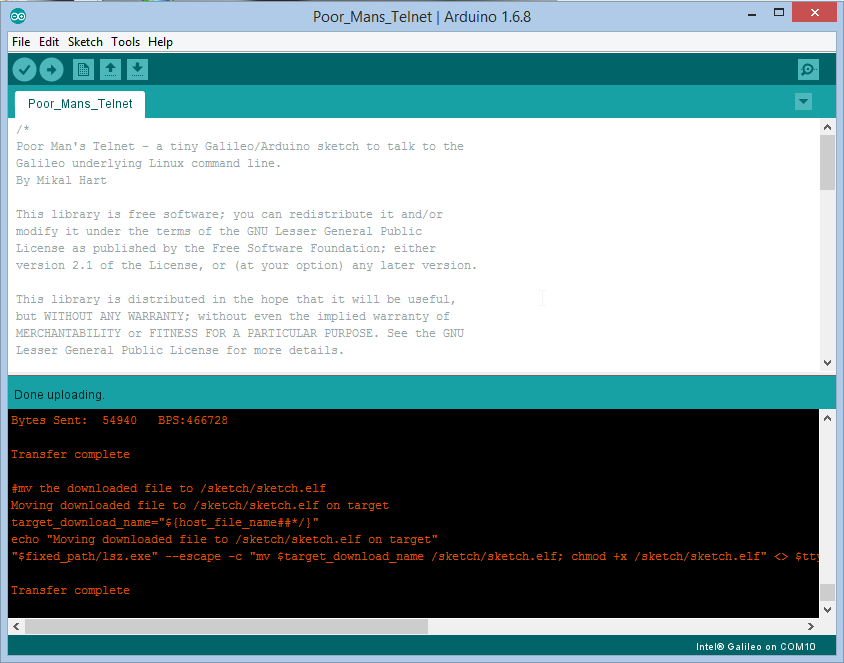
**Documentación - Proyecto Sistemas Embebidos**

**Conexión a la placa mediante el cable USB – micro USB**

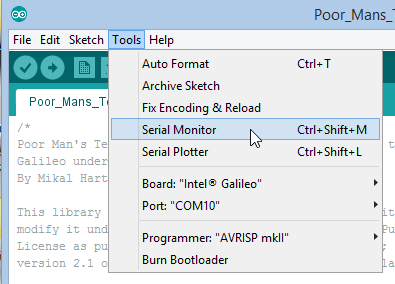
A continuación se muestra la forma que se utilizó para comunicarse con la placa en primera instancia.

Primero se procedió a hacer la conexión entre la placa y equipo Host mediante el cable USB – micro USB.

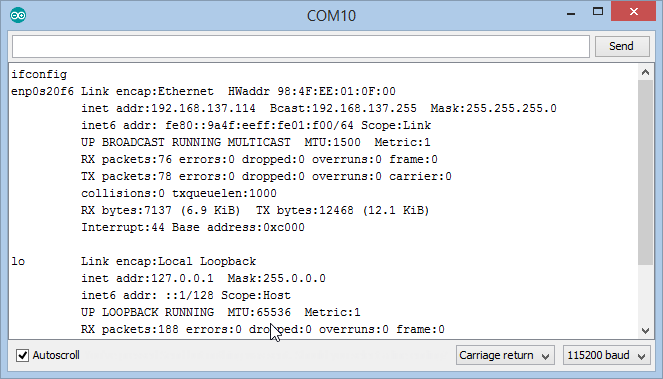
Una vez hecho esto se ejecutó el código que representa la solución sugerida en la misma página de Intel, provista por la Cátedra para emular una consola de Linux mediante el Arduino IDE: el Poor Man’s Telnet (o “Telnet del hombre pobre”).



Una vez compilado y ejecutado este código, se accede a Tools -> Serial Monitor.



En este monitor se pueden ejecutar comandos de Linux que nos devolverán información de la placa. En este caso lo usamos para chequear cual es la IP Address que le fue asignada.

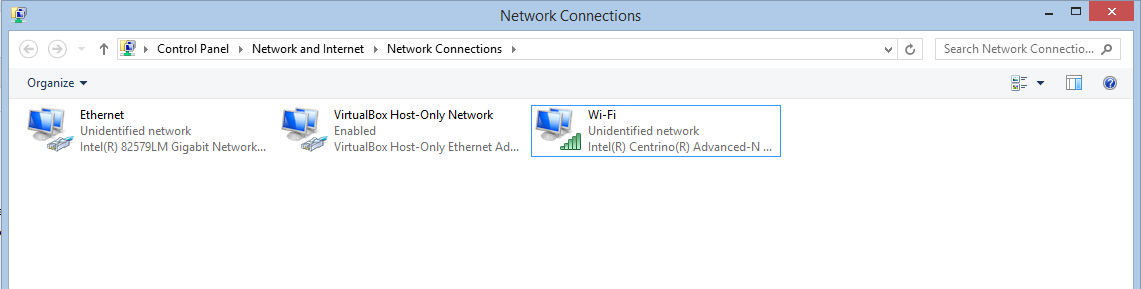


**Configuración del Network Bridge para tener internet en la Placa**

Los siguientes pasos muestran la forma en que se configuró el equipo Host, haciendo un Bridge entre el Adaptador de Red Ethernet y el Wi-Fi, para que la placa Intel Galileo pueda tener acceso a Internet.

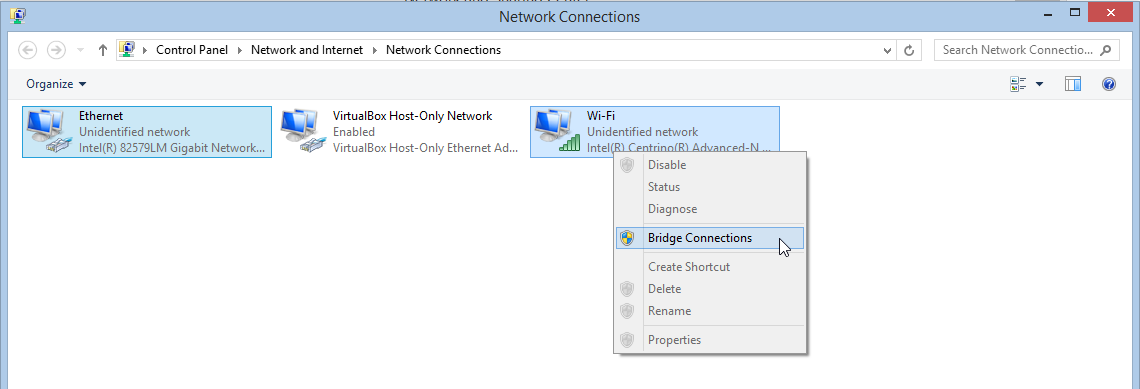
Primero se procedió a hacer la conexión entre la placa y el equipo Host mediante el cable de red.

Una vez hecho esto, desde el equipo Host se ingresó a Panel de Control -> View network status and tasks -> Change adapter settings.

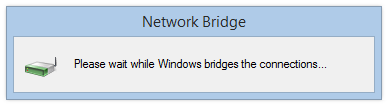


Como se puede ver en la imagen, tenemos la conexión Ethernet y la conexión Wi-Fi con acceso a Internet.

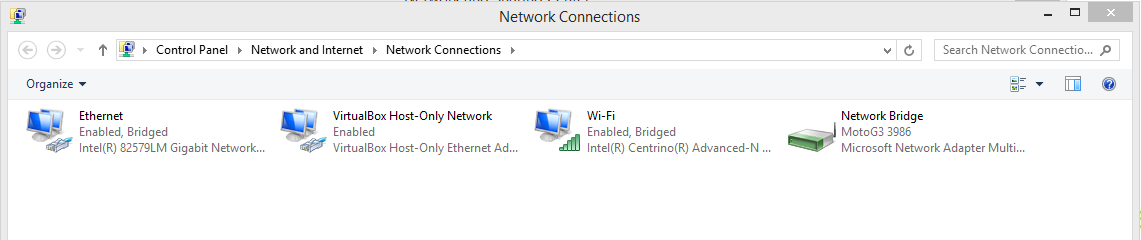
Luego, se procede a seleccionar ambos adaptadores y con botón derecho seleccionar “Bridge Connections”.



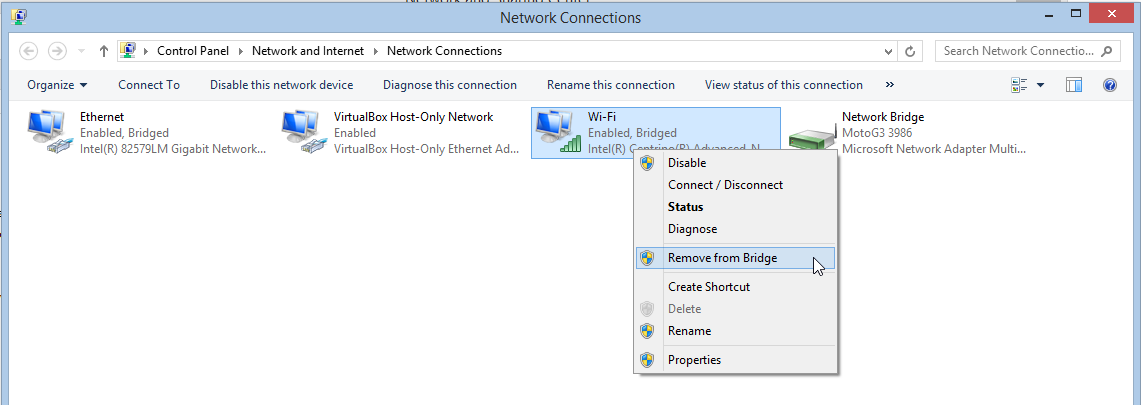
Al hacer esto el sistema comienza a generar el Bridge, esta operación demora unos minutos.



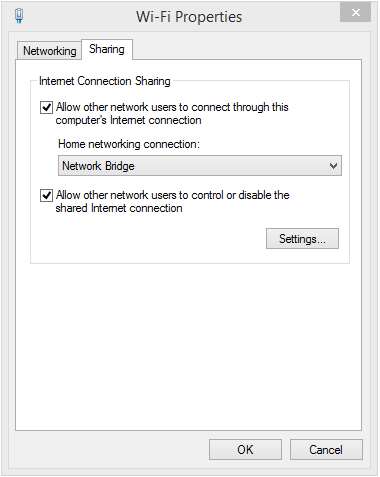
Al finalizar el Bridge, podemos ver como se agrega un nuevo icono en nuestras conexiones.



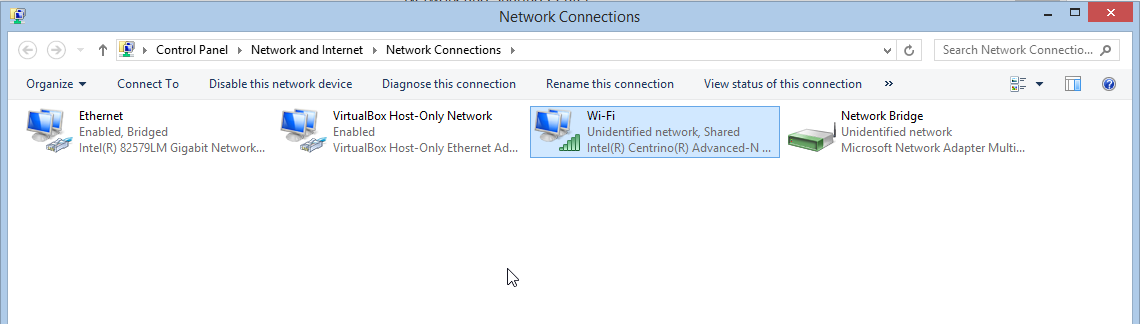
Lo siguiente es sacar el Wi-Fi del Bridge, como se ve en la siguiente imagen.



Pero luego de hacer esto, debemos volver a incluirlo, para esto, lo que hacemos es hacer click con el botón derecho sobre Wi-Fi e ir a las propiedades y en esta ventana, ir a la solapa “sharing” y tildar el primer checkbox y seleccionar la opción “Network Bridge” del combo.

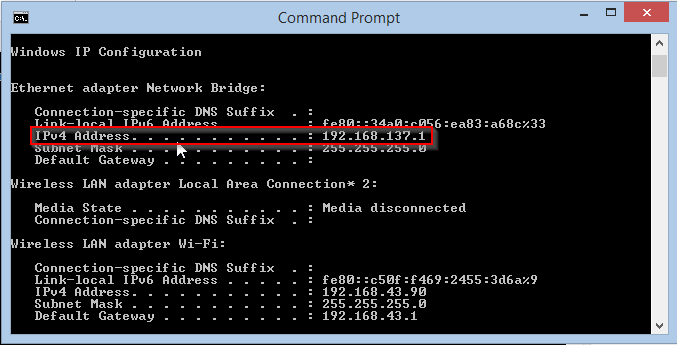


Al terminar este procedimiento, veremos que tanto el adaptador Ethernet y el Wi-Fi están puenteados, pero además, este último también aparece compartido (“shared”).



Para finalizar la configuración, podemos chequear que haya conexión entre el equipo Host y la placa de la siguiente manera:

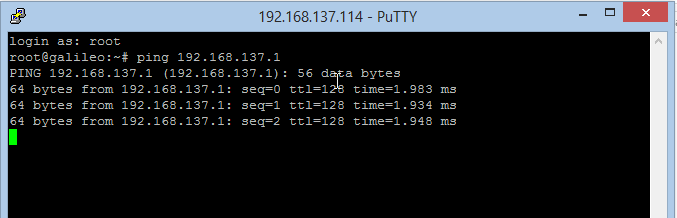
Primero abrimos una ventana Command en el equipo Host y mediante el comando “ipconfig” chequeamos la IP Address asignada al Bridge creado anteriormente.



Como se puede ver, la IP es 192.168.137.1, por lo que la placa debería tener una dirección en la subred 192.168.137.xxx

Esto puede verificarse utilizando el Poor Man’s Telnet como fue explicado con anterioridad.

Por ultimo mostramos como ingresando a la placa utilizando el Putty, se verifico que haya comunicación entre la placa y el equipo Host, utilzando el comando “ping”.



De esta forma nos aseguramos que la placa está accediendo a internet mediante el cable Ethernet cuyo adaptador esta puenteado al Wi-Fi.

**Código Fuente**

Bibliotecas Utilizadas:

#include <Wire.h>

Esta biblioteca le permite comunicarse con dispositivos I2C / TWI y facilitar las funciones.

Esta librería está incluida en el IDE de Arduino que utilizamos.

#include "rgb\_lcd.h"

Esta biblioteca permite interactuar con la pantalla LCD y teclas del shield. Se encarga de traducir los comandos a través del puerto de expansión. Esta biblioteca no está incluida en el IDE de arduino, Fue descargada desde este (https://github.com/Seeed-Studio/Grove\_LCD\_RGB\_Backlight)

#include <Ethernet.h>

Esta biblioteca permite que la placa pueda conectarse a internet. Puede servir como un servidor, aceptar conexiones entrantes o de un cliente que realiza conexiones salientes. La biblioteca es compatible con hasta cuatro conexiones simultáneas (entrante o saliente o una combinación). Esta librería está incluida en el IDE de Arduino.

#include <stdlib.h>

(standard library o biblioteca estándar)

Esta es la biblioteca estándar, permite utilizar las funciones Aritméticas, de números aleatorios y conversión de cadenas básicas que se utilizan en todo el código. Esta biblioteca se encuentra incluida en el IDE de Arduino pero también se puede descargar desde el siguiente link: <http://fullversion-net.eu/descargar-libreria-stdlib.h-para-arduino.html>

Variables Utilizadas:

Estas Variables se utilizan para colorear las pantalla conectada al shield, específicamente se settean los colores que mostrará la misma.

int colorR = 255;

int colorG = 255;

int colorB = 0;

Las siguiente variables son necesarias para la toma, parseo y cálculo específico del valor proveniente del sensor de temperatura.

const int b=3975;

float resistance;

float temperature;

Por último estas variables son utilizadas para la comunicación con la página Web, lectura, y envío de datos, conexión con internet.

String tempLimit = "";

float tempLimitFloat = 0;

long previousMillis = 0;

unsigned long currentMillis = 0;

long interval = 5000; // INTERVALO DE LECTURA CADA 30 SEGUNDOS

String data;

EthernetClient client;

EthernetClient client2;

En esta parte del código determinó a los pines 6 y 7 como salidas del sketch, la velocidad de la conexión serial e inicializo la pantalla lcd con los colores respectivos.

void setup()

{

pinMode(6, OUTPUT);

pinMode(7, OUTPUT);

Serial.begin(115200);

while (!Serial) {

; // wait for serial port to connect. Needed for native USB port only

}

lcd.begin(16, 2);

lcd.setRGB(colorR, colorG, colorB);

delay(2000);

data = "";

}

Parte inicial del código dentro del loop;

Inicio el String tempLimit, donde se guardará el valor temperatura, que fue seteado desde la página Web.

Género la conexión con la variable client2 que es un EthernetClient, a través de la función connect. Los parámetros son la dirección IP de la página destino y el puerto a utilizar para la comunicación.

Una vez hecha la conexión, se lee mediante un GET el archivo donde esta el dato que se ingresó desde la página Web, como temperatura límite para el LCD

En caso que no se pueda conectar con el servidor se imprimirá en consola, “connection failed”

void loop()

{

tempLimit = ""; Inicio el String

Serial.println("connecting...");

if (client2.connect("soa.hol.es", 80)) { Conecto con la direcion IP de la pagina y el puerto

Serial.println("connected");

client2.println("GET http://soa.hol.es/temperatura.dat HTTP/1.1"); Get del archivo a leer

client2.println("Host: soa.hol.es");

client2.println("Connection: close");

client2.println();

} else {

Serial.println("connection failed");

}

En esta sección se lee el archivo temperatura.dat mediante un char (c) y se desestima el header que solo tiene caracteres informativos y solo se toma el valor entre “|” que es la temperatura que se seteo desde la página Web principal. De esta forma queda la temperatura límite concatenada en la variable tempLimit.

Luego se convierte este String en flotante, con las ultimas tres lineas de esta porcion de codigo.

if (client2.available()) {

char c;

while((c = client2.read()) != '|');

while((c = client2.read()) != '|') {

tempLimit += c;

}

Serial.print(tempLimit);

char floatbuf[32]; // make this at least big enough for the whole string

tempLimit.toCharArray(floatbuf, sizeof(floatbuf));

tempLimitFloat = atof(floatbuf);

}

Finalizando la consulta sobre la temperatura límite seteada se debe finalizar la conexión de la siguiente manera:

En caso de que la conexión no sea viable de conectar, se desconecta definitivamente y se para la conexión establecida por client2.

Y de todas formas en caso de que vuelva a conectar tambien se para la conexión con el server.

if (!client2.connected())

{

Serial.println();

Serial.println("disconnecting.");

client2.stop();

while (true);

}

if (client2.connected())

{

client2.stop(); // Desconexión del servidor.

}

lcd.clear();

int val = analogRead(pinTemp);

resistance=(float)(1023-val)\*10000/val; // Cálculo intermedio para la temperatura

temperature=1/(log(resistance/10000)/b+1/298.15)-273.15; // Cálculo de la temperatura

// Si la temperatura es menor que el valor ingresado por la página web se colorea el lcd en azul y se muestra un mensaje. Caso contrario el lcd se colorea en rojo, muestra un mensaje y comienza a titilar un led.

if (temperature < tempLimitFloat)

{

digitalWrite(6, LOW);

colorR = 0;

colorG = 0;

colorB = 255;

lcd.setRGB(colorR, colorG, colorB);

lcd.print("Hace frio: ");

lcd.print(temperature);

}

else

{

colorR = 255;

colorG = 0;

colorB = 0;

lcd.setRGB(colorR, colorG, colorB);

lcd.print("Hace calor: ");

lcd.print(temperature);

digitalWrite(6, HIGH);

sleep(1);

digitalWrite(6, LOW);

sleep(1);

}

// Conversión de la variable temperature float a string.

String strEntero = String((int)temperature);

String strDecimal = String((int)((temperature-(int)temperature)\*100));

data = "temp1=" + strEntero + "." + strDecimal;

currentMillis = millis(); // Retorna el tiempo actual en milisegundos.

// Se comprueba que la diferencia sea menor a un intervalo dado. En caso de ser verdadero se conecta al servidor. Luego le vamos pasando la temperatura que va tomando el sensor.

if(currentMillis - previousMillis > interval)

{

previousMillis = currentMillis;

if (client.connect("soa.hol.es",80))

{

client.println("POST /add.php HTTP/1.1");

client.println("Host: soa.hol.es"); // SERVER ADDRESS HERE TOO

client.println("Content-Type: application/x-www-form-urlencoded");

client.print("Content-Length: ");

client.println(data.length());

client.println();

client.print(data);

}

if (client.connected())

client.stop();

}

**SERVIDOR WEB**

**add.php**

<?php

include("connect.php");

$link=Connection();

$temp1=$\_POST["temp1"];

$query = "INSERT INTO `tempLog` (`temperatura`)

VALUES ('".$temp1."')";

mysql\_query($query,$link);

mysql\_close($link);

header("Location: index.php");

?>

Esta página recibe los datos enviados por la placa via **POST** e inserta el valor en una base de datos mysql alojada en un servidor web externo.

**index.php**

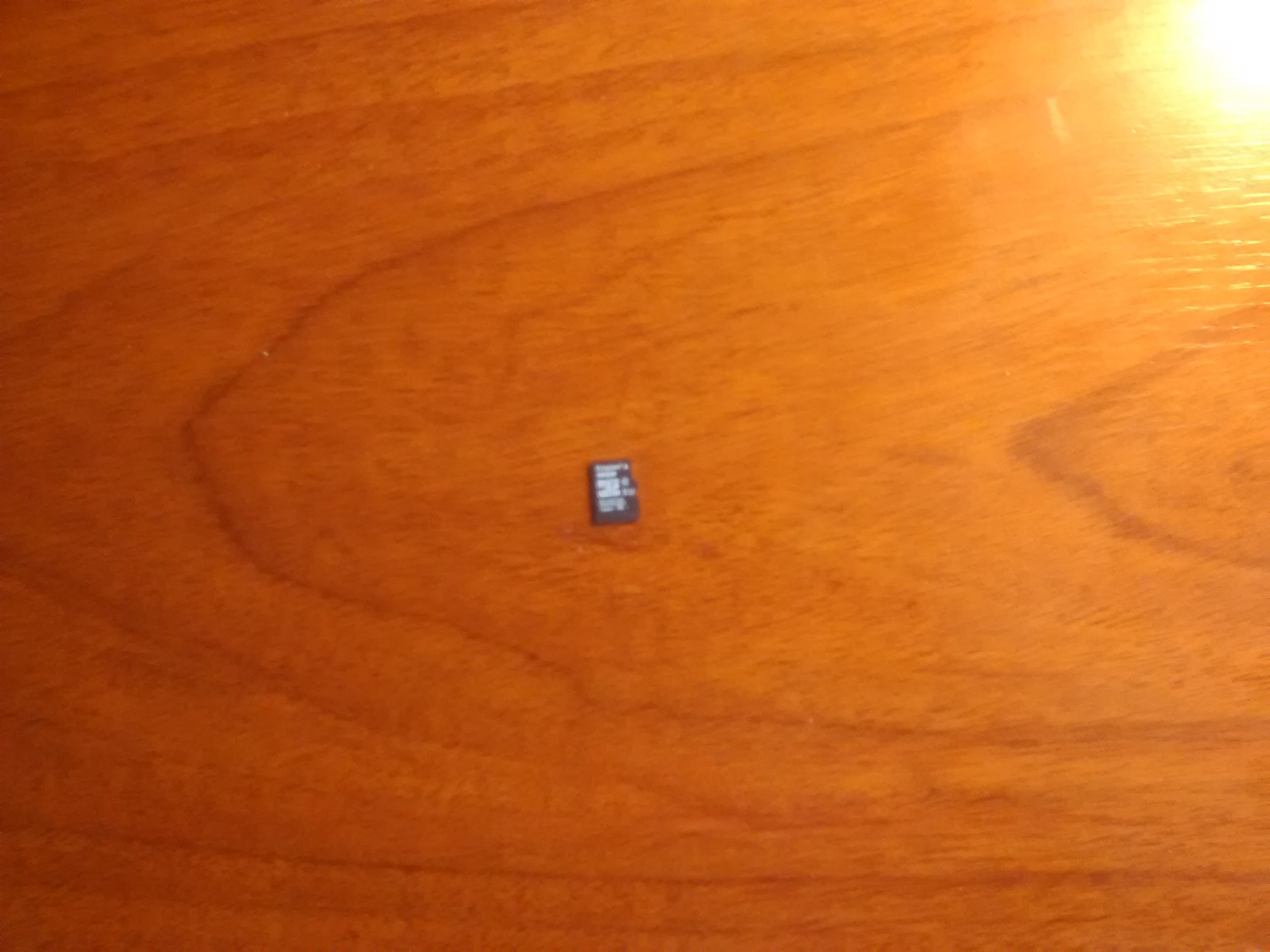
Esta página es la encargada de mostrar la vista al usuario con la temperatura actual leída desde la base de datos utilizando para esto AJAX, logrando así realizar consultas asíncronas y actualizar la temperatura cada cierto intervalo de tiempo sin refrescar la página.

Además de incluir un formulario por donde se puede establecer una temperatura máxima y la guarda en un archivo de texto (**temperatura.dat**) para que posteriormente la placa pueda leer este valor y trabajar con el mismo.

**Hardware y elementos Utilizados**

MicroSD de 8gb Kingston, clase 10 que la utilizaremos para almacenar el SO recomendado por la cátedra que se puede descargar desde el siguiente link:

<https://software.intel.com/en-us/iot/hardware/galileo/downloads>



Cable de Red, para garantizar una conexión de red por el cual se montará un bridge, anteriormente explicado, y así tener acceso a internet desde la placa Galileo gen 1.



Cable USB - MicroUSB, este es usado para conectar la PC remota con la placa y de esta manera sobre esta conexión, poder compilar el código desde la IDE de Arduino en la placa mediante un sketch que utilizaremos.

.

Placa Intel Galileo gen 1, el corazón de este proyecto, a través de esta placa se ejecutan el sketch infinitamente y tiene los puertos para conectar los sensores y actuadores e incluso el shield que describiremos el en siguiente ítem.



Shield Arduino, este circuito brinda la estandarización de conexiones para sensores y actuadores, en este caso utilizaremos las conexion digitales (pantalla LCD, LED) y las conexiones analogicas (sensor de temperatura)



Sensor de Temperatura, (incluyendo cable de conexión para la shield). Este sensor tomará la temperatura del ambiente que utilizaremos para informar en la página Web.



Actuador LED(Incluido cable de conexión para shield, y el socket led) Este led se encenderá cuando la temperatura supere la temperatura límite, setteada desde la página Web. El cable permite conectar con la interfaz de la shield y el socket es el intermediario entre este cable el led.



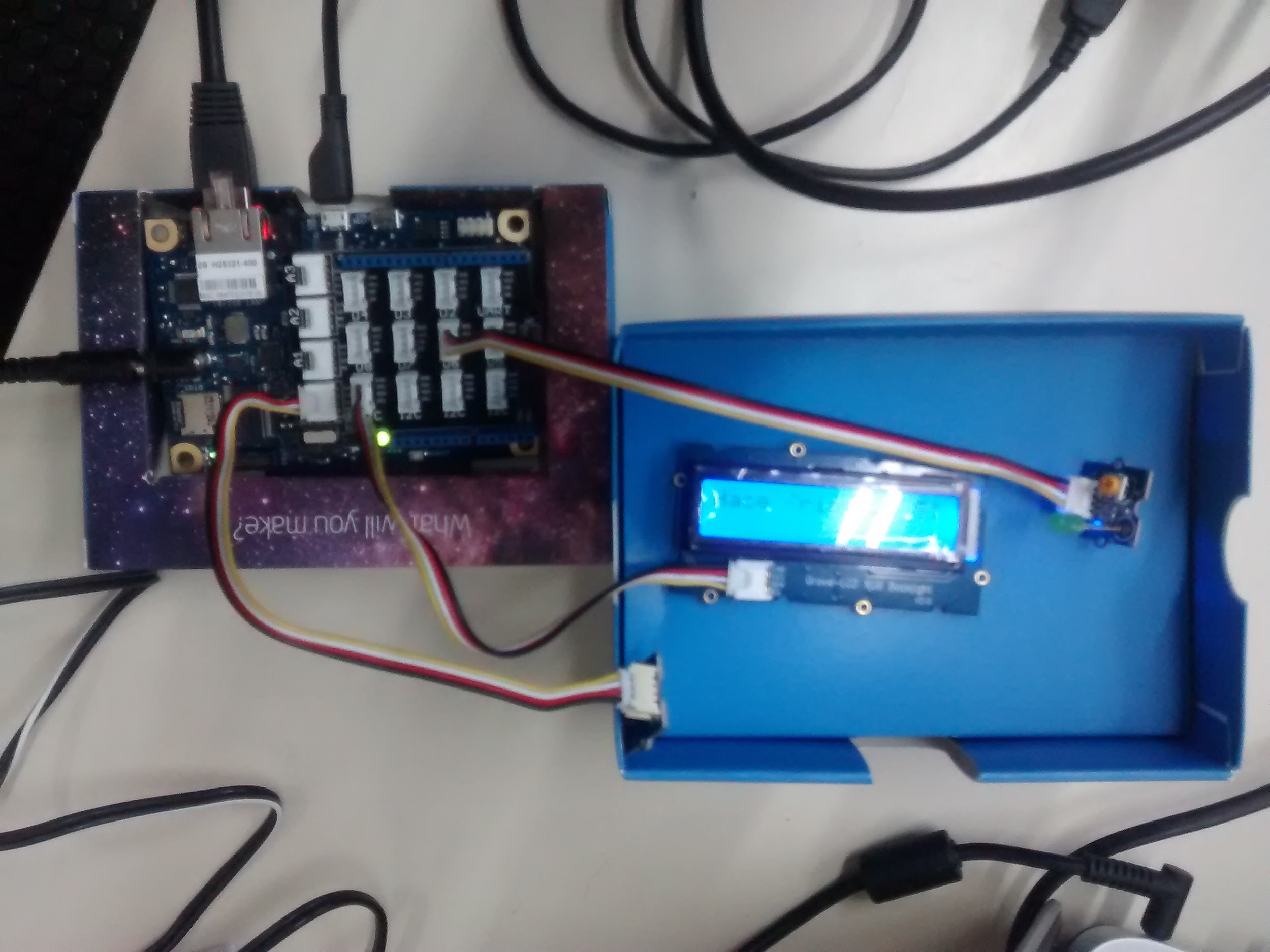
Actuador pantalla lcd (la imagen incluye el cable para conectar con la shield) En esta pantalla se visualizará todo el tiempo la temperatura medida por el sensor. Esta se encontrara de color azul mientras la temperatura actual no supere la temperatura límite, en caso de ser superada se verá de color Rojo.



En la siguiente imagen se grafica como es la conexión entre la shield y la placa Intel Galileo gen 1, esta shield permite conectar los sensores y actuadores con la placa de manera estandarizada a través de la interfaz que brinda la shield.



En la siguiente imagen se grafica el funcionamiento cuando la temperatura actual es inferior a la temperatura límite. Se visualizan pantalla azul, led apagado, y la lectura de la temperatura actual.



En la siguiente imagen se grafica el funcionamiento cuando la temperatura actual es superior a la temperatura límite. Se visualizan pantalla roja, led encendido, y la lectura de la temperatura actual.

