UNLaM

Departamento de Ingeniería e Investigaciones Tecnológicas

Sistemas Operativos Avanzados

Internet of Things

Sistemas Embebidos + Android

“Sistema Embebido de Análisis de Lluvia”

SEAL

1er cuatrimestre – Año 2017

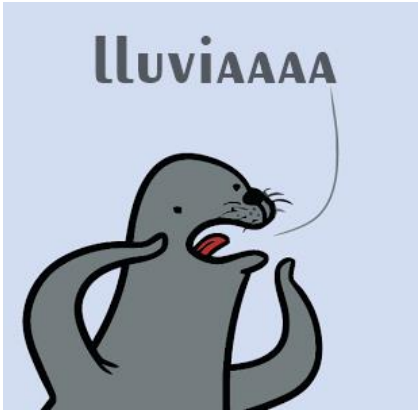
Integrantes:

Coria, Santiago

Greco, Maximiliano

Minardi, Matías

Piemontese, Gerardo



Contenido

1. Objetivos del Trabajo Práctico

Desarrollar los conocimientos y experiencias adoptados en la cursada acerca de Sistemas Embebidos, IoT y Android con el fin de implementar un sistema capaz de medir la turbiedad del agua de lluvia y determinar en base a las partículas suspendidas en el agua, si la misma es apta para consumo o es destinada para riego.

1. Entorno de Trabajo

1.1 Hardware

1.1.1 Sistema Embebido

- Notebook Lenovo G580 (Intel Core i5 – 4 GB RAM – Windows 10)
- Arduino Mega 2560
- Cables para la conexión entre placa y sensores
- 1 x Placa de desarrollo
- 1 x Sensor digital de temperatura sumergible DS18B20
- 1 x Sensor digital de ultrasonido HC-SR04
- 1 x Sensor analógico/digital de turbidez DFSEN0189 (usamos el modo analógico)
- 1 x LCD Module 20x4
- 1 x Interfaz I2C/SPI para LCD
- 1 x Resistencia 1/4W 4K7 Ohm para sensor de temperatura
- 1 x Módulo bluetooth HC-06
- 1 x Módulo de 2 relés de 5v EARE00102SL
- 2 x Bombas de agua

1.1.2 Aplicación Android

- DELL Inspiron (Intel Core i7 -16 GB RAM – Windows 10)
- LG 4 PLUS – Android 6.0.1
- Sensor Acelerómetro
- Sensor de Luz
- Vibrador del dispositivo

1.2 Software

1.2.1 Sistema Embebido

- Arduino IDE 1.8.2
- Windows 10 Home Single Language x64

1.2.2 Aplicación Android

- Android Studio versión 2.3.2

2. Alcance del Sistema

2.1 Sistema Embebido

2.1.1 El sistema embebido debe realizar las siguientes mediciones:

- Turbiedad del agua en base a las partículas suspendidas
- Temperatura del agua
- Nivel/Volumen de agua en litros.

2.1.2 El sistema embebido debe enviar constantemente las mediciones realizadas por medio de la conexión bluetooth hacia la aplicación Android y también mostrar dichos valores en la pantalla LCD.

2.1.3 El sistema embebido debe contemplar niveles máximos y mínimos de agua para que el nivel de agua no supere el máximo de 3,8 litros ni sea menor a 1,2 litros y el agua quede por debajo de las bombas de drenaje.

2.1.4 El sistema embebido debe analizar la turbiedad del agua cuando el nivel de la misma supere 3,8 litros.

2.1.5 El sistema debe realizar el drenaje del agua luego de analizar la turbiedad de la misma hasta llegar a un nivel tope de 1,2 litros.

2.1.6 El sistema embebido debe decidir qué bomba encender en base a la turbiedad del agua. Si la turbiedad es menor o igual a 2 NTU se debe encender la bomba correspondiente a “Agua para Consumo”. Si la turbiedad es mayor a 2 NTU se debe encender la bomba correspondiente a “Agua para Riego”.

2.1.7 El sistema embebido debe enviar constantemente, la información obtenida por medio de los sensores, a la aplicación por medio de una conexión bluetooth.

2.1.8 El sistema embebido debe reaccionar ante las instrucciones enviadas por medio de la aplicación.

2.1.9 El sistema embebido debe admitir dos modos de operación (automático y manual) determinado por el usuario de la aplicación.

2.1.10 El sistema embebido debe hacer parpadear el display LCD al momento de recibir una señal especifica de la aplicación Android.

2.2 Aplicación Android

2.2.1 La aplicación Android debe recibir los valores enviado por el sistema embebido, de temperatura, turbiedad y ultrasonido por medio de una conexión bluetooth y mostrarlo constantemente por medio de la interfaz.

2.2.2 La aplicación Android debe permitir cambiar el modo de operación del sistema embebido pudiendo elegir entre el modo manual y el modo automático.

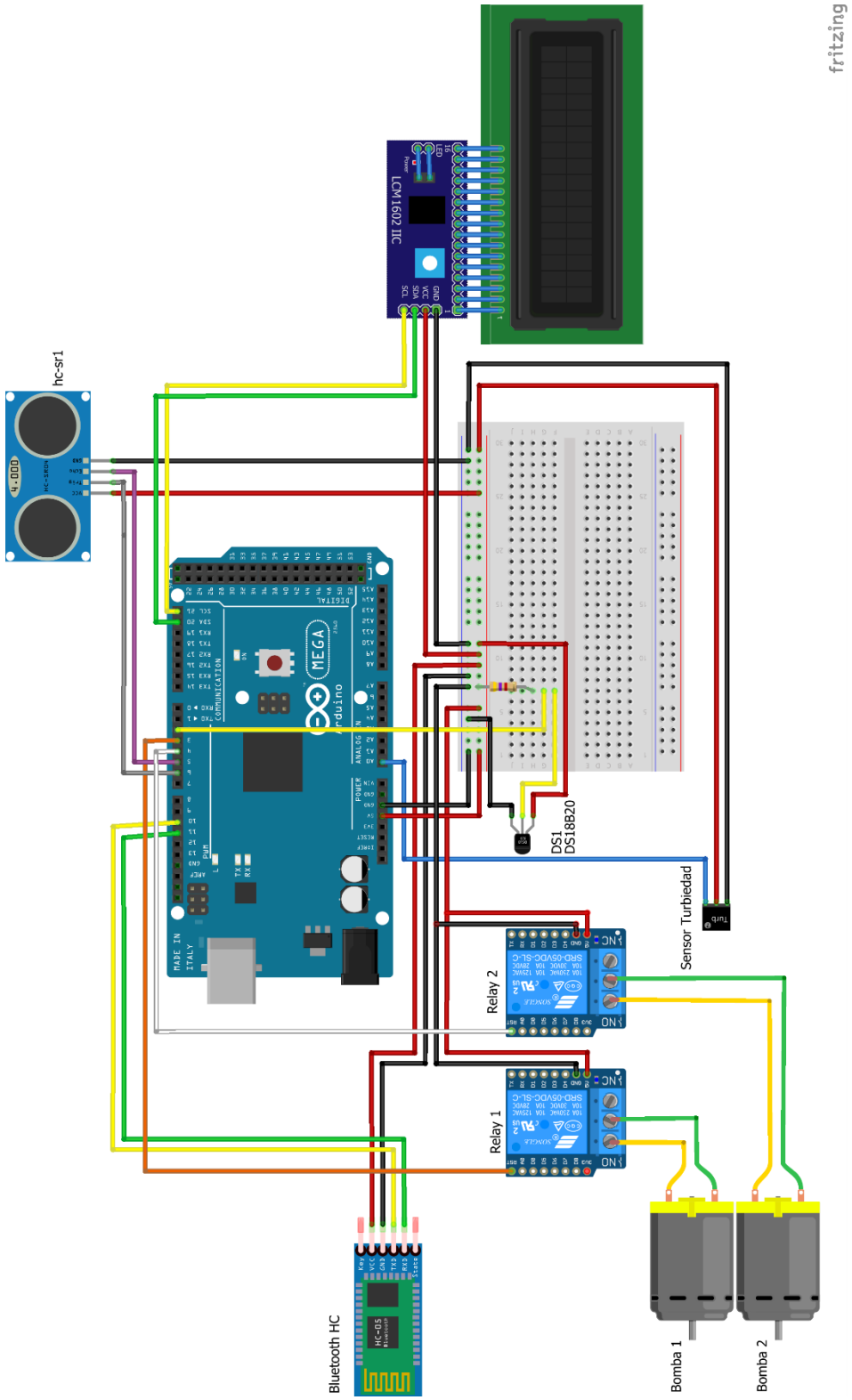
- *Modo Automático*: El sistema embebido lee constantemente el nivel de agua.
  Al llegar a la cantidad de 3,8 litros, el sistema embebido analiza la turbiedad y decide si destina el agua a riego o a consumo abriendo una u otra bomba, pudiendo así drenar el tanque principal hasta un nivel mínimo de 1.2 litros para no quedar por debajo de las bombas de drenaje.
- *Modo Manual:* Al estar en modo manual, el drenaje de las bombas se realizará en base a lo que seleccione el usuario por medio de la aplicación. El mismo tendrá la opción de abrir la bomba destinada a agua para consumo o aquella destinada para riego independientemente de la turbiedad del agua. Si bien el control será manual el sistema tendrá un corte automático para que el agua no quede en un nivel por debajo de las bombas y dañe el sistema.

2.2.3 La aplicación Android debe implementar el sensor acelerómetro para determinar qué bomba se debe encender en base a la inclinación hacia la izquierda (bomba de riego) o derecha (bomba de consumo) del dispositivo.

2.2.4 La aplicación Android debe accionar el vibrador del dispositivo en caso que se accione una u otra bomba de drenaje y el modo de operación sea manual.

2.2.5 La aplicación Android debe enviar una señal al sistema embebido cuando detecte un nivel de luz menor al 3 lumenes.

3. Diagrama General del Sistema



El sensor principal de nuestro proyecto es el sensor de TURBIEDAD el cual posee un switch para cambiar de modo analógico a digital o viceversa.
Este sensor utiliza la luz para detectar partículas en suspensión en el agua mediante la medición de la transmitancia de luz y la velocidad, que cambia con la cantidad de sólidos suspendidos totales (SST) en el agua. A medida que aumenta el SST, aumenta el nivel de turbidez.

Luego en base a las SST devueltos el sensor devuelve un valor que varia entre 0 a 1023. Se debe calcular el voltaje correspondiente a dicha medición mediante la siguiente fórmula:

$$Voltaje = SST\ x\ (\frac{5.0}{1024})$$

Una vez obtenido el voltaje se calcula los NTU (Unidad Nefelometrica de Turbidez) equivalentes mediante la siguiente fórmula:

$$NTU = -1120.4\ x\ Voltaje + 5742.3\ x\ Voltaje - 4352.9$$

Si bien esa es la forma que indica el fabricante para calcular los NTU del agua, cuando realizamos las pruebas del sensor, obtuvimos valores fuera de dichos márgenes. Sumado a esto, las mediciones tomadas se veían afectadas a condiciones externas como por ejemplo la luz ambiental. Por lo cual, decidimos adaptar dicho sensor para que se vea afectado lo mínimo posible por estas condiciones colocándole un capuchón que disminuya la incidencia de la luz.

Luego de estas modificaciones, realizamos nuevas mediciones y adoptamos el siguiente criterio para obtener los valores de turbiedad en NTU:

Valor informado por el Sensor (VS)	Valor correspondiente a NTU	Informe del Agua
VS >= 780	1	Apta para consumo
760 <= VS < 780	1.5	Apta para consumo
745 <= VS < 760	2	Apta para consumo
VS < 745	-(166/15) x VS + (24740/3)	Disponible para Riego

4. Implementación

4.1 Sistema Embebido

4.1.1 Display LCD

Para la implementación del LCD se deben agregar las librerías **Wire.h** y **LiquidCrystal_I2C.h** para la utilización de la interfaz I2C. Una vez realizado esto declaramos los parámetros de escritura del LCD de la siguiente manera:

```
LiquidCrystal_I2C  lcd(0x27,20,4);
```

Finalmente inicializamos el LCD y encendemos el backlight mediante las funciones **lcd.init()** y **lcd.backlight().**

De esta manera el LCD queda disponible para escribir por pantalla mediante las funciones **lcd.setCursor()** y **lcd.print()**

4.1.2 Sensor de temperatura sumergible

Para la implementación del sensor de temperatura se deben agregar las librerías **OneWire.h** y **DallasTemperature.h**. Luego debemos declarar el PIN utilizado para dicho sensor (nosotros utilizamos el pin digital 2) y finalmente declaramos el bus para la comunicación *OneWire*, instanciamos la biblioteca *DallasTemperature* e inicializamos el sensor mediante le siguiente código:

```
OneWire ourWire(pinTemp);
DallasTemperature sensorTempSumerg(&ourWire);
sensorTempSumerg.begin();
```

Finalmente obtenemos el valor de la temperatura mediante las siguientes llamadas a funciones:

```
sensorTempSumerg.requestTemperatures();
sensorTempSumerg.getTempCByIndex(0);
```

4.1.3 Sensor de Ultrasonido

Para la implementación del sensor de ultrasonido no se necesitan agregar librerías. Sólo se deben declarar los pines ECHO y TRIGGER. El pin ECHO de entrada determina el tiempo de rebote del ultrasonido y el pin TRIGGER de salida determina el pulso ultrasónico (nosotros utilizamos los pines 5 y 6):

```
pinMode(trigPin,OUTPUT);
pinMode(echoPin,INPUT);
```

Finalmente calculamos la distancia de la siguiente forma:

```
digitalWrite(trigPin, LOW);/*Estabilización del sensor*/
delayMicroseconds(2);
digitalWrite(trigPin, HIGH); /*Envío del pulso ultrasónico*/
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duracion = pulseIn(echoPin,HIGH);/* Función para medir la longitud del
pulso entrante. Mide el tiempo que transcurrido entre el
envío del pulso ultrasónico y cuando el sensor recibe el rebote */

distancia = (duracion/(float)2) / 29.1;
capacidad = -0.2656 * distancia + 4.7687;
```

4.1.4 Sensor de Turbiedad

Para la implementación del sensor de turbiedad tampoco es necesario agregar librerías. Simplemente se debe conectar el sensor a una de las entradas analógicas de la placa Arduino y realizar la lectura de dicho pin mediante la siguiente instrucción:

```
int sensorValue = analogRead(A0);
```

4.1.5 Relays para bombas

Para la utilización de los relays no es necesaria incluir librerías, pero se debe declarar los pines utilizados para ambos relays:

```
int pinRelay1 = 3;      //RELAY 1
int pinRelay2 = 4;      //RELAY 2
```

Al inicio del proceso, apagamos ambos relays mediante las siguientes instrucciones:

```
digitalWrite(pinRelay1,HIGH);
digitalWrite(pinRelay2,HIGH);
```

Finalmente en el proceso se deben abrir y cerrar las bombas mediante las instrucciones ***digitalWrite(pinRelay,LOW);*** o ***digitalWrite(pinRelay, HIGH);*** respectivamente

4.2 Aplicación Android

Al iniciar, la aplicación Android solicita encender el Bluetooth del dispositivo en caso de que el mismo no se encuentre conectado.
En la primera pantalla, el usuario debe seleccionar el Bluetooth dentro de la lista de dispositivos vinculados.

Una vez conectado, la aplicación Android redirecciona a la pantalla principal donde se visualizan los valores de temperatura, turbiedad y nivel de agua recibidos del sistema embebido. Además, da información sobre el modo de operación, los estados de las bombas de drenaje y la opción para habilitar la funcionalidad del acelerómetro.

Al cambiar el modo de operación a Manual, el usuario puede activar las bombas de drenaje si el nivel de agua es superior a 1,2 lts. haciendo click en los switchs o habilitando el sensor acelerómetro y luego, desplazando el celular de izquierda a derecha.

En cambio, el modo de operación Automático, únicamente permite la visualización de los valores e impide al usuario activar las bombas.

5. Comunicación Bluetooth

5.1 Arduino a Android

Para la utilización del bluetooth se debe incluir la siguiente librería:

```
#include <SoftwareSerial.h>
```

Luego declaramos los pines RX y TX donde conectamos el modulo HC-06 a la placa Arduino, con la siguiente instrucción:

```
SoftwareSerial BT(10,11);
```

Luego realizamos la lectura de los valores que enviamos desde el dispositivo Android mediante el siguiente Código:

```
 if(BT.available())    // Si llega un dato por el puerto BT se envía al monitor serial
 {
   msjin = LeerBT();
   Serial.print(msjin);
   if (msjin.equals("#FA~")){ArdCtl=1;}
   else if(msjin.equals("#FM~")){ArdCtl=0;}
 }
```

Donde LeerBt() es:

```
String LeerBT(){
 String msj;
 delay(10); //delay de contingencia, para cargar el buffer
//Cargo el buffer en una variable hasta el fin de linea
 while (BT.available()>0){
              msj+= char(BT.read());
              }
 return msj;
}
```

Por último, enviamos el string con un determinado formato que luego vamos a parsear y leer desde android mediante la siguiente función:

```
void EnviarBT(){
 String msjout;
 String ModoFun;
 String EstadoB1;
 String EstadoB2;

 // Armo la cadena de envío de valores de sensores al Bluetooth
 if (ArdCtl==1) ModoFun = "FA";
   else ModoFun = "FM";
 if (encendido1) EstadoB1 = "B11";
   else EstadoB1 = "B10";
 if (encendido2) EstadoB2 = "B21";
```

```
    else EstadoB2 = "B20";

  msjout = "#SE" + String(capacidad) + "|" + String(temperatura) + "|" +
String(ntu) + "|" + ModoFun + "|" + EstadoB1 + "|" + EstadoB2 + "~";
  char msjBT[msjout.length()+1];
  msjout.toCharArray(msjBT,msjout.length()+1);

  //Envio el mensaje al BT
  BT.write(msjBT);

}
```

5.2 Android a Arduino

La conexión se realiza por medio de dos clases que extienden de asyntacks. La primera (ConectBT) nos permite establecer la conexión, creando el socket Bluetooth que mantiene vinculado la aplicación Android con el sistema embebido. La segunda (LeerYEscribirBT) nos permite leer los datos recibidos por el Arduino y a través del método OnProgressUpdate(); actualizar el activity principal (valores y controles).

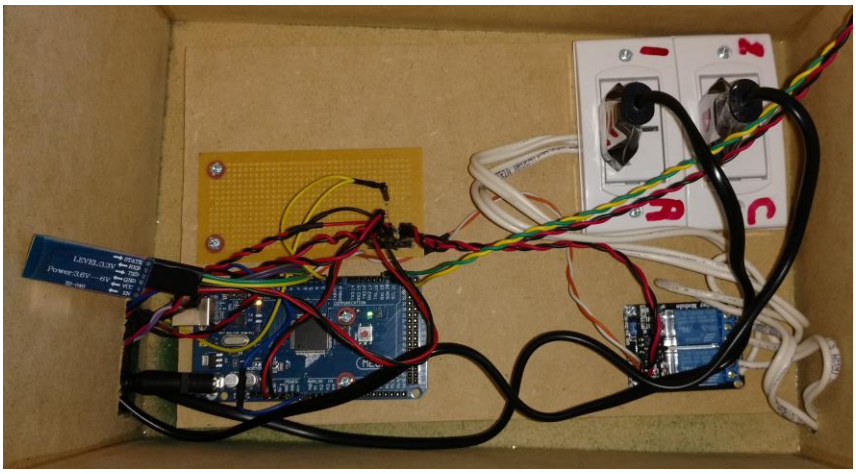
5.3 Formato del mensaje de Comunicación

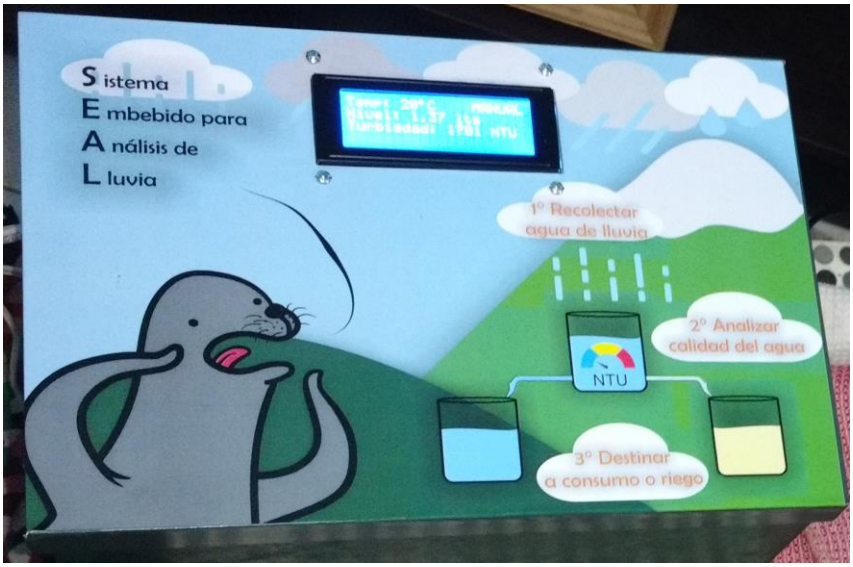
El mensaje de comunicación posee el siguiente formato:
#SECAP|TEMP|NTU|FUNC|EB1|EB2~
Donde:

- #SE: Inicio del mensaje
- CAP: Capacidad del tanque principal
- TEMP: Temperatura del agua
- NTU: Turbiedad del agua
- FUNC: Modo de funcionamiento del sistema, el cuál puede ser MANUAL(FM) o AUTOMATICO(FA)
- EB1: Estado de bomba 1, la cual puede estar encendida (B11) o apagada (B10)
- EB2: Estado de bomba 2, la cual puede estar encendida (B21) o apagada (B20)
- ~: Fin del mensaje.

Mensaje Adicional: #BLB~. Dicho mensaje lo utilizamos para hacer parpadear el LCD cuando la cantidad de luz indicada por el sensor de luz es menor a 3.

6. Proyecto Terminado





7. Webgrafía Utilizada

7.1 https://developer.android.com/

7.2 http://www.sgoliver.net/blog/tareas-en-segundo-plano-en-android-i-thread-y-asynctask/

7.3 http://www.androidcurso.com/index.php/tutoriales-android-basico/36-unidad-5-entradas-en-android-teclado-pantalla-tactil-y-sensores/365-ejecutar-una-tarea-en-un-nuevo-hilo-con-asynctask

7.4 https://es.stackoverflow.com/