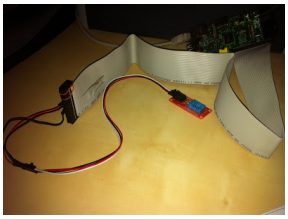
Crear una estacion meteorologica con raspberry pi y un sensor dth11

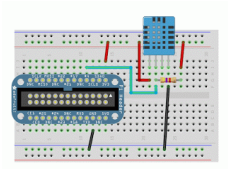
Ya puestos a trastear con la raspberry y sus conexiones al GPIO, vamos a ver como podemos crear una pequeña estacion meteorologica con la raspberry pi, y un sensor dth11.



Que necesitamos??

Una raspberry pi
Un sensor dth11 (unos 2 € en dealextreme)
Cables para conectarlo

Notas sobre los materiales necesarios
El sensor que utilizo yo, y el que aparece en la imagen, viene con una resistencia de 10k(amarillo violeta rojo oro) incorparada entre el conector de señal y nuestra conexion al gpio. Si lo compramos suelto, habra que hacer un circuito en un protoboard tal como aparece en esta imagen:



Montaje de los componentes:

Conectamos el sensor dth11 al gpio de la raspberry asi:

```
RPi VCC (pin 1) -> DHT11 pin 1
RPi GPIO4 (pin 7) -> DHT11 pin 2
RPi GND (pin 6) -> DHT11 pin 4 (pin3 si solo tiene 3 pins)
```

Creando un programa para poder acceder al sensor y leer temperatura y humedad:

Una vez realizadas las conexiones, vamos a crearnos un programa que realice

las lecturas, y que nos saque los resultados por la pantalla.

He elegido el lenguaje de programacion en C

Instalamos lo necesario:

nota: Todo lo voy a hacer desde el terminal. Esto me permitira hacerlo todo,

con una conexion SSH hacia mi raspberry.

Me situo en mi directorio HOME

cd ~

Ejecuto estos comandos:

```
sudo apt-get install git-core build-essential
git clone git://git.drogon.net/wiringPi
cd wiringPi
./build
```

Con esto ya tenemos lista la libreria wiringPi, para poder usar el puerto GPIO

de la raspberry de una manera mas sencilla

Creamos el siguiente archivo con este comando:

nano dth11.c

Escribimos el siguiente codigo:

CODIGO DESDE AQUI - Archivo dth11.c

```
 //Incluimos librerias necesarias
#include <wiringPi.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

//Definimos constantes
#define MAX_TIME 85
#define DHT11PIN 7
#define ATTEMPTS 5

//Definimos un vector global
int dht11_val[5]={0,0,0,0,0};

////////////////////////////////////////////////////////////
//Funcion principal para leer los valores del sensor.
int dht11_read_val(){
 uint8_t laststate=HIGH;
 uint8_t counter=0;
 uint8_t j=0,i;
 for(i=0;i<5;i++){
  dht11_val[i]=0;
 }
 pinMode(DHT11PIN,OUTPUT);
 digitalWrite(DHT11PIN,LOW);
 delay(18);
 digitalWrite(DHT11PIN,HIGH);
 delayMicroseconds(40);
 pinMode(DHT11PIN,INPUT);
 for(i=0;i<MAX_TIME;i++){
  counter=0;
  while(digitalRead(DHT11PIN)==laststate){
   counter++;
   delayMicroseconds(1);
   if(counter==255){
    break;
   }
  }
  laststate=digitalRead(DHT11PIN);
  if(counter==255){
   break;
  }
  //Las 3 primeras transiciones son ignoradas
  if((i>=4)&&(i%2==0)){
   dht11_val[j/8]<<=1;
   if(counter>16){
    dht11_val[j/8]|=1;
   }
   j++;
  }
 }

// Hacemos una suma de comprobacion para ver si el dato es correcto. Si es asi, lo mostramos
 if((j>=40)&&(dht11_val[4]==((dht11_val[0]+dht11_val[1]+dht11_val[2]+dht11_val[3])& 0xFF))){
  printf("%d.%d,%d.%d\n",dht11_val[0],dht11_val[1],dht11_val[2],dht11_val[3]);
```

```
  return 1;
 }else{
  return 0;
 }
}

//////////////////////////////////////////////////////////////////////
//Empieza nuestro programa principal.
int main(void){
 //Establecemos el numero de intentos que vamos a realizar
 //la constante ATTEMPTS esta definida arriba
 int attempts=ATTEMPTS;

 //Si la libreria wiringPi, ve el GPIO no esta listo, salimos de la aplicacion
 if(wiringPiSetup()==-1){
  exit(1);
 }

 while(attempts){
  //Intentamos leer el valor del gpio, llamando a la funcion
  int success = dht11_read_val();

  //Si leemos con exito, salimos del while, y se acaba el programa
  if (success){
   break;
  }

  //Si no lee con exito, restamos 1, al numero de intentos
  attempts--;

  //Esperamos medio segundo antes del siguiente intento.
  delay(500);
 }
 return 0;
}
```

CODIGO HASTA AQUI-------- FIN DE ARCHIVO dth11.c

Una vez escrito, lo compilamos con el siguiente comando:

```
gcc -o dht11 dht11.c -L /usr/local/lib -l wiringPi
```

Si compila bien, ejecutamos con este comando:

```
sudo ./dht11
```

Deberia mostrarnos la temperatura, y la humedad separadas por comas.

Siguiente paso!!!

Como lo que nos interesa realmente, es tener unos datos actualizados cada minuto,
y en forma de grafica, vamos a instalar un servidor web, y crearnos asi una
pagina web donde podamos consultar los resultados que nos dara nuestra medicio
de una forma grafica. Para eso, instalamos apache, y creamos la web necesaria.

Instalacion de apache:

Ejecutamos el siguiente comando:

```
apt-get install apache2
```

Arrancamos apache:

```
service apache2 start
```

Una vez instalado el

Creacion automatica de la medicion de temperatura y humedad:

Vamos a crearnos un fichero, para que vaya guardando de forma automatica
los valores de temperatura y humedad, asi como la fecha en la que se ha
tomado dicha medicion.

Nos situamos de nuevo en el directorio HOME:

```
cd ~
```

Creamos el siguiente fichero con el siguiente codigo:

```
nano dth11.sh
```

CODIGO DESDE AQUI - Archivo dth11.sh

```
#!/bin/bash
FECHA=$(date +\%Y\%m\%d\%H\%M\%S)
COMA=","
TEMP=$(/home/usuario/dth11)
echo "$FECHA$COMA$TEMP" >> /var/www/temp.log
```

CODIGO HASTA AQUI-------- FIN DE ARCHIVO dth11.sh

Con esto lo que hacemos, es un script que nos dara como resultado la fecha
en el formato adecuado, seguido de la temperatura y humedad, y lo almacenara
en un archivo llamado temp.log, en el directorio raiz de APACHE por defecto,
para su lectura posterior.

Tarea programada con cron

A continuacion, le vamos a decir a cron (nuestro programador de tareas en linux),
que ejecute el script cada minuto. Para eso ejecutamos el siguiente comando:

```
sudo crontab -e
```

Y añadimos la siguiente linea al final:

```
* * * * * /home/usuario/dth11.sh
```

Creamos la pagina web para mostrarnos los resultados leidos:

Nos situamos en el directorio de nuestro servidor web con el siguiente comando:

```
cd /var/www
```

Si existe un archivo index.html, lo borramos con el comando:

```
rm index.html
```

Descargamos la libreria dygraphs para poder generar la grafica:

```
wget -P /var/www http://dygraphs.com/dygraph-combined.js
```

Nos creamos el archivo index.html para almacenar la web:

```
nano index.html
```

CODIGO DESDE AQUI - Archivo index.html

```
<html>
<head>
<script type="text/javascript" src="dygraph-combined.js"></script>
```

```
</head>
<body>

<h1>Grafica con temperaturas y humedad</h1>

<p>Esta gr&aacute;fica, esta generada a partir de las mediciones tomadas por un sensor dth11, conectado a una raspberry pi. Se accede al sensor mediante lenguaje de programacion C, usando las librerias wiringPi, para poder leer y utilizar el conector GPIO de la raspberry Pi.</p>


<hr/>

<div id="graphdiv" style="width:750px; height:400px;"></div>
<script type="text/javascript">

 //Funcion para añadir ceros a los numeros de un solo digito
 function addZero(num){
  var s=num+"";
  if (s.length < 2){
   s="0"+s;
  }
  return s;
 }

 //Funcion para dar formato a la fecha, devolviendo la misma formateada
 function dateFormat(indate){
  var hh = addZero(indate.getHours());
  var MM = addZero(indate.getMinutes());
  //var ss = addZero(indate.getSeconds()); //No usamos los segundos
  var dd = addZero(indate.getDate());
  var mm = addZero(indate.getMonth()+1);
  var yyyy = addZero(indate.getFullYear());
  return dd+'/'+mm+' '+hh+':'+MM;
 }

 //Funcion principal para generar el grafico. Se genera en el id="graphdiv"
 g = new Dygraph(
 document.getElementById("graphdiv"),
 "temp.log",
 {
   xValueParser: function(x) {
  var date = new Date(x.replace(
    /^(\d{4})(\d\d)(\d\d)(\d\d)(\d\d)(\d\d)$/,
    '$4:$5:$6 $2/$3/$1'
  ));
  return date.getTime();
   },
   axes: {
  x: {
    ticker: Dygraph.dateTicker,
    axisLabelFormatter: function(x) {
   return dateFormat(new Date(x));
    },
    valueFormatter: function(x) {
   return dateFormat(new Date(x));
    }
  }
   },
   labelsDivWidth: 310,
   rollPeriod: 30,
   strokeWidth: 2.0,
   labels: ['Date','Humedad (%)','Temp (&deg;C)']
 }
 );
</script>

<hr/>

</body>
</html>
```

CODIGO HASTA AQUI-------- FIN DE ARCHIVO index.html

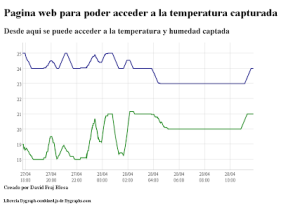
Probandolo todo!!!

Si todo va bien, podemos probar que todo esto funciona, desde nuestro navegador

de paginas web, apuntando a la direccion ip de nuestra raspberry

http://ip.de.mi.pi

Deberia aparecer algo como esto:



Resumen:

Espero que os sirva de utilidad. A mi me ha funcionado todo a la perfeccion.

Un saludo a todos!!!

Fuente de informacion:

http://chrisbaume.wordpress.com/2013/02/10/beer-monitoring/

http://dygraphs.com/

http://rpidude.blogspot.com.es/2012/11/temp-sensor-and-wiringpi.html

http://dx.com/es/p/arduino-digital-temperature-humidity-sensor-module-121350

http://ubuntuone.com/6mT9cTREz90BUfvQD1AGNy

Agradecimientos:

Muchas gracias a Alberto, que me ha dejado todo el hardware necesario para poder

hacer mis pinitos con todo esto. Un abrazo!!!