Sensors and relays for home automation using Arduino

Home automation via Internet becomes exceedingly popular. Simple and affordable interface of sensors and relays to computer is needed for hobbyists. Arduino is very attractive for this purpose providing a standardized and very popular platform, easily programmable via USB and reliable.

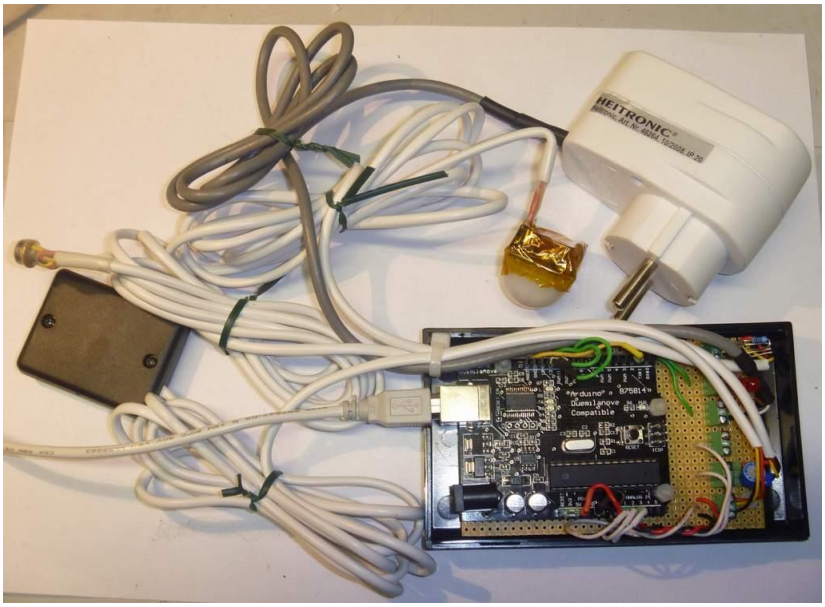
This Instructable describes how to connect to Arduino:

- sound level detector
- photodiode light sensor
- 1-wire bus temperature sensors
- PIR motion detector
- power outlet controlled by a relay

Applications:

- Impress your friends by switching on light via Internet using mobile phone.
- Quantify Self movement: measure sound level while you sleep (snorking), bed temperature - how long you sleep.
- Baby monitoring
- Temperature
- Sound or PIR activated lights.
- House alarm using PIR, sound, temperature
- Switch on light in the evening, or heating pump, *etc*.

Arduino can be connected to Internet via Ethernet shield, Windows PC, Ubuntu or some OpenWRT router with USB. Temperature or sunshine data can be uploaded to Pachube to generate plots. Alarm data could be emailed. Everyone needs to customize according to own wishes.

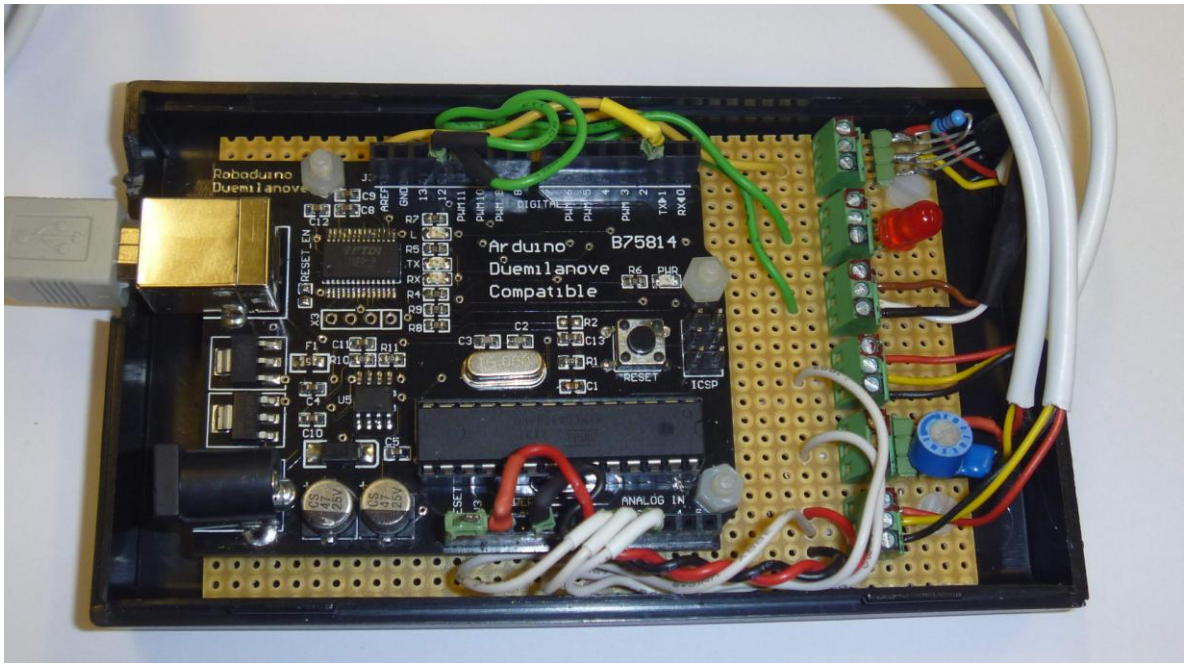


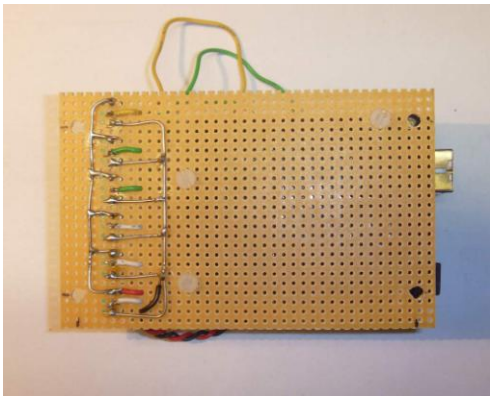
Construction

Goal was to make a simple and reliable board for connecting sensors using long cables.
For long cables it is preferable to use screw connectors instead of soldering. That allows to adjust cable length easily.
Screw connectors distribute GND, +5V and signal wires. Signal wire of the screw connector can be plugged into any pin connection of Arduino making setup easily customizable without soldering.

Picture shows implementation example for

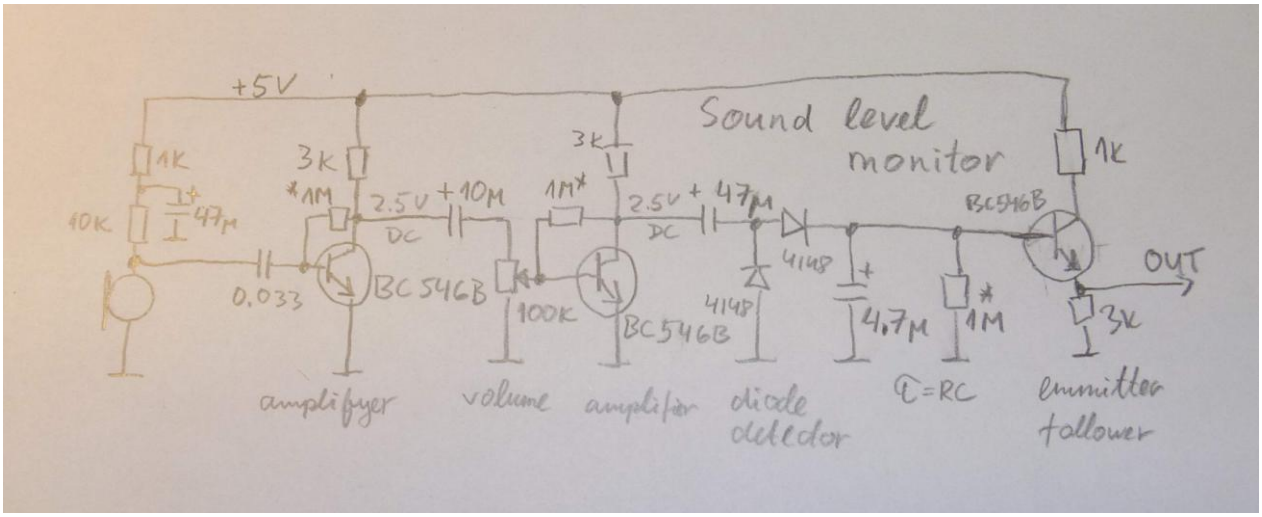
- 3 analog inputs: (PIR, photodiode with variable resistor, sound level monitor)
- 2 digital outputs: (power outlet relay, red LED).
- 1-wire bus with 4.7k pull-up resistor.



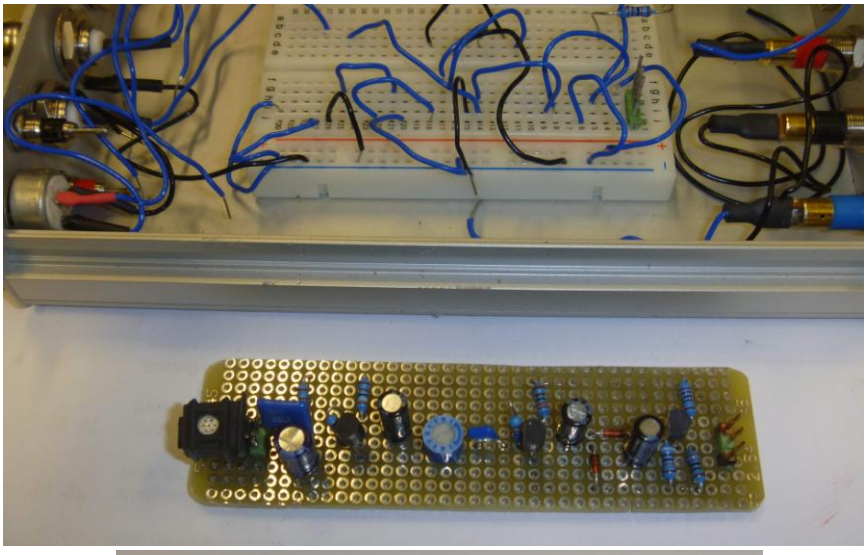


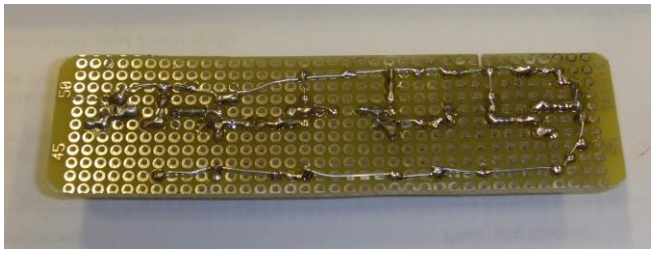
Sound level monitor

Snorkling in the night, sound activated lights, alarm Smart bay monitor:
http://www.withings.com/en/babymonitor
http://www.iphones.ru/iNotes/216707



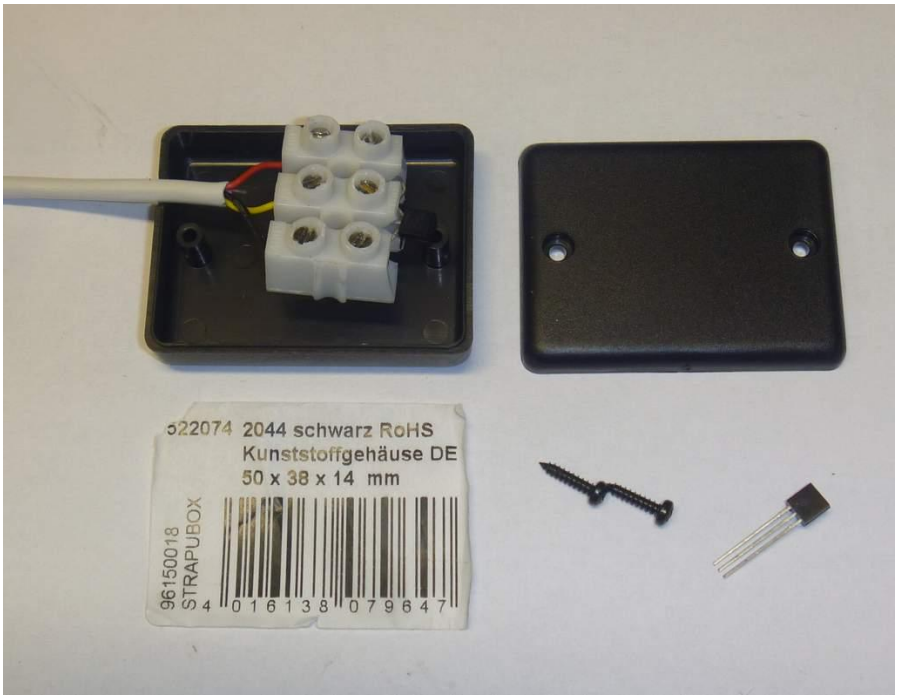
Mic, followed by two amplifier stages, diode detector like in old radios, emmiter follower as output. Buffer. Build circuit on a solderless breadboard, adjust parts and then solder.
Microphone from old mobile phone. Adjust the frequency range using 0.033uF to 1uF capacitor after microphone. Diode detector time constant 4.7 uF x 1Meg= ca 5s.
Adjust amplifier collector voltages to ca 2.5 V by slecting R*. Powered from Arduino by cable.

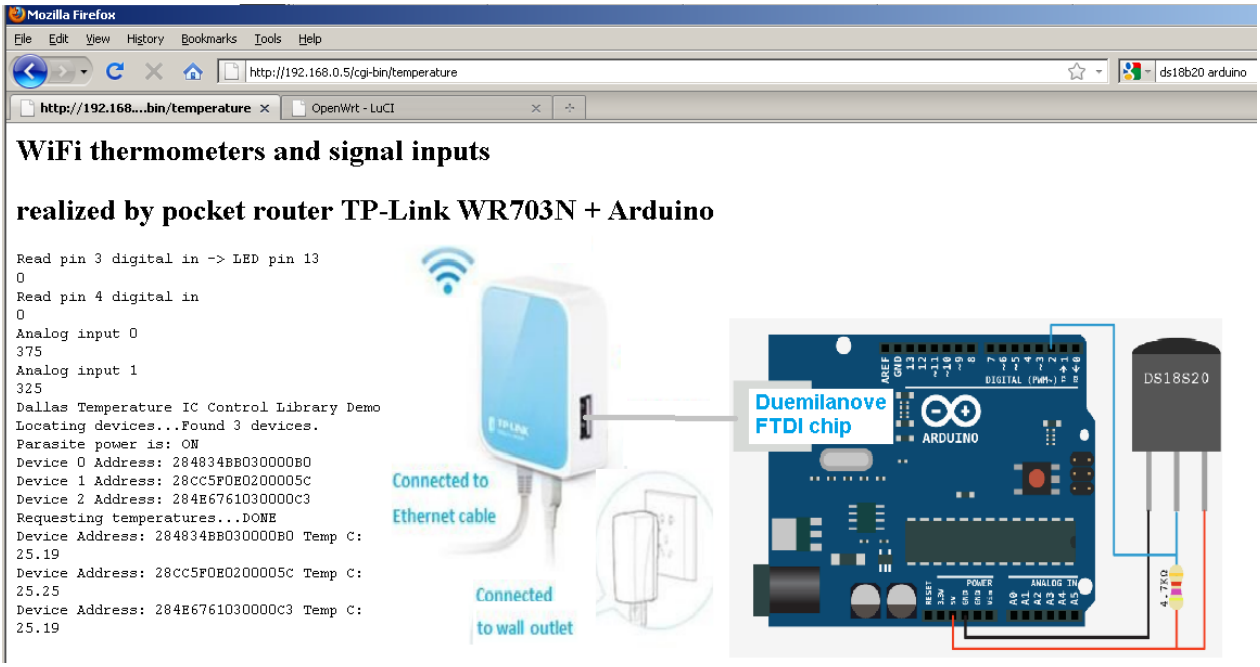




1-wire temperature sensors

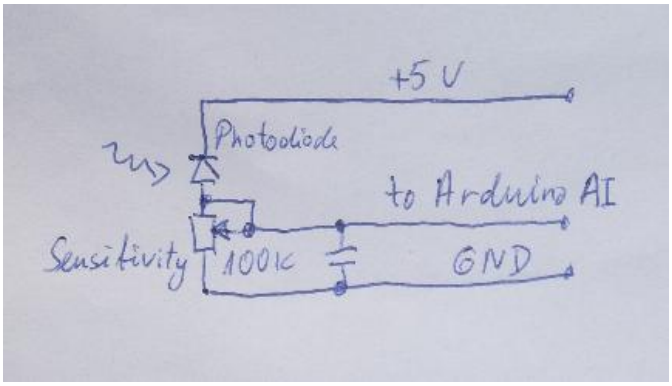
Arduino can simulate 1-wire bus on one of its pins and read out many DS18B20 temperature sensors connected to it. Small boxes can be used to hide sensors and screw connections for cables.



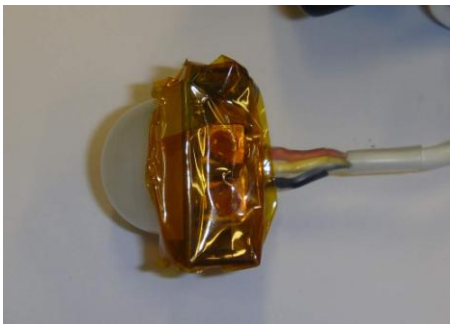


Light sensor

Can use to switch on light in the evening. Can monitor sunshine.
Photodiode in reversed bias mode. No current flowing in darkness.
Current is transformed into voltage across the load resistor.
Ouput linearly proportional to light level. Sensitivity set with variable load resistor.
Capacitor filters out 50 Hz.



PIR motion detector



From Ebay:
5x Adjust Infrared IR PIR Motion Sensor Detector Module US $12,68 from other.deal
http://www.ebay.de/itm/5x-Adjust-Infrared-IR-PIR-Motion-Sensor-Detector-Module-/150632915292?_trksid=p5197.m7&_trkparms=algo%3DLVI%26itu%3DUCI%26otn%3D4%26po%3DLVI%26ps%3D63%26clkid%3D4001763721082665159

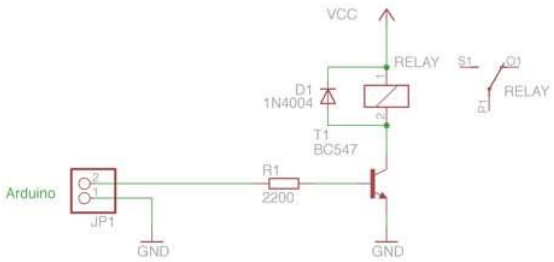
Good sensitivity, large range 7 m. Consumes 60 uA.
Unfortunately it is specified to run from 4.5V and runs unreliably from a single LiPo and needs a 1.5V battery in series with power line to boost voltage.

Another modules
5× Sensor Melder Module PIR Low Voltage Bewegungsmelder
http://www.ebay.de/itm/5-Sensor-Melder-Module-PIR-Low-Voltage-Bewegungsmelder-/270786796651?pt=DE_Haus_Garten_Heimwerker_Sicherheitstechnik&hash=item3f0c269c6b

This is low voltage version that runs from a single LiPo, but range 3 m.

Relay controlled power outlet

Classical circuit.
Diode protects from inductive spike when switching relay off.



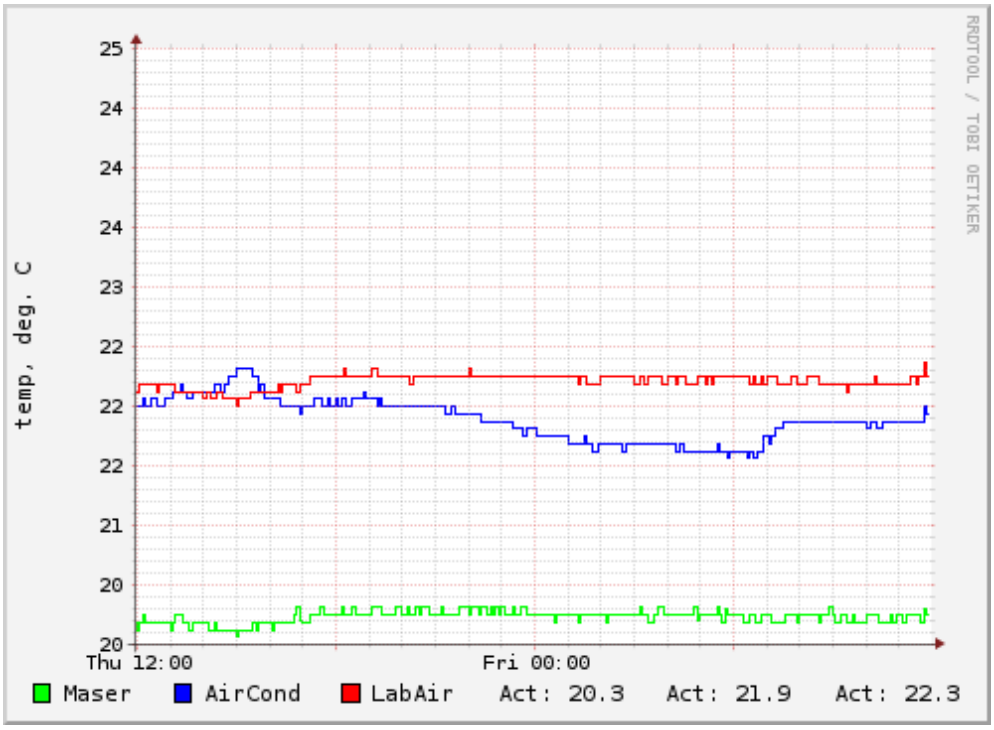
Usually relays are having coil resistance 160 Ohm, but if lucky can get 500 ohm. Small resistance can cause a spike on power line and hang Arduino.
Needs a modification into high voltage part. Dangerous if you don't have qualification.
Shock and fire hazard. Is not CE certified and probably illegal to sale or even building by not crtified person.

```
// Programm for Arduino t read out sensors and to control power outlets. Setup your serial port to have no echo back to Arduino!
#include <OneWire.h>
#include <DallasTemperature.h>

// Data wire is plugged into port 2 on the Arduino
#define ONE_WIRE_BUS 2
#define TEMPERATURE_PRECISION 12

// Setup a oneWire instance to communicate with any OneWire devices (not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

// arrays to hold device addresses
DeviceAddress insideThermometer, outsideThermometer, thirdThermometer;

void setup() {
 Serial.begin(9600);
  sensors.begin();
 if (!sensors.getAddress(insideThermometer, 0)) Serial.println("Unable to find address for Device 0");
 if (!sensors.getAddress(outsideThermometer, 1)) Serial.println("Unable to find address for Device 1");
 if (!sensors.getAddress(thirdThermometer, 2)) Serial.println("Unable to find address for Device 2");
 sensors.setResolution(insideThermometer, TEMPERATURE_PRECISION);
 sensors.setResolution(outsideThermometer, TEMPERATURE_PRECISION);

 pinMode(11, OUTPUT);
 pinMode(12, OUTPUT);
 digitalWrite(11, LOW);
 digitalWrite(12, LOW);
}
long i=0;
String relay="OFF";
String relay2="OFF";


void printAddress(DeviceAddress deviceAddress)
{
 for (uint8_t i = 0; i < 8; i++)
 {
   // zero pad the address if necessary
  // if (deviceAddress[i] < 16) Serial.print("0");
  // Serial.print(deviceAddress[i], HEX);
 }
}

void printTemperature(DeviceAddress deviceAddress)
{
 float tempC = sensors.getTempC(deviceAddress);
 //Serial.println("Temp C: ");
 Serial.print(tempC);
 Serial.print(" ");
}


void printResolution(DeviceAddress deviceAddress)
{
//  Serial.print("Resolution: ");
//  Serial.print(sensors.getResolution(deviceAddress));
//  Serial.println();
}
```

```
void printData(DeviceAddress deviceAddress)
{
 //Serial.print("Device Address: ");
 printAddress(deviceAddress);
 //Serial.print(" ");
 printTemperature(deviceAddress);
 //Serial.println();
}


void loop() {

   sensors.requestTemperatures();
   printData(insideThermometer);
 printData(outsideThermometer);
 printData(thirdThermometer);
String kommand="";
String report="";

while (Serial.available())
 {
 char ch = Serial.read();
 if (ch == 'C')
   {
  for (int x=1; x<8;x++)
     {
     if (Serial.available())
       {
       char ch = Serial.read();
       kommand = kommand + ch;
       }
     }
   }
 }
 if (kommand=="ommand0") {relay="OFF"; digitalWrite(11, LOW);}
 if (kommand=="ommand1") {relay="ON"; digitalWrite(11, HIGH);}
 if (kommand=="ommand2") {relay2="OFF"; digitalWrite(12, LOW);}
 if (kommand=="ommand3") {relay2="ON"; digitalWrite(12, HIGH);}
 if (kommand=="ommandW") i=0;
 if (kommand=="ommandR") i=1990;
report = report+analogRead(A0)+" "+analogRead(A1)+" "+analogRead(A2)+" "+analogRead(A3)+" "+analogRead(A4)+" "+analogRead(A5)+" "+i+"
"+relay+" "+relay2;
Serial.println(report);
delay(500);


if (i>2000) {
 i=0;
 //digitalWrite(12, HIGH);
 //delay(5000);
 //digitalWrite(12, LOW);
}

i++;

}
```

Ubuntu

Read temperature from Arduino and graph using RRDtool

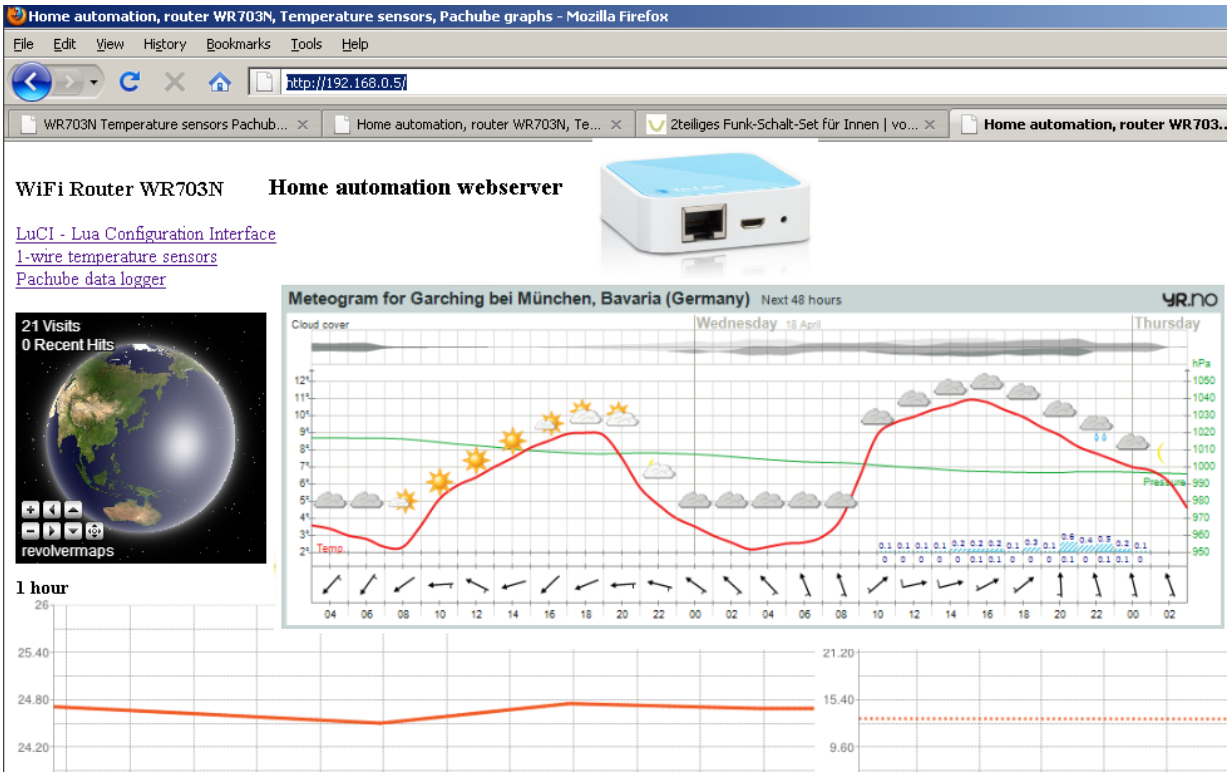



```
# Code for UBUNTU
#stty -F /dev/ttyUSB0 raw speed 9600
cd /var/www
while true
do
a=""
a=$(head -1 /dev/ttyUSB0)
#a=$(head -1 /dev/ttyUSB0)

echo "this is read from Arduino "$a
if [ -n "$a" ]
then
echo "not empty"
break
else
echo "empty"
fi
done

set -- $a
t1=$1
t2=$2
t3=$3
echo $t1 $t2 $t3

t1=$(echo "scale=3; $t1+0" | bc)
t2=$(echo "scale=3; $t2+0" | bc)
t3=$(echo "scale=3; $t3+0" | bc)
E=1.110
t=$t1":"$t2":"$t3":"$E;
echo $t
```

```
rrdtool update /var/www/t.rrd --template t1:t2:t3:t4 N:$t;
rrdtool fetch /var/www/t.rrd AVERAGE

rrdtool graph /var/www/webcam/t1.png --start -1d --end now  --vertical-label "temp, deg. C"  \
-w 400 -h 300 \
DEF:average=/var/www/t.rrd:t1:AVERAGE LINE1:average#00FF00:"Maser"     \
DEF:average1=/var/www/t.rrd:t2:AVERAGE LINE1:average1#0000FF:"AirCond"     \
DEF:average2=/var/www/t.rrd:t3:AVERAGE LINE1:average2#FF0000:"LabAir" \
GPRINT:average:LAST:"Act\: %3.1lf" \
GPRINT:average1:LAST:"Act\: %3.1lf" \
GPRINT:average2:LAST:"Act\: %3.1lf"

#rrdtool create t.rrd --step 300  \
#DS:t1:GAUGE:600:-20:100  \
#DS:t2:GAUGE:600:-20:100  \
#DS:t3:GAUGE:600:-20:100  \
#DS:t4:GAUGE:600:-20:100  \
#RRA:AVERAGE:0.5:1:2016  \
#RRA:AVERAGE:0.5:6:1344  \
#RRA:AVERAGE:0.5:24:2190  \
#RRA:AVERAGE:0.5:144:3650
```

OpenWRT

Please visit my Instructable how to set up webserver on WR703N router:
http://www.instructables.com/id/How-to-set-up-OpenWRT-on-a-pocket-router-WR703N/




```
# read Arduino analog input value

/usr/bin/stty -F /dev/ttyUSB0  speed 9600 -echo

a=""
a=$(head -1 /dev/ttyUSB0)
#echo $a
set -- $a
t1=$5
t1=$(($t1 + 0))
echo " light level: " $t1
```

```
# send command to switch relay
a=$(cat remotecontrol.dat)

if [ $a == "0" ]; then
echo "Komanda OFF 0"
echo "Command0" > /dev/ttyUSB0
/www/webcam/stream.sh 1
fi
if [ $a == "1" ]; then
echo "Komanda ON 1"
echo "Command1" > /dev/ttyUSB0
/www/webcam/stream.sh 1
fi
```