SISTEMAS EMBEBIDOS
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Segundo Cuatrimestre de 2019

Laboratorio N° 4
Comunicación en sistemas embebidos

Fecha de evaluación: Lunes 07/10/19
Objetivo: El objetivo de este laboratorio consiste en introducir a los alumnos en la comunicación entre sistemas embebidos, principalmente en el diseño e implementación de protocolos de comunicación, y en el uso de los estándares I^2C y Ethernet en Sistemas Embebidos.
Desarrollo: El laboratorio deberá realizarse en comisiones de no más de 2 alumnos. Al finalizar la evaluación, se deberá comprimir y enviarse por mail respetando el siguiente formato: *LaboratorioX-ApellidosComision.zip*.

Descripción del hardware

El Hardware a utilizar se compone de:

- una placa Arduino Uno [1] con un microcontrolador ATmega328P [2].
- Una placa Intel Galileo 1st Gen [3].
- protoboard y cables para realizar el conexionado.
- una resistencia entre 75 y 120 ohms para la conexión del sensor (opcional).
- un capacitor de 1uF para la conexión del sensor (opcional).
- un sensor de temperatura lm35 conectado en modo de sensado básico, con un damper R-C, tal y como se muestra en la Figura 4 de la hoja de datos (ver [4]).
- Cables para implementar un bus I^2C, y cables de red para lograr la conectividad de las placas con el host.

Actividad 1: Introducción a Intel Galileo

1. Conectar la placa Intel Galileo por Ethernet a la red. Conectar la alimentación y encenderlo. Utilizando PuTTY o algún otro cliente, acceda por SSH, sabiendo que la dirección del Galileo es `172.16.30.<nro>`, donde `<nro>` es el número de placa. Experimente el entorno Linux utilizando comandos estándar como `top`, `ls`, `cd`, etc.
2. Abrir el entorno de trabajo Intel System Studio IoT.
3. Crear el proyecto `blink`, siguiendo la guía presentado por la cátedra. Ejecutarlo desde Intel System Studio.

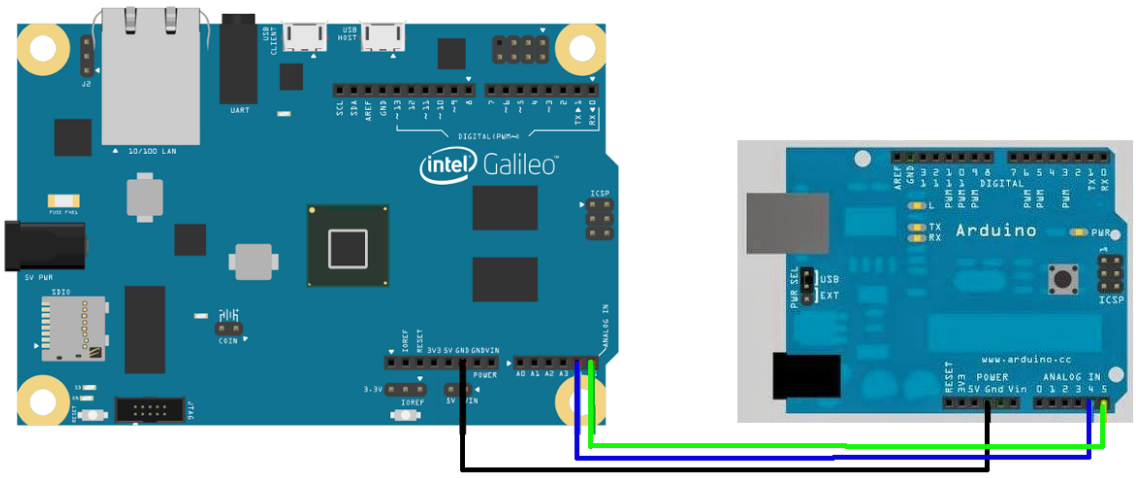
4. Ejecutar el mismo programa generado, accediendo por SSH al sistema en Galileo, sin utilizar la IDE.

5. Utilizando el IDE Arduino, analizar el scketch de ejemplo Wire >`slave_send`.

6. Descargar del sitio web de la materia el ejemplo de software para Galileo, que comunica con I^{2}C. Abrir el ejemplo utilizando el IDE Intel System Studio. Analizar el ejemplo y familiarizarse con el uso de la biblioteca mraa.

7. Conectar Arduino y Galileo mediante I^{2}C como indica la figura 1. Descargar en Arduino el ejemplo `slave_send`, y correr en Galileo el ejemplo provisto por la materia `master_receiver`. Observar la salida por consola provista por `master_receiver`.

Figura 1: Circuito Arduino y Galileo I^{2}C



Actividad 2: Mediciones de temperatura por I^{2}C

Se desea obtener desde Galileo, las medidas de temperatura provistas por el sistema implementado en el Laboratorio 3. Para la comunicación entre ambas placas, se utilizará el bus I^{2}C.

La placa Galileo actuará como *maestro* en la comunicación, y la placa Arduino como *esclavo*.

Galileo podrá solicitar a Arduino la temperatura actual, la máxima, la mínima o la promedio calculada hasta el momento.

1. Diseñar en papel, o utilizando herramientas electrónicas de dibujo o procesamiento de textos, el protocolo que realizará la comunicación entre Arduino y Galileo. Este protocolo debe:

   - Armar un paquete que contenga un campo con el tamaño total, un payload de tamaño variable (puede ser de tamaño cero), y un campo que indique el tipo de mensaje.
   - Definir un símbolo que indica el comienzo, un símbolo que indica el fin y un símbolo que permite hacer escape de los símbolos de fin y comienzo.

- Los tipos de mensaje soportados por el protocolo pueden ser OBTENER_TEMP, OBTENER_MAX, OBTENER_MIN, OBTENER_PROM, OBTENER_TODO, RESPONDER_TEMP, RESPONDER_MAX, RESPONDER_MIN, RESPONDER_PROM, RESPONDER_TODO. Es posible agregar otros tipos de mensaje si considera necesario hacerlo.

2. ¿Qué beneficios tiene indicar el comienzo y el fin de un mensaje? ¿Por qué es necesario un símbolo especial de escape?

3. Modificar el sistema para Arduino desarrollado en el Laboratorio 3, para que funcionando como *esclavo* del bus I^2C, con un ID de I^2C igual al que tiene la placa Arduino utilizada, reciba los mensajes OBTENER_TEMP, OBTENER_MAX, OBTENER_MIN, OBTENER_PROM, OBTENER_TODO, y responda respectivamente con los mensajes RESPONDER_TEMP, conteniendo la información de luminosidad actual; RESPONDER_MAX y RESPONDER_MIN, conteniendo la información de luminosidad máxima y mínima registradas; RESPONDER_PROM, con la luminosidad promedio; y RESPONDER_TODO, que contiene temperatura actual, mínima, máxima y promedio.

4. Escribir un programa en el entorno de Intel Studio para la placa Galileo, que se comunique como *maestro* por I^2C con Arduino. Este programa debe funcionar como programa de consola. Debe aceptar opciones en la línea de comandos parar solicitar la temperatura actual, temperatura máxima y mínima, y temperatura promedio a Arduino. Una vez recibida la respuesta de Arduino, debe mostrarla por la salida estándar de la consola.

5. Suponiendo que el sistema ejecutando en Arduino no necesite mostrar por pantalla datos de temperatura ¿Es conveniente que Arduino envíe la información en temperatura, o quizá otra medida, como los ticks de ADC? Justificar.

Referencias

[1] Arduino Uno WebSite. `https://arduino.cc/en/Main/arduinoBoardUno`.

[2] Atmel AVR ATmega48A/48PA/88A/88PA/168A/168PA/328/328P Data Sheet.

[3] Intel Galileo 1 Gen.https://www.intel.com/content/www/us/en/support/articles/000005735/boards-and-kits/intel-galileo-boards.html.

[4] LM35 Precision Centigrade Temperature Sensors Data Sheet.