Sistemas Embebidos
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Segundo Cuatrimestre de 2017

Laboratorio N° 3
Comunicación en Sistemas Embebidos

Fecha límite de presentación: Miércoles 18 de Octubre 17:00 hs.
Fecha de evaluación: Viernes 20 de Octubre.

Objetivo: Los objetivos del laboratorio consisten en:

- Comunicación de sistemas embebidos.
- Diseño e implementación de protocolos.
- I2C y Ethernet en sistemas embebidos.

Desarrollo: El laboratorio deberá realizarse en comisiones de no más de 2 alumnos.

Actividad 1: Introducción a Intel Galileo

1. Conectar la placa Intel Galileo [2] 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 [1]. 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^2C. 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^2C 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`.

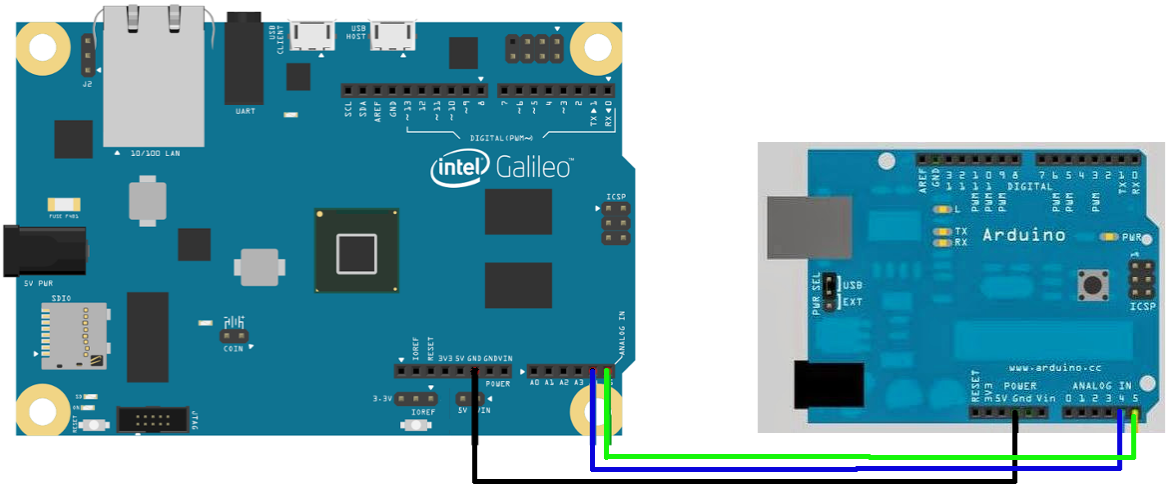
Actividad 2: Mediciones de luminosidad por I^2C

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

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

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

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



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_LUX, OBTENER_MAX, OBTENER_MIN, OBTENER_PROM, OBTENER_TODO, RESPONDER_LUX, 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 2, para que funcionando como *esclavo* del bus I^{2}C, con un ID de I^{2}C igual al que tiene la placa Arduino utilizada, reciba los mensajes OBTENER_LUX, OBTENER_MAX, OBTENER_MIN, OBTENER_PROM, OBTENER_TODO, y responda respectivamente con los mensajes RESPONDER_LUX, 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 luminosidad 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^{2}C con Arduino. Este programa debe funcionar como programa de consola. Debe aceptar opciones en la línea de comandos parar solicitar luminosidad actual, luminosidad máxima y mínima, y luminosidad 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 luminosidad ¿Es conveniente que Arduino envíe la información en lux, o quizá otra medida, como los ticks de ADC? Justificar.

Actividad 3: Driver virtual de teclado

Se desea añadir a los sistemas implementados para Arduino y Galileo, la posibilidad de simular la operación de teclas y pulsador desde la aplicación de consola de Galileo.

1. Modificar el diseño del protocolo implementado en la actividad anterior, para que acepte enviar desde el *maestro* al *esclavo* los nuevos comandos de TECLA_UP, TECLA_DOWN, TECLA_LEFT, TECLA_RIGHT y BOTON_A2.

2. Modificar el sistema en Arduino para que interprete los nuevos comandos a recibir por I^2C. Estos comandos deben generar en el programa el mismo efecto que tendría pulsar TECLA_UP, TECLA_DOWN, TECLA_LEFT, TECLA_RIGHT y BOTON_A2 en el LCD Keypad Shield. Para ello deberá modificar el driver de teclado implementado añadiendo soporte para la abstracción del dispositivo utilizado en las comunicaciones por I^2C. ¿Qué ventaja presenta contar con una capa de drivers en relación a la modificación realizada?

3. Modificar el programa para Galileo implementado en la actividad anterior, para que al recibir en la ejecución de línea de comandos up, down, left, right o boton, envíe respetando el nuevo protocolo los comandos por I2C TECLA_UP, TECLA_DOWN, TECLA_LEFT, TECLA_RIGHT y BOTON_A2 respectivamente.

4. Ejecutar los programas modificados y probar el sistema.

Actividad 4: Interfaz con una PC

El objetivo de esta actividad es implementar en una PC un panel de control, que utilizando comunicación a través de una red TCP/IP, solicite a la placa Galileo información de luminosidad. El panel de control debe proveer al usuario de una gráfica de las fluctuaciones producidas en la luminosidad actual, promedio, máxima y mínima. Como modelo de referencia, se ofrece junto con este laboratorio el ejemplo el ejemplo SE_Lab3_HostLuxApp [4] conteniendo componentes para la implementación de la interface del panel de control, implementado utilizando el scketchbook y lenguaje Processing.

La placa Galileo deberá obtener estos datos valiéndose de su conexión I^2C con Arduino, utilizando los lineamientos establecidos en las actividades anteriores. En la placa Arduino se deberá ejecutar el sistema implementado en la actividad anterior, ofreciendo la posibilidad de controlar el modo de operación de Arduino desde la interface en el host (a través de las modificaciones realizadas en la actividad anterior).

El software de la placa Galileo deberá ejecutar como un servicio, abriendo un puerto de comunicación y esperando las solicitudes de un cliente remoto (i.e. el host) a través de la red.

1. Estudiar y familiarizarse con la biblioteca Network [3] de Processing.

2. Descargar y abrir el ejemplo SE_Lab_HostLuxApp [4] de Processing disponible en la página web de la materia. Examinar y familiarizarse con el funcionamiento del mismo.

3. Utilizando como base el programa de consola implementado en la actividad anterior, escribir un servicio a ejecutar en Galileo, que abra un socket y espere recibir comandos a través de este. Los comandos son solicitar luminosidad actual, luminosidad máxima y mínima, luminosidad promedio, up, down, left, right o boton. Estos comandos recibidos a través de la red tienen el mismo efecto que los implementados en las actividades anteriores.

4. Siguiendo el ejemplo SE_Lab_HostLuxApp, escribir un sckech de processing que se conecte al servicio en Galileo implementado en el punto anterior. El scketch debe, utilizando los comandos provistos por el servicio, obtener las medidas de luminosidad actual, máxima, mínima y promedio para graficar sus fluctuaciones en el tiempo; y proveer al usuario de botones o algún otro

mecanismo para controlar a Arduino como si se pulsaran las teclas up, down, left, right o boton A2 del LCD Keypad Shield.

5. Implementar el sistema, construir y depurar el proyecto.

Referencias

[1] BlinkingALedUsingC/C++. https://software.intel.com/en-us/blinking-an-led-using-c-with-intel-system-studio-iot-edition/.

[2] Intel Galileo 1st Gen. https://www.arduino.cc/en/ArduinoCertified/IntelGalileo.

[3] Processing Network library. http://processing.org/reference/libraries/net.

[4] SE Lab3 HostLuxApp. http://cs.uns.edu.ar/materias/se/index.php?accion=download.