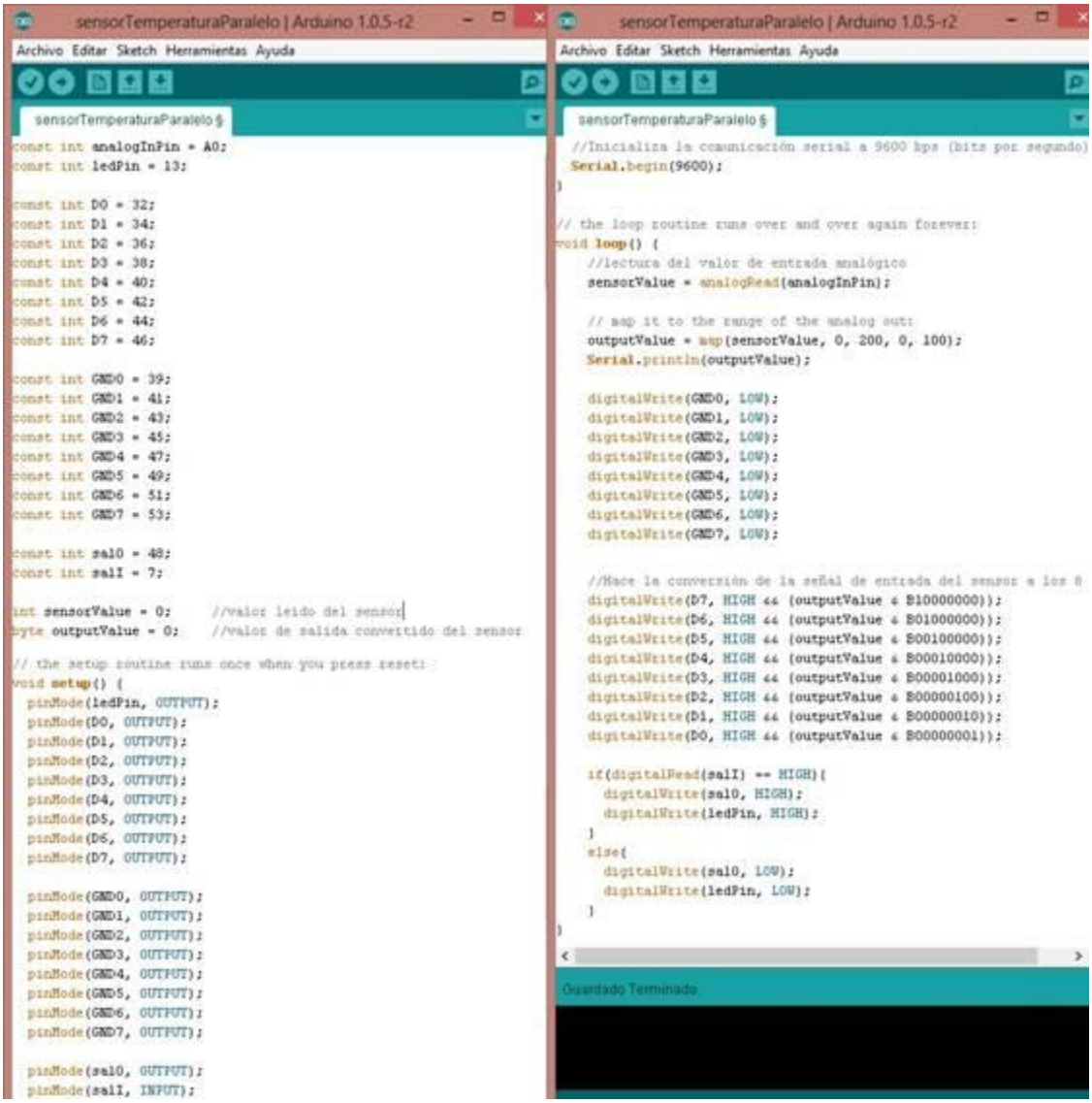

Figura 3. Código en Arduino

Para la elaboración del circuito se requiere un sensor de temperatura LM35 que es el que mide la temperatura ambiental. Fue instalado un botón que envía una señal a la computadora para salga del programa de ensamblador. En la Figura 4 se observa el circuito con sus respectivas conexiones en la protoboard y en Arduino.

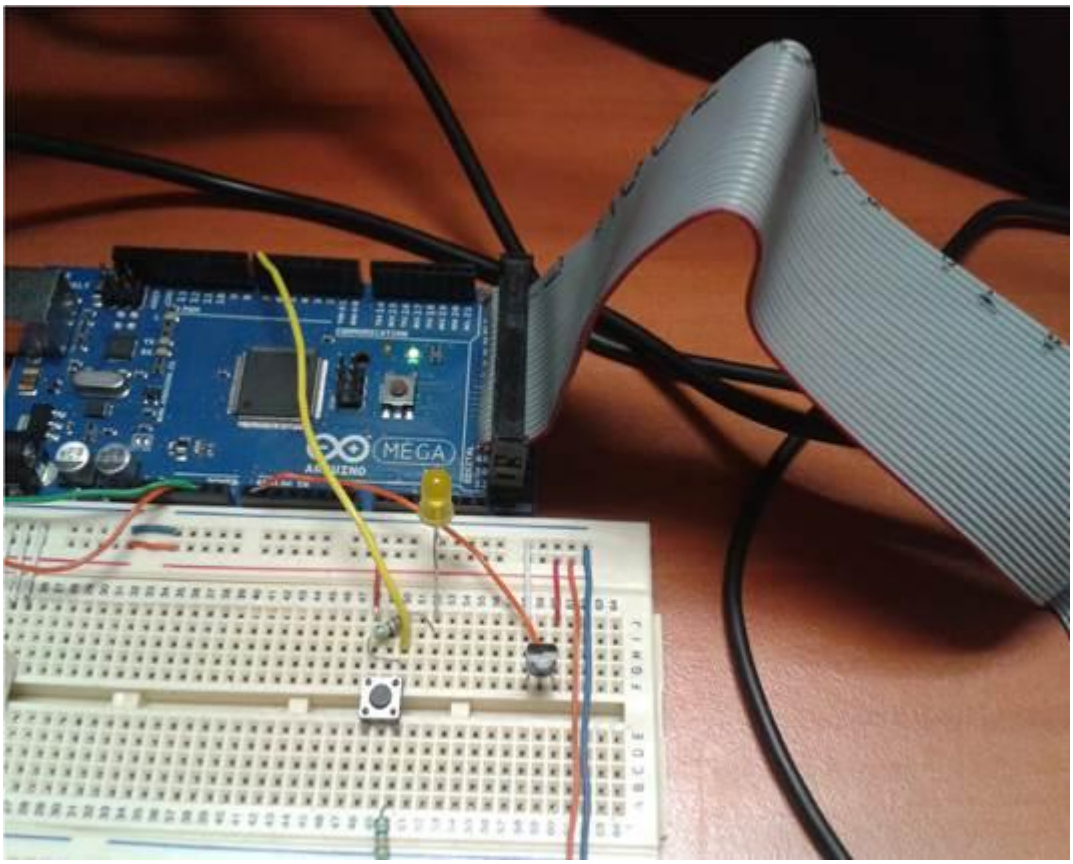

Figura 4. Circuito con sus conexiones

Para realizar el programa en ensamblador antes se debe conocer la dirección del puerto por el que va a ser transmitida la información, generalmente para los puertos paralelo son las direcciones 378 y 278. En este caso es 378. En caso de que se desconozca la dirección del puerto, desde símbolo de sistema en el debug se puede conocer escribiendo la instrucción d 40:0. A su vez, es necesario conocer las instrucciones de entrada y salida para puertos, estas son in y out. Las primeras líneas del programa en ensamblador son las que le indican a la computadora que se van a utilizar los pines de datos como lectura, esto se hace a través del puerto 37a y escribiendo un 20, para lectura de los datos se puede escribir df. Para la lectura del botón se utiliza el puerto 379 y simplemente se hace una comparación, en caso de que se cumpla la condición de que el botón esté presionado, se saldrá del programa en ensamblador, esto se puede apreciar en las últimas líneas de código de la Figura 5, donde se muestra el código en ensamblador.

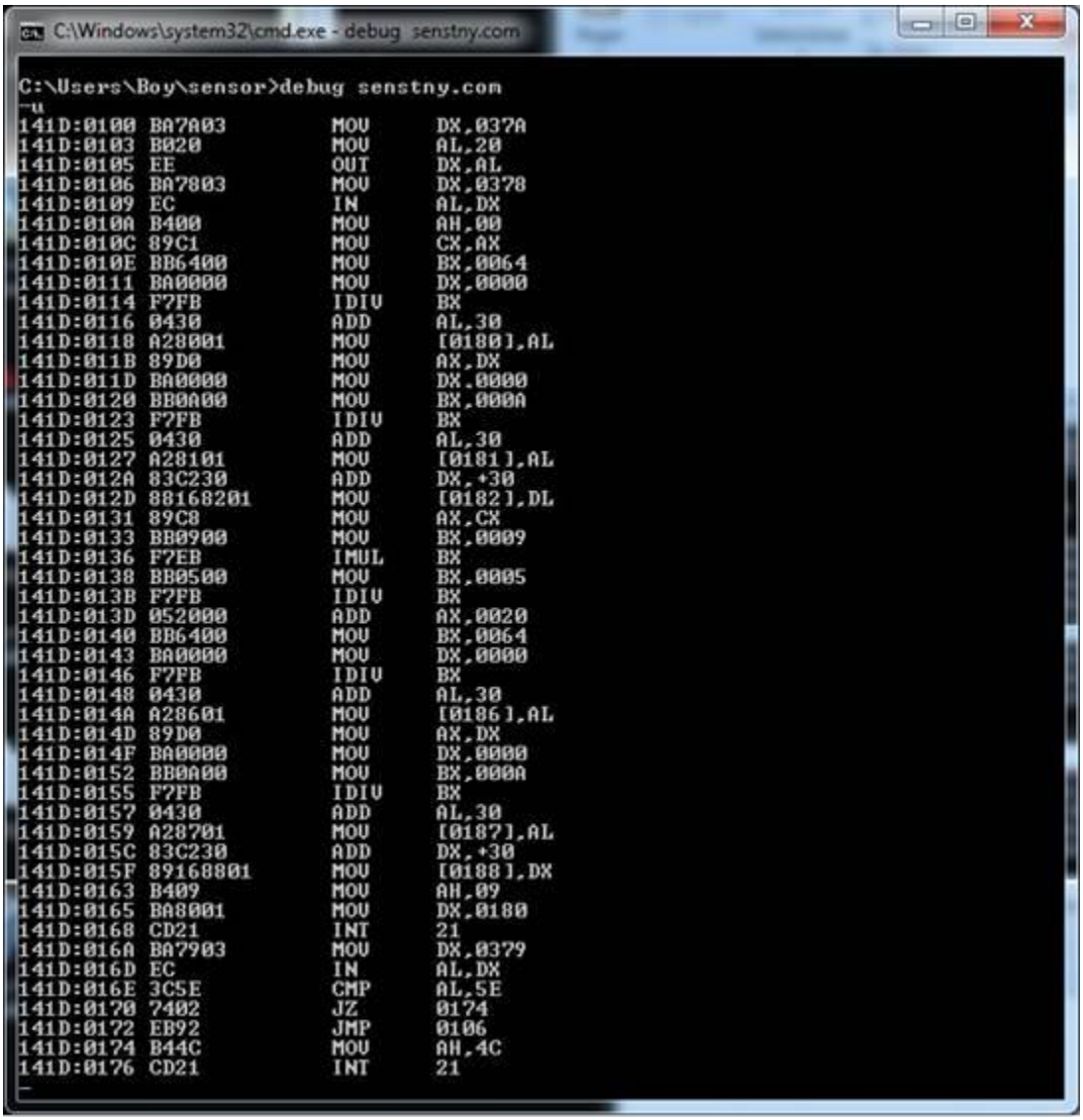

Figura 5. Código en lenguaje ensamblador que realiza la lectura del puerto paralelo

En el programa de ensamblador se muestra la temperatura en grados Centígrados y en grados Fahrenheit. Lo que se obtiene en lenguaje ensamblador está en código hexadecimal, se hace la conversión de hexadecimal a decimal para que muestre de la manera deseada la temperatura, esto es, en decimal. Para la conversión se realizaron divisiones entre 100 para las centenas, entre 10 para las decenas y el residuo corresponde a las unidades. Posteriormente, para la conversión a grados Fahrenheit se multiplica por 9, se divide entre 5 y finalmente se suma 32 (se suma 20 puesto que está en hexadecimal).En la Figura 6 se observa el programa en ejecución.

Figura 6. Programa en ejecución

Si se desea ver el video del programa en funcionamiento ir a este enlace: http://youtu.be/SmuoQW6zMow.

Conclusiones

Se obtuvieron resultados satisfactorios en la transmisión de los datos por el puerto paralelo. Al utilizar Arduino se ahorra mucho tiempo de ensamblado de componentes en la protoboard, así como el cableado y los posibles errores comunes de la manipulación de dispositivos que puedan presentarse.

Referencias

•http://platea.pntic.mec.es/vgonzale/cyr_0204/cyr_01/control/puerto_paralelo.htm

•http://computer.howstuffworks.com/parallel-port2.htm

•http://www.scribd.com/doc/9408080/Parallel-Port

•http://www.s-w-r.com/Ports2.html

•http://en.wikipedia.org/wiki/Input/output_base_address Fecha de consulta del 1 de Abril del 2014.