



UNLaM

Departamento de Ingeniería e Investigaciones Tecnológicas

Sistemas Operativos Avanzados

Internet of Things

Sistemas Embebidos + Android

“Sistema Embebido de Enfriamiento de Líquidos”

SEAL 1er cuatrimestre – Año 2017 Integrantes:

Gutierrez, Cristian Rafael

Navarro, Leonardo Emanuel

Ramírez, Martín Ariel

## **Indice**

**1. Objetivos del Trabajo Práctico**

**2. Materiales utilizados**

**3 Alcance**

**4 Desarrollando el Sistema embebido**

**5 Activities**

**6 Comunicación**

**7 Código**

**8 Fuente**

### **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 enfriar un recipiente.

## **2. Materiales utilizados**

### Hardware

1 placa peltier

1 sensor de temperatura analógico

1 placa Arduino UNO

1 fuente de PC de 550watts

1 Gabinete de PC

1 Pote de helado (1kg)

2 pulsadores

1 disipador

1 módulo bluethoot HC05

2 relé 6volt

1 protoboard

### Cables para conectar

1 LCD 1602<sup>a</sup>

1 transistor

1 diodo

5 resistencias (3x 300 ohm, 2x 5komh )

### **3 Alcance**

#### **3.1 Sistema embebido:**

El sistema embebido deberá:

- Medir la temperatura actual del recipiente
- Enfriar el recipiente el tiempo pedido
- Enfriar el recipiente hasta la temperatura pedida
- El sistema embebido mostrará por medio de un LCD los datos.
- El sistema embebido enviará a Android los datos.
- El sistema embebido podrá activarse y desactivarse a través de pulsadores.
- El sistema embebido deberá poder recibir órdenes de la aplicación de Android.

#### **3.2 Android:**

La aplicación de Android deberá:

- Recibir los datos enviados por el sistema embebido
- Poder establecer un tiempo de enfriamiento para el sistema embebido
- Poder establecer una temperatura de enfriamiento para el sistema embebido.
- Con el sensor de proximidad podrá detener el sistema embebido.
- Con un shake (usando el acelerómetro) deberá adicionar 30 segundos al tiempo de enfriamiento.
- Con el GPS obtener la zona geográfica

#### **4 Desarrollando el Sistema embebido**

Primero se tuvo la idea de hacer un aparato que pueda enfriar líquidos, y para ello se busco información y componentes electrónicos acorde.

Utilizamos las celdas peltier que aplicandole una diferencia de potencial (en nuestro caso 5v y 3.3v) se produce una diferencia de temperatura entre sus dos caras (una se enfriara y otra se calentara). Para la que caliente se necesita usar un disipador y ventilador (de CPU 12v) para poder bajar la temperatura del lado frío. ya que la diferencia de temperatura se mantiene “constante”.

Además, el Arduino UNO podría alimentar nuestras celdas en lo que respecta a la tensión necesaria, pero no en cuanto a la corriente necesaria ( 5 amperes por celda). Por lo cual se usó una fuente de PC. Y al tener una fuente de alimentación externa al Arduino, se necesitó también dos Relés (uno para cada celda) de 5volt (se usó de 6v por no conseguir el de 5v) y su transistor y diodo necesario para el correcto funcionamiento del Relé.

Se usaron dos pulsadores, uno para Iniciar y otro para Enfriar. Y para cada pulsador se usó una resistencia de 5 Kohm ya que si se presiona el pulsador sin alguna resistencia se produciría un cortocircuito que podría quemar nuestro Arduino. Al momento de probar nuestro pulsador se produjo un error, y es que el estado leído era “al azar”. Esto fue debido al llamado “efecto rebote” que significa que una vez que una persona presiona un pulsador, este tiene internamente un resorte, que queda oscilando. El arduino lee todas estas oscilaciones. Por lo cual hubo que modificar el Sketch y en principio se resolvió leyendo una vez el estado del pulsador, y haciendo un delay() de 300 ms hasta que se pueda leer otro estado de dicho pulsador. Luego esto se mejoró utilizando la función millis().

Para la comunicación (envío y recepción de mensajes hacia la aplicación de Android) se utilizó un módulo Bluetooth, y se eligió el modulo HC-05 antes que el HC-06, porque este último solo posea el modo Esclavo. Aunque al final se terminó usando el modo Esclavo del HC-05 por su simplicidad de conexión. Además, el Arduino tiene los puertos digitales 0 y 1 para la comunicación en Serie; pero nosotros usamos otros puertos debido a que los puertos 0 y 1 son los que utiliza el Compilador Arduino para comunicarse con la placa Arduino UNO a través del cable usb para

actualizar los sketches que programemos y subamos al Arduino.

Para mostrar los mensajes se utilizó un pantalla LCD de 16 caracteres por 2 renglones (1602A). Y para poder utilizarla en los Sketchs se incluye la librería pertinente ( <LiquidCrystal.h> ) que traía las funciones básicas del lcd.

Para medir la temperatura se utilizó un Termistor sumergible, que varía su resistencia de acuerdo a la temperatura en su extremo. Se lo ubico en el puerto Analogico 0 y se utilizo una resistencia de 5 Komh, para su correcta conexión. Y se utilizó funciones especiales propias del sensor, investigados y probados antes de su uso.

Se hicieron mediciones de las celdas peltier directamente sobre la superficie a diferentes tensiones de entrada, detalladas a continuación:

Sin Ventilador	Temperatura Ambiente	Lado Frío	Lado Caliente
5v en 10 min	19	11	45
3.3v en 10 min	19	12	40

Con Ventilador	Temperatura Ambiente	Lado Frío	Lado Caliente
5v en 10 min	19	10	-
3.3v en 10 min	19	11	-

Con Ventilador y placas aisladas	Temperatura Ambiente	Lado Frío	Lado Caliente
5v en 10 min	19	3	-
3.3v en 10 min	19	5	-
12v en 10 min	19	15	-
6v en 10 min(2 placas pegadas físicamente)	19	10	-
5v y 3.3v en 10 min(2 placas pegadas)	19	0	-

físicamente)			
--------------	--	--	--

Por lo cual se decidió utilizar Dos placas peltier pegadas físicamente, y una alimentada con 5v (la que va pegada al disipador con ventilador) y otra alimentada por 3.3v (la celda que queda a la vista).

Cabe aclarar que también se probó ambas celdas pegadas físicamente alimentadas en paralelo con 5v y se empezaron a derretir los cables, por todo el Amperaje que estas consumen y calentaban los mismos.

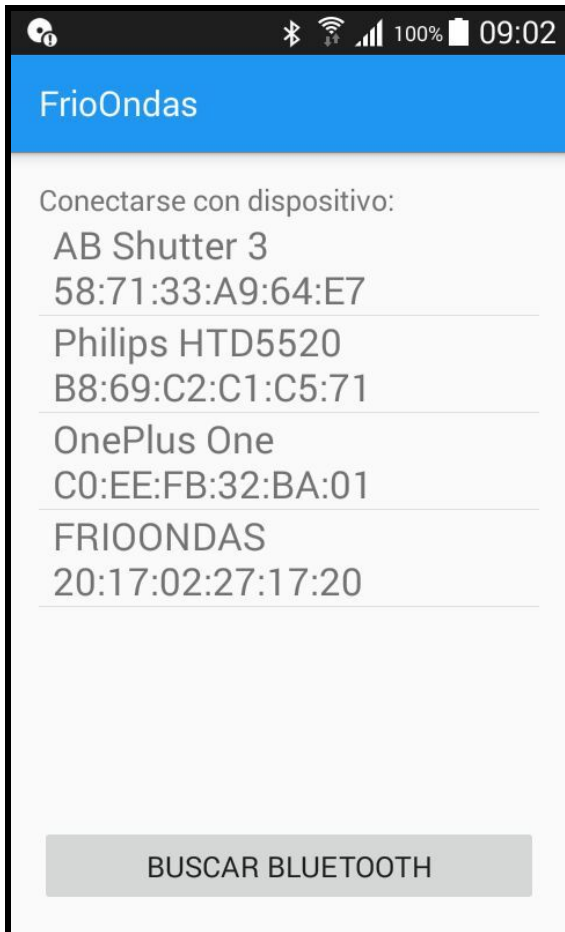
Finalmente se introdujeron las celdas peltier en un recipiente de poliestireno expandido (pote de helado de 1 kg) y se lo adaptó, para el prototipo final.

Durante la programación del Sketch se necesito contabilizar el tiempo en segundos, mientras se leen los sensores y se utilizan los actuadores, por lo cual resultó inútil la función “delay();” por lo que se tuvo que usar la función “millis();” que cuenta los milisegundos desde que se enciende el Arduino en una variable y puede contar hasta 53 días después del cual su contador pasará a 0. Por lo que es más que suficiente para nuestro Frioondas.



## 5 Activities

### Conexión con Bluetooth



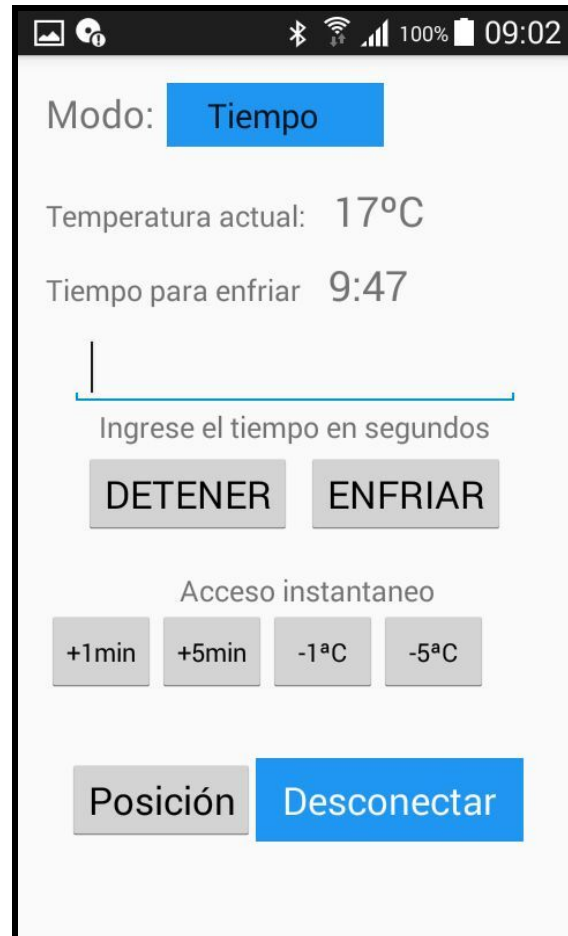
The screenshot shows the 'FrioOndas' app interface. At the top, there's a blue header with the app name. Below it, a list of Bluetooth devices is displayed, each with its name and MAC address. A 'BUSCAR BLUETOOTH' button is at the bottom.

Conectarse con dispositivo:

- AB Shutter 3  
58:71:33:A9:64:E7
- Philips HTD5520  
B8:69:C2:C1:C5:71
- OnePlus One  
C0:EE:FB:32:BA:01
- FRIOONDAS  
20:17:02:27:17:20

BUSCAR BLUETOOTH

### Control de Frioondas



The screenshot shows the 'Control de Frioondas' app interface. It displays the current mode as 'Tiempo', the current temperature as 17°C, and the time to cool as 9:47. There's a text input field for entering time in seconds, with 'DETENER' and 'ENFRIAR' buttons. Below that, 'Acceso instantaneo' is shown with buttons for '+1min', '+5min', '-1°C', and '-5°C'. At the bottom, there are 'Posición' and 'Desconectar' buttons.

Modo: **Tiempo**

Temperatura actual: 17°C

Tiempo para enfriar 9:47

Ingrese el tiempo en segundos

DETENER ENFRIAR

Acceso instantaneo

+1min +5min -1°C -5°C

Posición **Desconectar**

## **6 Comunicación**

Para la comunicación entre Android y Arduino se utiliza Bluetooth.

Android envía una cadena con el siguiente formato:

`"Tiempo;Temperatura Deseada"`

Arduino envía datos cada 1 segundo, con el siguiente formato:

`"Temperatura Actual;Tiempo Restante;Temperatura Deseada"`

## 7 Código

El código de Arduino (Sketch) esta subido a la siguiente carpeta de google drive:

<https://drive.google.com/file/d/0B24izbc8UAkXajlRc2d2bjJnTDA/view?usp=sharing>

Y el código de Android (pkt) están subidos a la siguiente carpeta github:

<https://github.com/gielDevelop/FrioOndas>

## 8 Fuentes

<https://developer.android.com/guide/topics/connectivity/bluetooth.html>

<http://mycyberacademy.com/creando-una-app-que-use-el-sensor-de-proximidad-de-tu-dispositivo-android/>

<https://stackoverflow.com/questions/19465049/changing-api-level-android-studio>

<https://javaheros.blogspot.com.ar/2016/03/ejemplos-de-alertdialogbuilder-en.html>

<https://www.hrpin.com/2011/04/android-gps-using-how-to-get-current-location-example>

<https://stackoverflow.com/questions/20151414/how-can-i-use-onitemselected-in-android>

<http://jasonmcreynolds.com/?p=388>

<https://playground.arduino.cc/ComponentLib/Thermistor2>

<https://www.openhacks.com/uploadsproductos/eone-1602a1.pdf>

<https://www.youtube.com/watch?v=LofnpxvFSG0>

<http://www.pesadillo.com/pesadillo/?p=5719>

<https://www.luisllamas.es/leer-un-pulsador-con-arduino/>