LAMPIRAN

Programable

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27 ,2,1,0,4,5,6,7,3, POSITIVE);

#define RELAY_ON 0
#define RELAY_OFF 1
#include "DHT.h"
#define DHTPIN_1 9     // pin for data
#define DHTPIN_2 7
#define DHTTYPE DHT22   // type  sensor

DHT dht1(DHTPIN_1, DHTTYPE); // initializes sensor
DHT dht2(DHTPIN_2, DHTTYPE); // initializes sensor

int RELAY_1 = 8;
int RELAY_2 = 12;
int led_pin = 13;
int chk;
float hum1;           // stores humidity
float suhu, suhu_2;;  // stores temperature
float hum2;           //   stores humidity

// Program: Sound Level Measurement

int num_Measure = 128 ;   // Set the number of measurements
int pinSignal = A0;       // pin connected to pin O module sound sensor
long Sound_signal;        // Store the value read Sound Sensor
long sum = 0 ;            // Store the total value of n measurements
long level = 0 ;          // Store the average value

//#include <RBDdimmer.h>//
//#define zerocross  2    // for boards with CHANGEBLE input pin
int AC_LOAD = 3;          // Output to Opto Triac pin
int dimming=128;          // Dimming level (0-255)  0 = ON, 255 = OFF

float keluaran1;
float a,b,c;
float member_suhu;
float member_kelembaban;
float dingin, sejuk, normal, panas, sangatpanas;
float dingin2, sejuk2, normal2, panas2, sangatpanas_2;
float min1, min2, min3, min4, min5, min6, min7, min8, min9, min10,
min11,min12, min13, min14, min15, min16, min17, min18, min19, min20,
min21,min22, min23, min24, min25;
```

```
int PD=0, PL=65, NR=125, CP=175, SC=255;
float A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V,W, X, Y;
float output=0;
float crispy;
unsigned long interval=1000;      // the time we need to wait
unsigned long previousMillis=0;   // millis() returns an unsigned long.

// definis pin unutuk relay heatern dan valve juga led
#define temppeste A3
#define umidpeste A2

// definisi variable
float te, t1, tes, dete;      // variable untuk temperature
int has, hass, dehas;        // variable untuk humiditiy

#define BUT1 6    // - switch
#define BUT2 11    // + switch
#define BUT3 10    // MENU switch

byte meniu = 0; // if MENIU = 0 untuk clasical
               // if MENIU = 1 untuk temperature set (tes)
               // if MENIU = 2 untuk dt temperature (dete)
               // if MENIU = 3 untuk humidity set (hass)
               // if MENIU = 4 untuk dh humidity (dehas)
byte grad[8] = {
  B01100,
  B10010,
  B10010,
  B01100,
  B00000,
  B00000,
  B00000,
};
void setup()
{
  Serial.begin(9600);
// Initialize LCD display
 lcd.begin(20, 4);
// create custom simbol
lcd.createChar(0, grad);


// definisi output
 pinMode(temppeste, OUTPUT);
 pinMode(umidpeste, OUTPUT);

// atur status default untuk output
  digitalWrite(temppeste, LOW);
  digitalWrite(umidpeste, LOW);

// mengatur tombol untuk menu
  pinMode(BUT1, INPUT);
  pinMode(BUT2, INPUT);
  pinMode(BUT3, INPUT);

  digitalWrite(BUT1, HIGH); // pull-up aktif
  digitalWrite(BUT2, HIGH);
  digitalWrite(BUT3, HIGH);
  lcd.setCursor(0,0);
  lcd.print("monitoring");

  lcd.setCursor(0,1);
  lcd.print("inkubator bayi");

  lcd.setCursor(0,2);
  lcd.print("set poin suhu");

  lcd.setCursor(0,3);
  lcd.print("dan kelembaban ");

delay(3000);
lcd.clear();

// nilai yang telah ditetapkan untuk suhu dan kelembaban
tes = 35;
dete = 0.2;
hass = 65;
dehas = 2;

  Serial.begin(9600);
  dht1.begin();
  dht2.begin();
  pinMode (pinSignal, INPUT); // Tetapkan pin sinyal sebagai input
  pinMode(led_pin,OUTPUT);
 // Set pin output.
  pinMode(RELAY_1, OUTPUT);
   // Set pin output.
  pinMode(RELAY_2, OUTPUT);
// Inisialisasi relai sebagai mati sehingga saat disetel ulang akan mati secara
default
  digitalWrite(RELAY_1, RELAY_OFF);
  digitalWrite(RELAY_2, RELAY_ON);
  digitalWrite(13, HIGH);

  lcd.begin(20, 4);
  lcd.setCursor(2,0);
  lcd.print("Di Buat Oleh");
  lcd.setCursor(2,1);
  lcd.print("kuat ");
  delay(3000);
  lcd.clear();


pinMode(AC_LOAD, OUTPUT);// Tetapkan pin Muat AC sebagai output // pwm
attachInterrupt(0, zero_crosss_int, FALLING); //pin d2
Serial.begin(9600);

}
void zero_crosss_int()
{

  int dimtime = (35*dimming);    // untuk 60Hz =>65
  delayMicroseconds(dimtime);    // Off cycle
  digitalWrite(AC_LOAD, HIGH);   // triac firing
  delayMicroseconds(10);
  digitalWrite(AC_LOAD, LOW);    // triac Off

}
void loop()

{


   unsigned long currentMillis = millis(); // grab current time
   if ((unsigned long)(currentMillis - previousMillis) >= interval) {
  hum1 = dht1.readHumidity();            // membaca dan menyimpan hum
  suhu= dht1.readTemperature();          // membaca dan menyimpan temp
  hum2 = dht2.readHumidity();            // membaca dan menyimpan hum
  suhu_2= dht2.readTemperature();        // membaca dan menyimpan temp

keluaran1 = output;
dimming=keluaran1;
crispy = map(dimming, 0,255,255,0);
membership();
defuzifikasi();
Output();
tampilan();
previousMillis = millis();

// baca input pada pin analog 0
  int sensorValue = analogRead(A0);

   for ( int i = 0 ; i <num_Measure; i ++)
  {
   Sound_signal = analogRead (pinSignal);
    sum =sum + Sound_signal;
  }

  level = sum / num_Measure;  // Hitung nilai rata-rata
  sum = 0 ;                   // Atur ulang jumlah nilai pengukuran
  delay(50);

 // perintah pada relay valve
  if (sensorValue <58)// ON
{
  digitalWrite(RELAY_1, RELAY_ON);
;
  digitalWrite(13, HIGH);
}
 else if (sensorValue >59)//OFF
{
  digitalWrite(RELAY_1, RELAY_OFF);

  digitalWrite(13, LOW);
}
  if (hum2 <70)// ON
{
  digitalWrite(RELAY_2, RELAY_ON);
}
 else if (hum2 >71)//OFF
{
  digitalWrite(RELAY_2, RELAY_OFF);
}
//menampilkan informasi kelcd
  lcd.setCursor(0,2);
  lcd.print("Hum:");
  lcd.print(hum1);
  lcd.setCursor(10,2);
  lcd.print("C Out:");
  lcd.print(suhu_2);
  lcd.setCursor(10,3);
  lcd.print("C In :");
  lcd.print(suhu);
  lcd.setCursor(0,3);
  lcd.print(" dB: ");
  lcd.print(sensorValue);

 }
//========================program
setpoin====================================
if (digitalRead(BUT3) == LOW)
```

```
{ meniu = meniu + 1;
lcd.clear();
delay(250);
}
if (meniu >= 5) meniu = 0;

if (meniu == 0)
{
 if (digitalRead(BUT3) == LOW)
  { meniu = 1;
  delay(250);
  }

// Suhu atau kelembaban membaca
 has = dht1.readHumidity();
 te = dht1.readTemperature();
delay(500);

 lcd.setCursor(0, 0);
  // lcd.print("t=");
 if (te < 0)
 {t1=-te;}
 else t1=te;

    if ( t1 < 10)
   {
     lcd.print(" ");
   }
   if (te>0) lcd.print("+");
   if (te==0) lcd.print(" ");
   if (te<0) lcd.print("-");
   lcd.print(t1,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");

   lcd.setCursor(12, 0);
  // lcd.print("H=");
   lcd.print(has);
   lcd.print("%RH");


if (te > tes)
 {
 digitalWrite(temppeste, LOW);
   lcd.setCursor(0, 1);
   lcd.print("melebihi ");
 }
```

```
if (tes - dete > te)
  {
   digitalWrite(temppeste, HIGH);
   lcd.setCursor(0, 1);
   lcd.print("kurang");
 }
if ((te <= tes) and (tes - dete <= te))
 {
   lcd.setCursor(0, 1);
   lcd.print("pas !  ");
 }


 if (has > hass)
 {
 digitalWrite(umidpeste, HIGH);
   lcd.setCursor(12, 1);
   lcd.print("melebihi ");
 }
 if (has < hass - dehas)
 {
   digitalWrite(umidpeste, LOW);
   lcd.setCursor(12, 1);
   lcd.print("kurang");
 }
 if ((has <= hass) and (has >= hass - dehas))
 {  lcd.setCursor(12, 1);
   lcd.print("pas !  ");
}
} // last line for MENIU = 0


if (meniu == 1) {
  while (meniu == 1) {
//   teset(tset);
   lcd.setCursor(0, 0);
   lcd.print("Temperature SET:");
   lcd.setCursor(0, 1);
//   lcd.print("+");
   lcd.print(tes,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");

   lcd.setCursor(8, 1);
   lcd.print("dt=");
   lcd.print(dete,1);
//   lcd.write(0b11011111);
```

```
   lcd.write(byte(0));
   lcd.print("C  ");


   if (digitalRead(BUT1) == LOW)
  { tes = tes - 0.1;
  delay(250);
  }
   if (digitalRead(BUT2) == LOW)
  { tes = tes + 0.1;
  delay(250);
  }
  if (digitalRead(BUT3) == LOW)
  { meniu = 2;
  delay(250);
  lcd.clear();
  }
//  delay(15);
    }
   delay (100);
}   // end loop for MENIU = 1


if (meniu == 2) {
 // if (tmax <= tmin) tmax = tmin + 10;
   while (meniu ==2) {
   lcd.setCursor(0, 0);
   lcd.print("hyst.temp. SET:");
   lcd.setCursor(0, 1);
//   lcd.print("+");
   lcd.print(tes,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");

   lcd.setCursor(8, 1);
   lcd.print("dt=");
   lcd.print(dete,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");

    if (digitalRead(BUT1) == LOW)
   { dete = dete - 0.1;
   delay(250);
   }
   if (digitalRead(BUT2) == LOW)
   { dete = dete + 0.1;
   delay(250);
  }
   if (digitalRead(BUT3) == LOW)
   { meniu = 3;
   delay(250);
   lcd.clear();
  }
//   delay(15);
if (dete < 0.1) dete = 0.1;
    }
}  // end loop for MENIU = 2


if (meniu == 3) {
  while (meniu == 3) {
//   teset(tset);
   lcd.setCursor(0, 0);
   lcd.print("Humidity SET:");
   lcd.setCursor(0, 1);
//   lcd.print("+");
   lcd.print(hass,1);
//   lcd.write(0b11011111);
//   lcd.write(byte(0));
   lcd.print("%RH  ");

   lcd.setCursor(7, 1);
   lcd.print("dh=");
   lcd.print(dehas);
//   lcd.write(0b11011111);
//   lcd.write(byte(0));
   lcd.print("%RH");


   if (digitalRead(BUT1) == LOW)
  { hass = hass - 1;
  delay(250);
  }
   if (digitalRead(BUT2) == LOW)
  { hass = hass + 1;
  delay(250);
  }
  if (digitalRead(BUT3) == LOW)
  { meniu = 4;
  delay(250);
  lcd.clear();
  }
//  delay(15);
     }
   delay (100);
}  // end loop for MENIU =3


if (meniu == 4) {
   while (meniu ==4) {
   lcd.setCursor(0, 0);
   lcd.print("hyst.humid. SET:");
   lcd.setCursor(0, 1);
//   lcd.print("+");
   lcd.print(hass);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("%RH");

   lcd.setCursor(7, 1);
   lcd.print("dh=");
   lcd.print(dehas);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("%RH");


   if (digitalRead(BUT1) == LOW)
   { dehas = dehas - 1;
   delay(250);
  }
   if (digitalRead(BUT2) == LOW)
   { dehas = dehas + 1;
   delay(250);
  }
   if (digitalRead(BUT3) == LOW)
   { meniu = 5;
   delay(250);
   lcd.clear();
  }
//   delay(15);

if (meniu == 5) {
  while (meniu == 5) {
//   teset(tset);
   lcd.setCursor(0, 0);
   lcd.print("Temperature SET:");
   lcd.setCursor(0, 1);
//   lcd.print("+");
   lcd.print(tes,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");

   lcd.setCursor(8, 1);
   lcd.print("dt=");
   lcd.print(dete,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");


   if (digitalRead(BUT1) == LOW)
  { tes = tes - 0.1;
  delay(250);
  }
   if (digitalRead(BUT2) == LOW)
  { tes = tes + 0.1;
  delay(250);
  }
  if (digitalRead(BUT3) == LOW)
  { meniu = 6;
  delay(250);
  lcd.clear();
  }
//  delay(15);
     }
   delay (100);
}   // end loop for MENIU = 1

if (meniu == 6) {
 // if (tmax <= tmin) tmax = tmin + 10;
   while (meniu ==6) {
 //    maxim(dt/10); // this is number to diplay
 //    dete(dt); // this is number to diplay
   lcd.setCursor(0, 0);
   lcd.print("hyst.temp. SET:");
   lcd.setCursor(0, 1);
//   lcd.print("+");
   lcd.print(tes,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");

   lcd.setCursor(8, 1);
   lcd.print("dt=");
   lcd.print(dete,1);
//   lcd.write(0b11011111);
   lcd.write(byte(0));
   lcd.print("C  ");
```

```
    if (digitalRead(BUT1) == LOW)
   { dete = dete - 0.1;
   delay(250);
  }
   if (digitalRead(BUT2) == LOW)
   { dete = dete + 0.1;
   delay(250);
  }
   if (digitalRead(BUT3) == LOW)
   { meniu = 7;
   delay(250);
   lcd.clear();
  }
//   delay(15);
if (dete < 0.1) dete = 0.1;
    }
}  // end loop for MENIU = 2

if (dehas < 1) dehas = 1;
    }

}

}
```

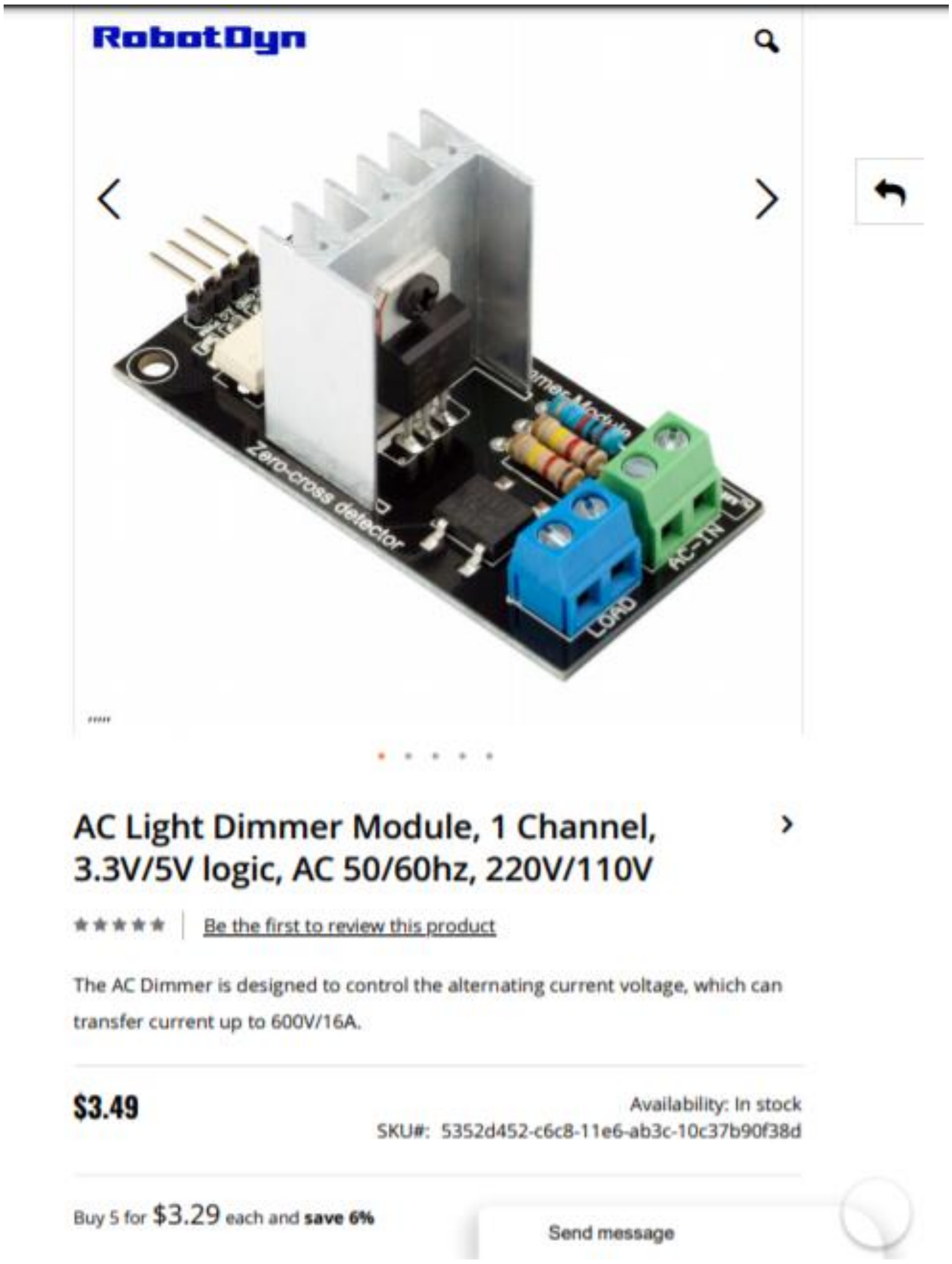
```
//========================================================================================
//=================================defuzifikasi=================================
void defuzifikasi()
{
A = min1*PD; B = min2*PD; C = min3*PL; D = min4*NR; E = min5*CP;
F = min6*PD; G = min7*PL; H = min8*PL; I = min9*NR; J = min10*CP;
K = min11*PL; L = min12*PL; M = min13*NR; N = min14*CP; O = min15*SC;
P = min16*NR; Q = min17*NR; R = min18*CP; S = min19*SC; T = min20*SC;
U = min21*CP; V = min22*CP; W = min23*SC; X = min24*SC; Y = min25*SC;
output = (A + B + C + D + E + F + G + H + I + J + K + L + M + N + O + P + Q +
R + S + T + U + V + W + X + Y )/(min1 + min2 + min3 + min4 + min5+ min6 +
min7 + min8 + min9 + min10 + min11 + min12 + min13 + min14 + min15 +
min16 + min17 + min18 + min19 + min20 + min21 + min22 + min23 + min24 +
min25);
}

//=====================tampilan pada serial monitor dan
BT==========================
void tampilan()
{
 /* Serial.print("DHT22_1: ");
  Serial.print("hum1 = ");
  Serial.print(hum1);
  delay(1000);*/
  Serial.print("suhu dalam= ");
  Serial.print(suhu);
  Serial.println(" C");
  delay(1000);
  /*Serial.print("DHT22_2: ");
  Serial.print("hum2 = ");
  Serial.print(hum2);
  delay(1000);*/
  Serial.print("suhu luar= ");
  Serial.print(suhu_2);
  Serial.println(" C");
  delay(1000);
  Serial.print("suhu dalam= ");
  Serial.print(suhu);
  Serial.println(" C");
  delay(1000);

    Serial.print("sound: ");
    Serial.print(level);
    Serial.println(" dB");
    //Serial.print(" | ");
    delay(1000);
    /*Serial.print("hyst C: ");
    Serial.print(dete);
    delay(1000);
    Serial.print("|setpoin C: ");
    Serial.print(tes);
      delay(1000);
    Serial.print("|hyst %: ");
    Serial.print(dehas);
      delay(1000);
    Serial.print("|setpoin %: ");
    Serial.print(hass);
      delay(1000);
    /*Serial.print("|temperature C: ");
    Serial.print(te);
      delay(1000);*/
    Serial.print("kelembaban %: ");
    Serial.println(has);
     delay(1000);

/*Serial.print("SUHU 1= ");
Serial.print(suhu);
Serial.print("*C");
  delay(1000);
Serial.print(" | ");
Serial.print("SUHU 2= ");
Serial.print(suhu_2);
Serial.print("*C");
  delay(1000);
*/
//Serial.print(" | ");
Serial.print("Fuzzy output:");
Serial.print( output);
Serial.println(" ");
delay(1000);
Serial.print("Crisp output:");
Serial.print( crispy);
Serial.println(" ");
delay(1000);

}

//========================================= fuzzyfikasi
=========================================
void membership()
{

//fuzzyfikasi
```

```
//SENSOR suhu1
member_suhu = 0;
Sensor_1(a=0, b=30, c=35); //Buat Dingin
dingin = member_suhu;
Sensor_1(a=30, b=35, c=40); //Buat Sejuk
sejuk = member_suhu;
Sensor_1(a=35, b=40, c=45); //Buat Normal
normal = member_suhu;
Sensor_1(a=40, b=45, c=50); //Buat Panas
panas = member_suhu;
Sensor_1(a=45, b=50, c=55); //Buat Sangat Panas

sangatpanas = member_suhu;
//SENSOR kelembaban
member_kelembaban = 0;
Sensor_1(a=0, b=15, c=35); //Buat sangat kering
dingin2 = member_kelembaban;
Sensor_1(a=15, 35, c=45); //Buat kering
sejuk2 = member_kelembaban;
Sensor_1(a=35, b=45, c=65); //Buat normal
normal2 = member_kelembaban;
Sensor_1(a=45, b=65, c=70); //Buat lembab
panas2 = member_kelembaban;
Sensor_1(a=65, b=70, c=80); //Buat Sangat lembab
sangatpanas_2 = member_kelembaban;
//fuzzyfikasi
min1 = min(dingin, dingin2);
min2 = min(dingin, sejuk2);
min3 = min(dingin, normal2);
min4 = min(dingin, panas2);
min5 = min(dingin, sangatpanas_2);
min6 = min(sejuk, dingin2);
min7 = min(sejuk, sejuk2);
min8 = min(sejuk, normal2);
min9 = min(sejuk, panas2);
min10 = min(sejuk, sangatpanas_2);
min11 = min(normal, dingin2);
min12 = min(normal, sejuk2);
min13 = min(normal, normal2);
min14 = min(normal, panas2);
min15 = min(normal, sangatpanas_2);
min16 = min(panas, dingin2);
min17 = min(panas, sejuk2);
min18 = min(panas, normal2);
min19 = min(panas, panas2);
min20 = min(panas, sangatpanas_2);
min21 = min(sangatpanas, dingin2);
min22 = min(sangatpanas, sejuk2);
```

```
min23 = min(sangatpanas, normal2);
min24 = min(sangatpanas, panas2);
min25 = min(sangatpanas, sangatpanas_2);
}
```

```
// ================================ Rule Base
=========================================================
===============
void Output ()
{

if (output < 32) { Serial.print("SANGAT DINGIN");}
if (output >= 33 && output < 65) {Serial.print("DINGIN");}
if (output >= 65 && output < 125) {Serial.print("AGAK HANGAT");}
if (output >= 125 && output < 142) {Serial.print("HANGAT");}
if (output >= 142 && output < 256) {Serial.print("PANAS");}
//if (output > 255) { output = 255; Serial.print("SANGAT CEPAT ");}
Serial.println();
}
//================= Rumus keanggotaan
=========================================================
==
float Sensor_1(float a, float b, float c)
{
if ((suhu >= a) && (suhu < b))
{
member_suhu = (suhu - a) / (b - a);
}
if ((suhu >= b) && (suhu < c))
{
member_suhu = (c - suhu) / (c - b);
}
if ((suhu < 0) || (suhu > 100))
{
member_suhu = 1;
}
if ((suhu > c) || (suhu < a))
{
member_suhu = 0;
}
if ((hum1>= a) && (hum1< b))
{
member_kelembaban = (hum1- a) / (b - a);
}
if ((hum1>= b) && (hum1< c))
{
member_kelembaban = (c - hum1) / (c - b);
}
if ((hum1< 0) || (hum1> 100))
{
member_kelembaban = 1;
}
if ((hum1> c) || (hum1< a))
{
member_kelembaban = 0;
}
}
```

AC Light Dimmer Modul



AC Light Dimmer Module, 1 Channel, 3.3V/5V logic, AC 50/60hz, 220V/110V

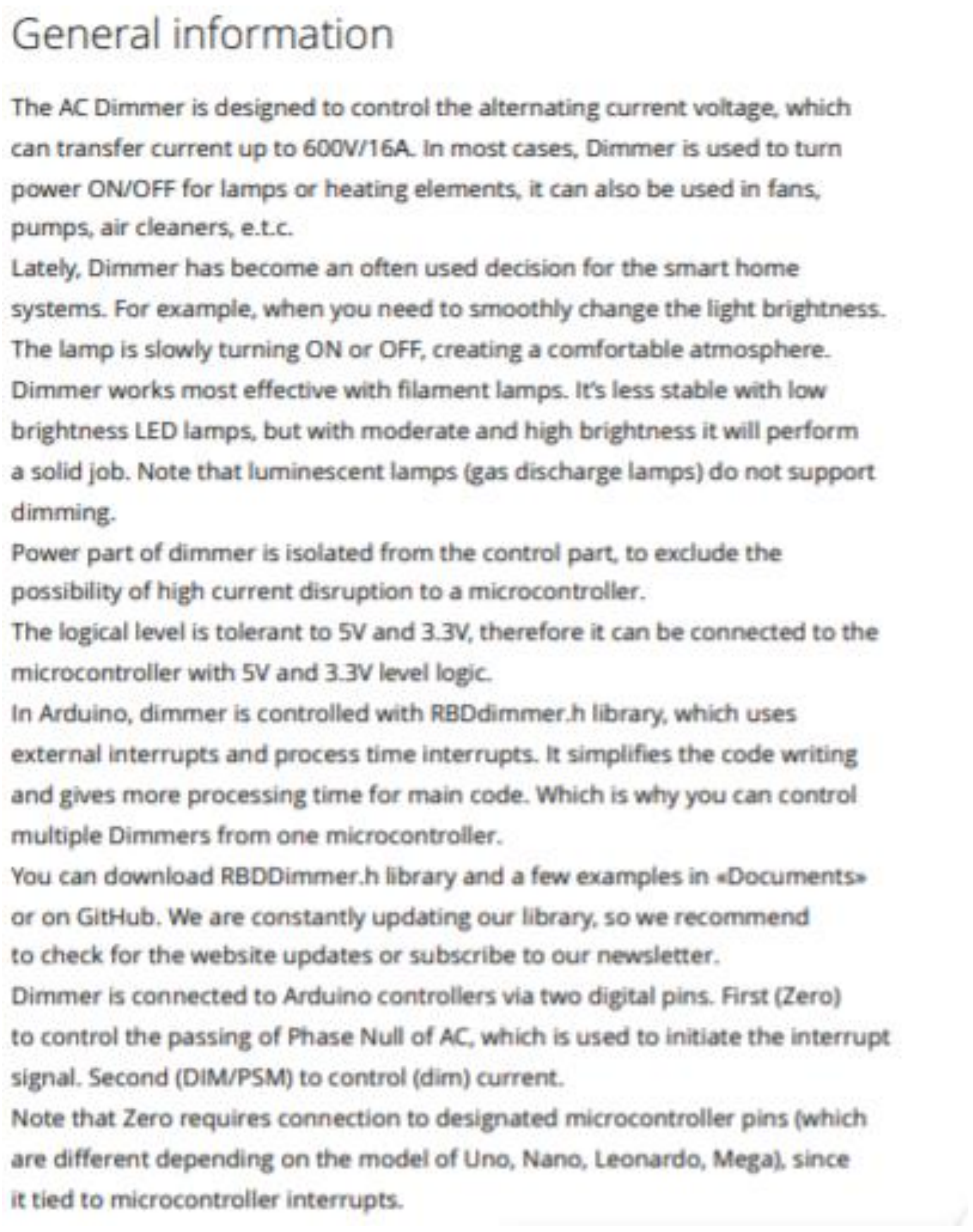
Be the first to review this product

The AC Dimmer is designed to control the alternating current voltage, which can transfer current up to 600V/16A.

$3.49

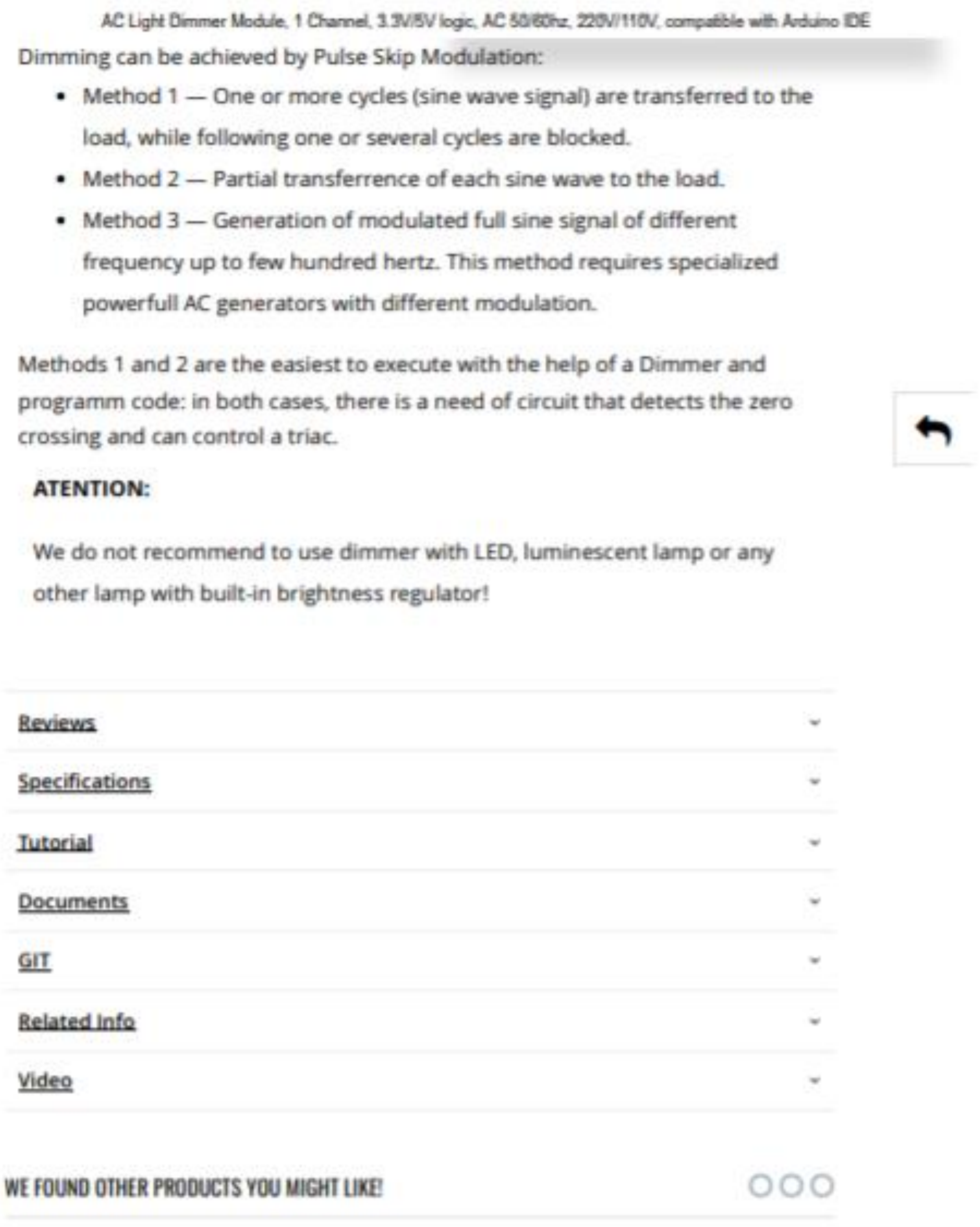
Availability: In stock
SKU#: 5352d452-c6c8-11e6-ab3c-10c37b90f38d

Buy 5 for $3.29 each and **save 6%**



General information

The AC Dimmer is designed to control the alternating current voltage, which can transfer current up to 600V/16A. In most cases, Dimmer is used to turn power ON/OFF for lamps or heating elements, it can also be used in fans, pumps, air cleaners, e.t.c.
Lately, Dimmer has become an often used decision for the smart home systems. For example, when you need to smoothly change the light brightness. The lamp is slowly turning ON or OFF, creating a comfortable atmosphere.
Dimmer works most effective with filament lamps. It's less stable with low brightness LED lamps, but with moderate and high brightness it will perform a solid job. Note that luminescent lamps (gas discharge lamps) do not support dimming.
Power part of dimmer is isolated from the control part, to exclude the possibility of high current disruption to a microcontroller.
The logical level is tolerant to 5V and 3.3V, therefore it can be connected to the microcontroller with 5V and 3.3V level logic.
In Arduino, dimmer is controlled with RBDdimmer.h library, which uses external interrupts and process time interrupts. It simplifies the code writing and gives more processing time for main code. Which is why you can control multiple Dimmers from one microcontroller.
You can download RBDDimmer.h library and a few examples in «Documents» or on GitHub. We are constantly updating our library, so we recommend to check for the website updates or subscribe to our newsletter.
Dimmer is connected to Arduino controllers via two digital pins. First (Zero) to control the passing of Phase Null of AC, which is used to initiate the interrupt signal. Second (DIM/PSM) to control (dim) current.
Note that Zero requires connection to designated microcontroller pins (which are different depending on the model of Uno, Nano, Leonardo, Mega), since it tied to microcontroller interrupts.



AC Light Dimmer Module, 1 Channel, 3.3V/5V logic, AC 50/60hz, 220V/110V, compatible with Arduino IDE

Dimming can be achieved by Pulse Skip Modulation:

- Method 1 — One or more cycles (sine wave signal) are transferred to the load, while following one or several cycles are blocked.
- Method 2 — Partial transferrence of each sine wave to the load.
- Method 3 — Generation of modulated full sine signal of different frequency up to few hundred hertz. This method requires specialized powerfull AC generators with different modulation.

Methods 1 and 2 are the easiest to execute with the help of a Dimmer and programm code: in both cases, there is a need of circuit that detects the zero crossing and can control a triac.

ATENTION:

We do not recommend to use dimmer with LED, luminescent lamp or any other lamp with built-in brightness regulator!

Reviews

Specifications

Tutorial

Documents

GIT

Related Info

Video

WE FOUND OTHER PRODUCTS YOU MIGHT LIKE!

Analog Slide Position - Potenti... Rotation potentiometer (anal... Buttor

$1.39 $0.99 As

RobotDyn

CONTACT INFORMATION

CHINA
2 East Gate, 6/B, 5th East Nanping Rd., Nanping Industrial Park, Xiangzhou District, Zhuhai, GD, China
PHONE
+86-756-3358590
RUSSIA
Varshavskoe shosse, 42, Moscow, Russia
PHONE
+7-499-110-24-82

SUBSCRIBE TO NEWSLETTER

Stay for updates.

Sound sensor Arduino

Tuesday, April 19, 2016

Sound sensor Arduino

Tutorial on how to interface sound sensor with Arduino. More precisely, Sound Sensor. I will show how to detect ambient sound and how to handle the signal generated by this module. This board along with the microphone, has a small built-in amplifier (integrated circuit LM386), because only the microphone would not be able to send data for Arduino. The connection scheme is very clean, composed of only 3 pins: Vcc, GND and S (signal). In the middle of the plate, there is a potentiometer for sensitivity adjustment.

The board works with 5V voltage, and the signal pin should be connected preferably to an analog port of Arduino, since the generated signal is variable, and thus we can see the different levels of noise picked up by the microphone. The circuit tests will consist of the module Sound Sensor , plus the display 16x2 LCD , which we've covered here on the site, in this article . If you do not have this display, do not worry because the program is easily adaptable to other models of LCD displays . The display at its top, will show the sound level (Low, Medium and High), and bottom , a bar that will follow in real time the sound level detected by the microphone will be shown:

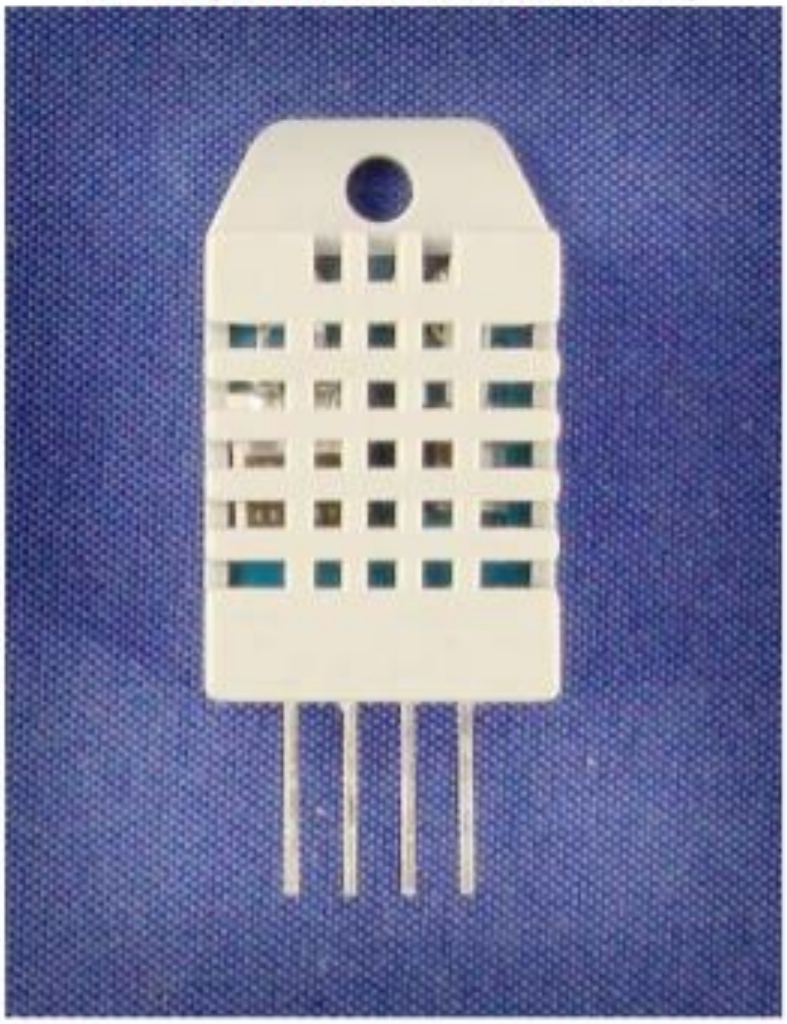
Sensor DHT 22 (DHT 22 also named as AM 2302

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

Digital-output relative humidity & temperature sensor/module

DHT22 (DHT22 also named as AM2302)

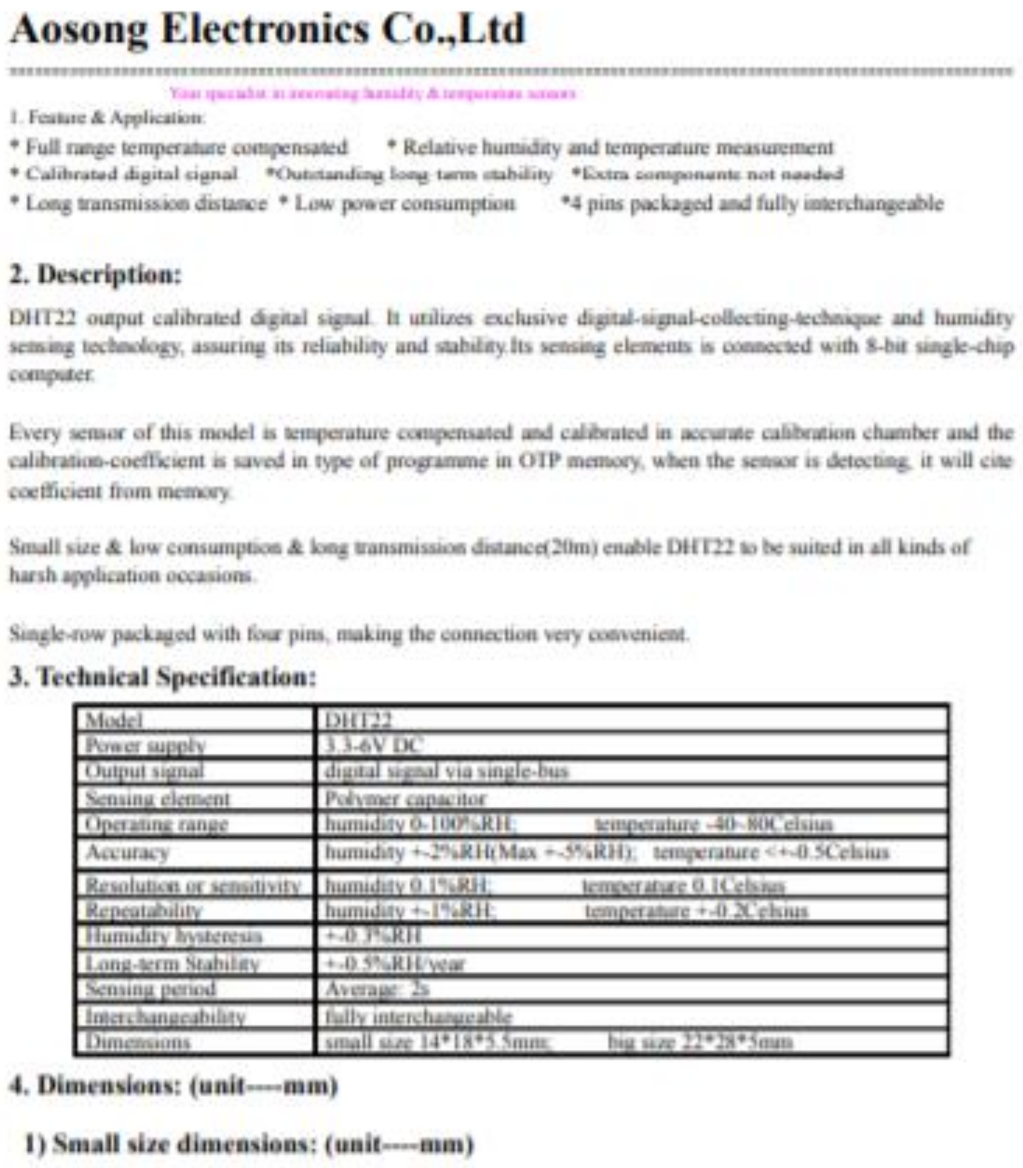


Capacitive-type humidity and temperature module/sensor

1

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn



Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

1. Feature & Application:

* Full range temperature compensated * Relative humidity and temperature measurement
* Calibrated digital signal *Outstanding long term stability *Extra components not needed
* Long transmission distance * Low power consumption *4 pins packaged and fully interchangeable

2. Description:

DHT22 output calibrated digital signal. It utilizes exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability.Its sensing elements is connected with 8-bit single-chip computer.

Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory.

Small size & low consumption & long transmission distance(20m) enable DHT22 to be suited in all kinds of harsh application occasions.

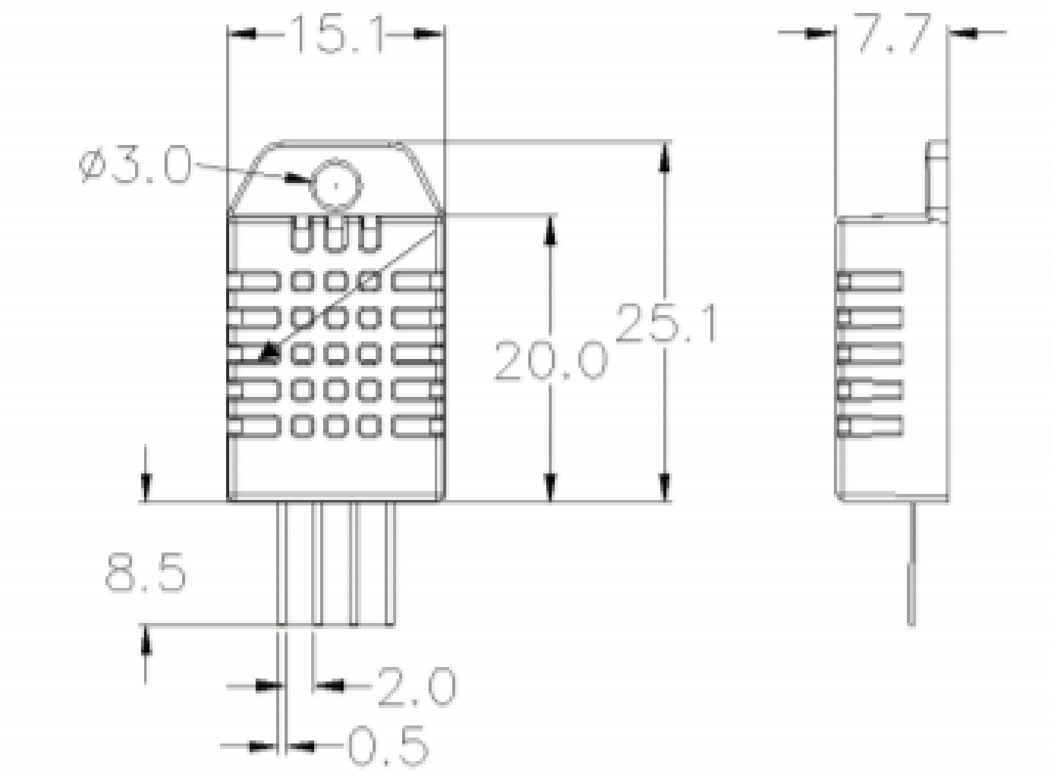
Single-row packaged with four pins, making the connection very convenient.

3. Technical Specification:

Model	DHT22
Power supply	3.3-6V DC
Output signal	digital signal via single-bus
Sensing element	Polymer capacitor
Operating range	humidity 0-100%RH; temperature -40~80Celsius
Accuracy	humidity +-2%RH(Max +-5%RH); temperature <+-0.5Celsius
Resolution or sensitivity	humidity 0.1%RH; temperature 0.1Celsius
Repeatability	humidity +-1%RH; temperature +-0.2Celsius
Humidity hysteresis	+-0.3%RH
Long-term Stability	+-0.5%RH/year
Sensing period	Average: 2s
Interchangeability	fully interchangeable
Dimensions	small size 14*18*5.5mm; big size 22*28*5mm

4. Dimensions: (unit----mm)

1) Small size dimensions: (unit----mm)



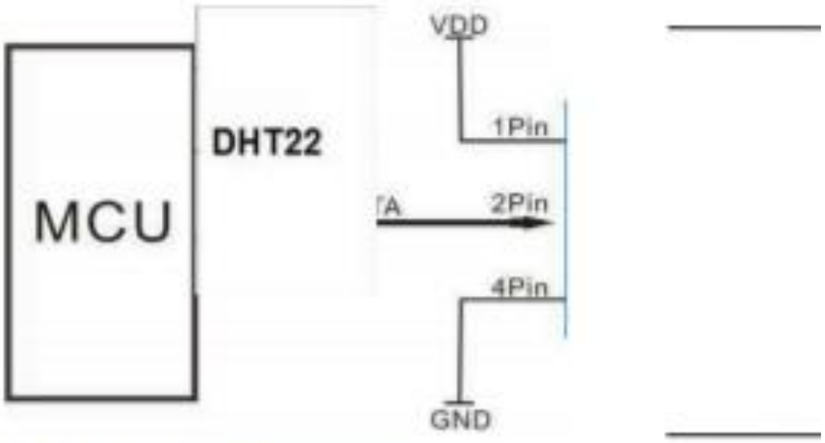
Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD----power supply
2	DATA--signal
3	NULL
4	GND

Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

5. Electrical connection diagram:



3Pin---NC, AM2302 is another name for DHT22

6. Operating specifications:

(1) Power and Pins

Power's voltage should be 3.3-6V DC. When power is supplied to sensor, don't send any instruction to the sensor within one second to pass unstable status. One capacitor valued 100nF can be added between VDD and GND for wave filtering.

(2) Communication and signal

Single-bus data is used for communication between MCU and DHT22, it costs 5mS for single time communication.

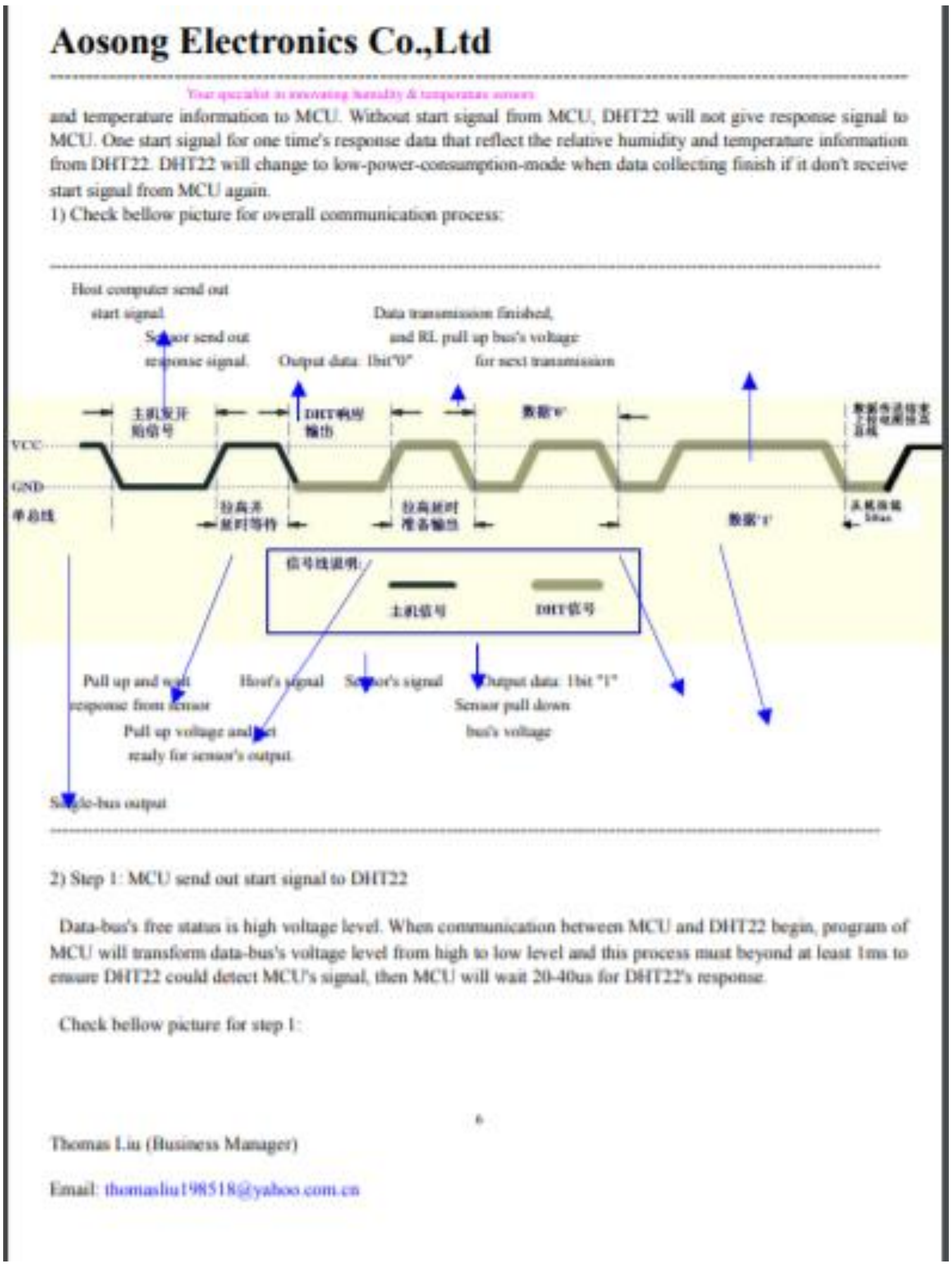
Data is comprised of integral and decimal part, the following is the formula for data.

DHT22 send out higher data bit firstly!
DATA=8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data+8 bit check-sum
If the data transmission is right, check-sum should be the last 8 bit of "8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data".

When MCU send start signal, DHT22 change from low-power-consumption-mode to running-mode. When MCU finishs sending the start signal, DHT22 will send response signal of 40-bit data that reflect the relative humidity

Thomas Liu (Business Manager)

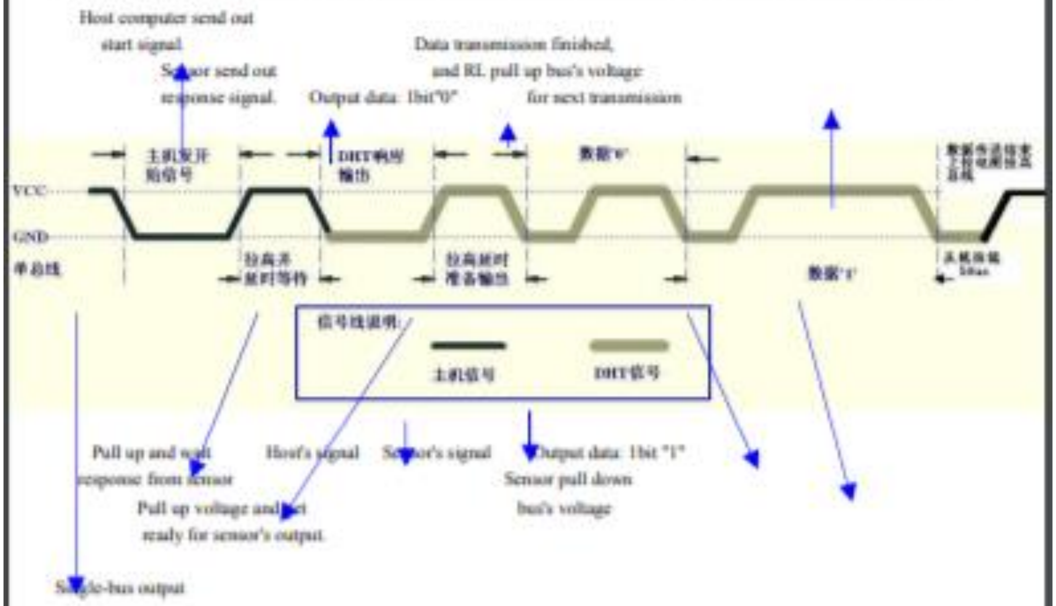
Email: thomasliu198518@yahoo.com.cn



Aosong Electronics Co.,Ltd

and temperature information to MCU. Without start signal from MCU, DHT22 will not give response signal to MCU. One start signal for one time's response data that reflect the relative humidity and temperature information from DHT22. DHT22 will change to low-power-consumption-mode when data collecting finish if it don't receive start signal from MCU again.

1) Check bellow picture for overall communication process:


Host computer send out start signal
Sensor send out response signal.
Output data: 1bit"0"
Data transmission finished, and RL pull up bus's voltage for next transmission
主机发开始信号
DHT响应输出
数据'0'
数据传送结束上拉电阻拉高总线
VCC
GND
单总线
拉高并延时等待
拉高延时准备输出
数据'1'
主机释放 20us
信号线说明:
主机信号
DHT信号
Pull up and wait response from sensor
Host's signal
Sensor's signal
Output data: 1bit "1"
Sensor pull down bus's voltage
Pull up voltage and get ready for sensor's output
Single-bus output

2) Step 1: MCU send out start signal to DHT22

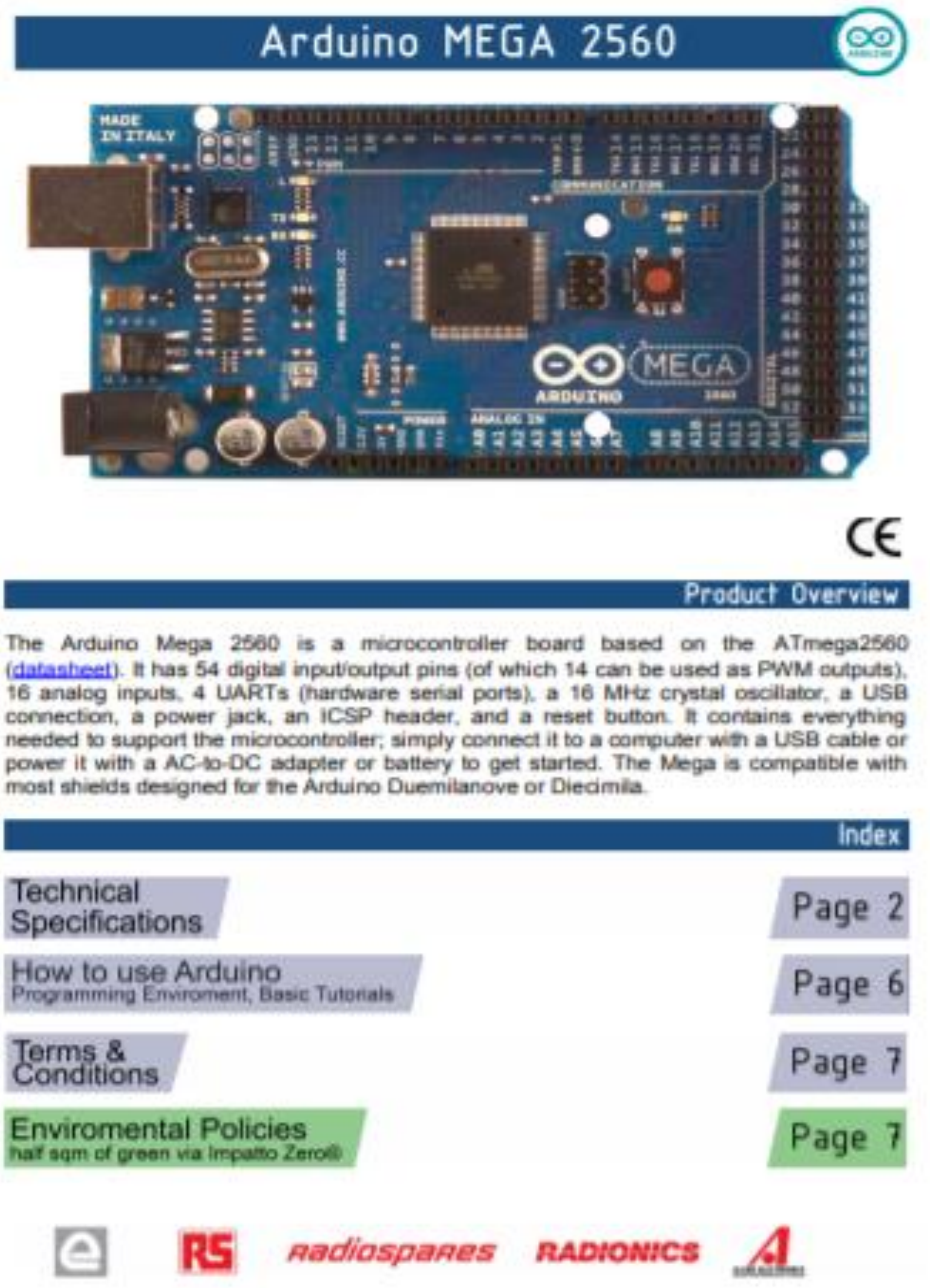
Data-bus's free status is high voltage level. When communication between MCU and DHT22 begin, program of MCU will transform data-bus's voltage level from high to low level and this process must beyond at least 1ms to ensure DHT22 could detect MCU's signal, then MCU will wait 20-40us for DHT22's response.

Check bellow picture for step 1:

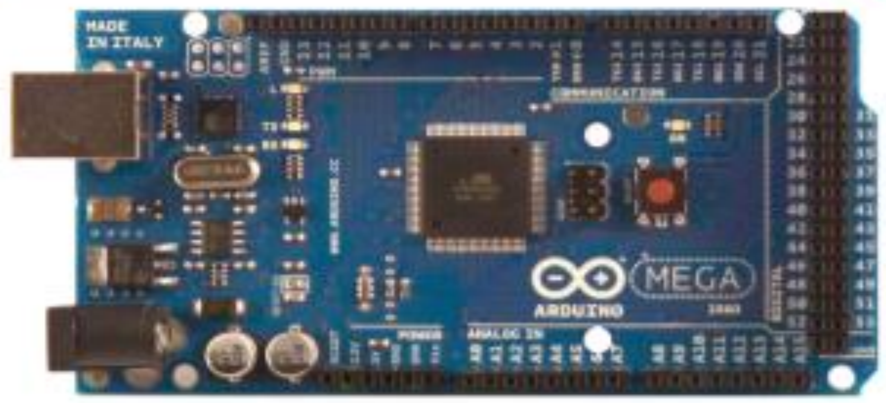
Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Arduino Mega



Arduino MEGA 2560



Product Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Index

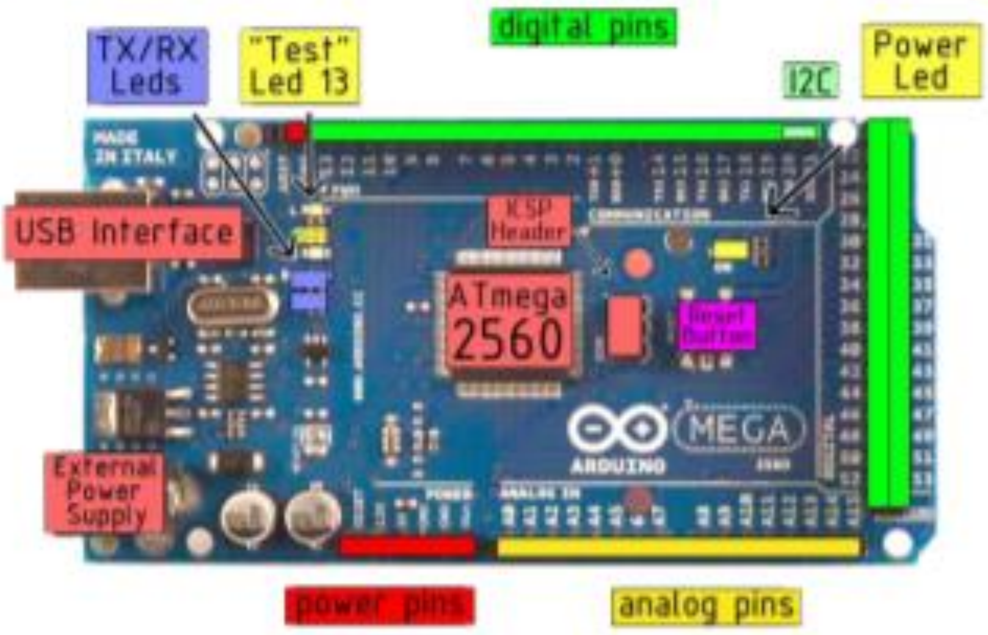
Technical Specification

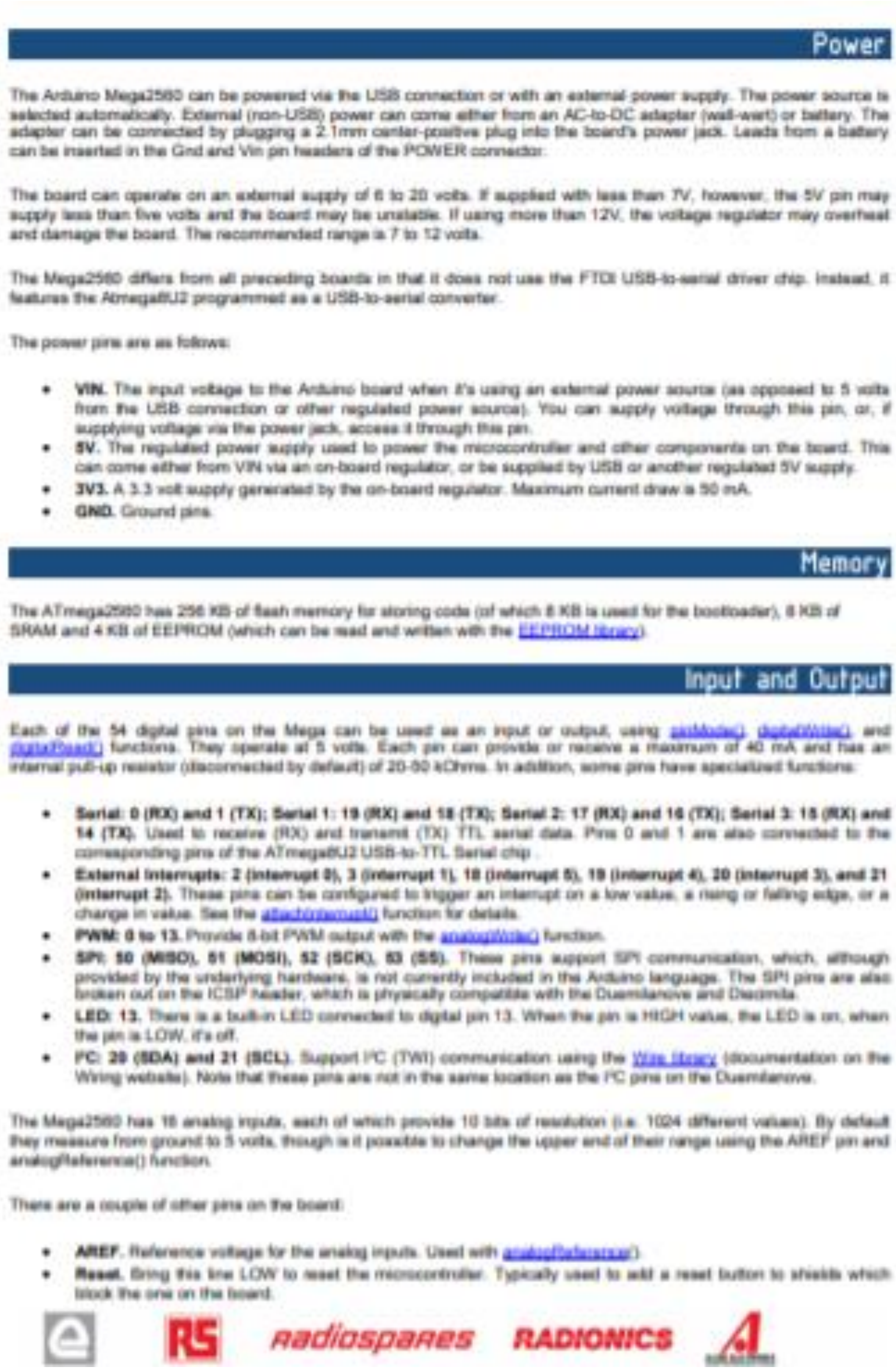
EAGLE files: arduino-mega2560-reference-design.zip Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

the board





Power

The Arduino Mega2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I²C: 20 (SDA) and 21 (SCL).** Support I²C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I²C pins on the Duemilanove.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

RS radiospares RADIONICS

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega's digital pins.

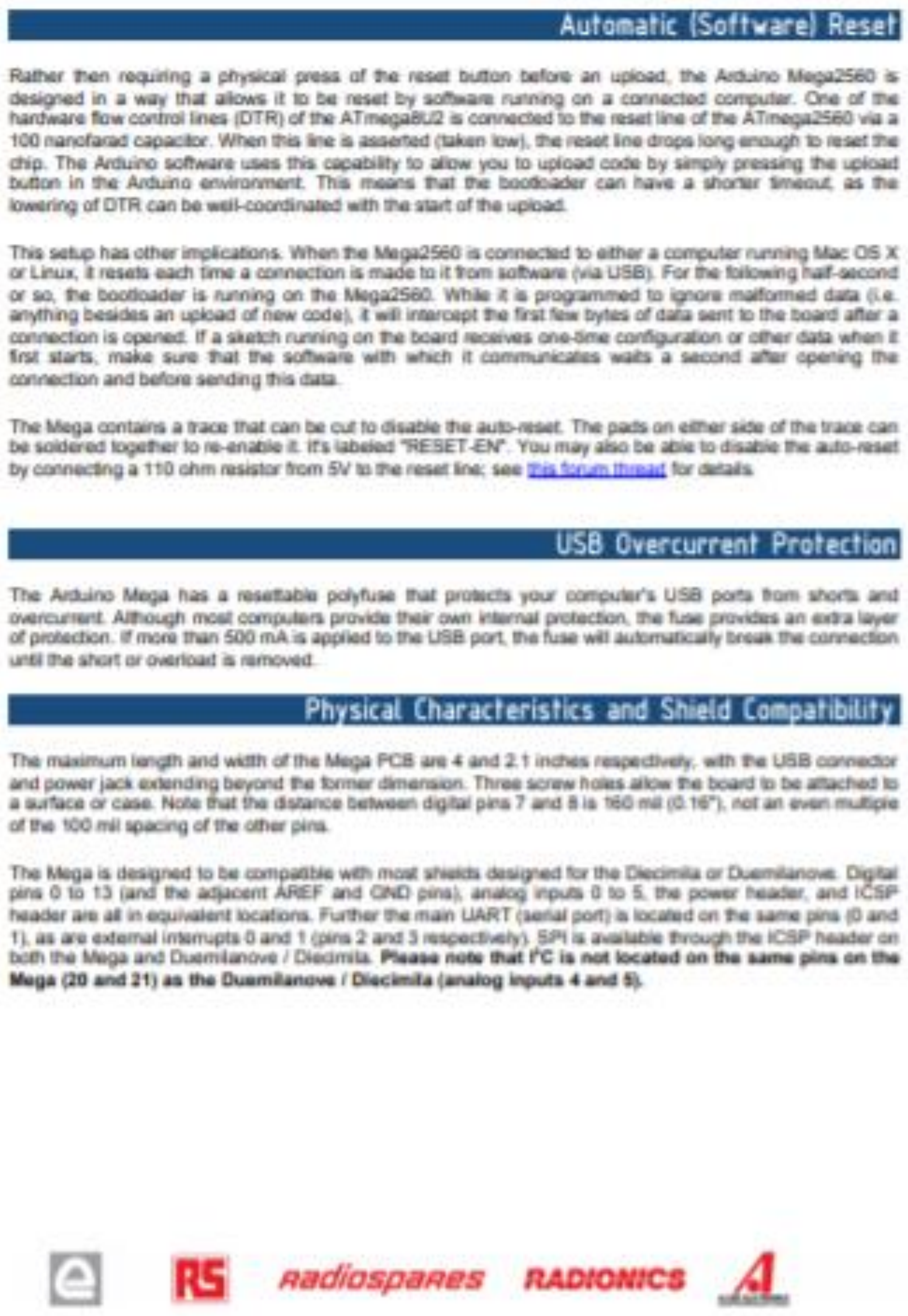
The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the ATmega2560 datasheet.

Programming

The Arduino Mega2560 can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Atmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.



Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila. **Please note that I^2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).**

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platoform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. http://arduino.cc/en/Guide/HomePage

Linux Install **Windows Install** **Mac Install**

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

Blink led

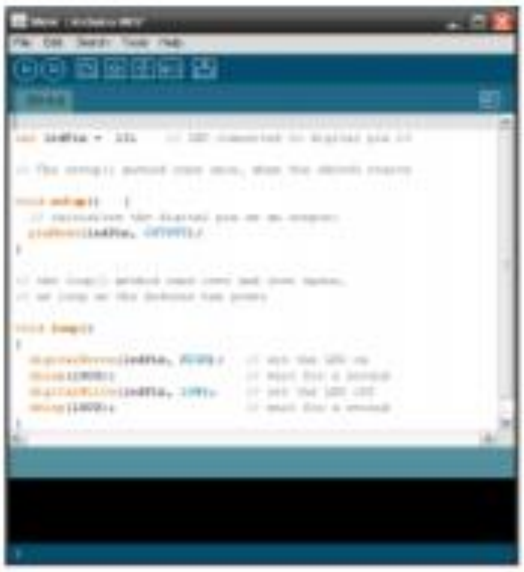
Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

File>Sketchbook>
Arduino-0017>Examples>
Digital>Blink

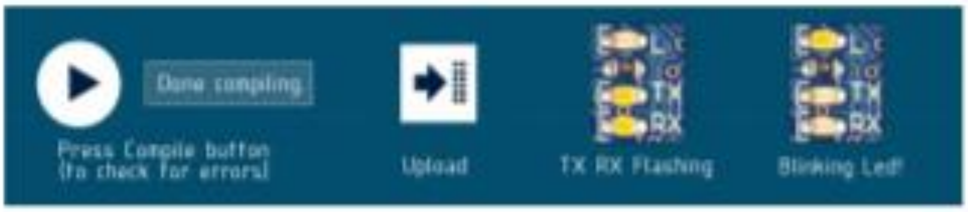
Once you have your skecth you'll see something very close to the screenshot on the right.

In **Tools>Board** select MEGA

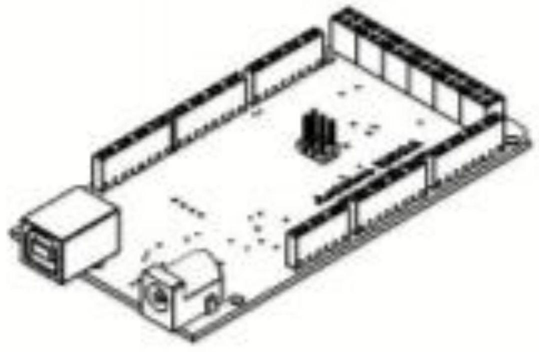
Now you have to go to
Tools>SerialPort
and select the right serial port, the one arduino is attached to.

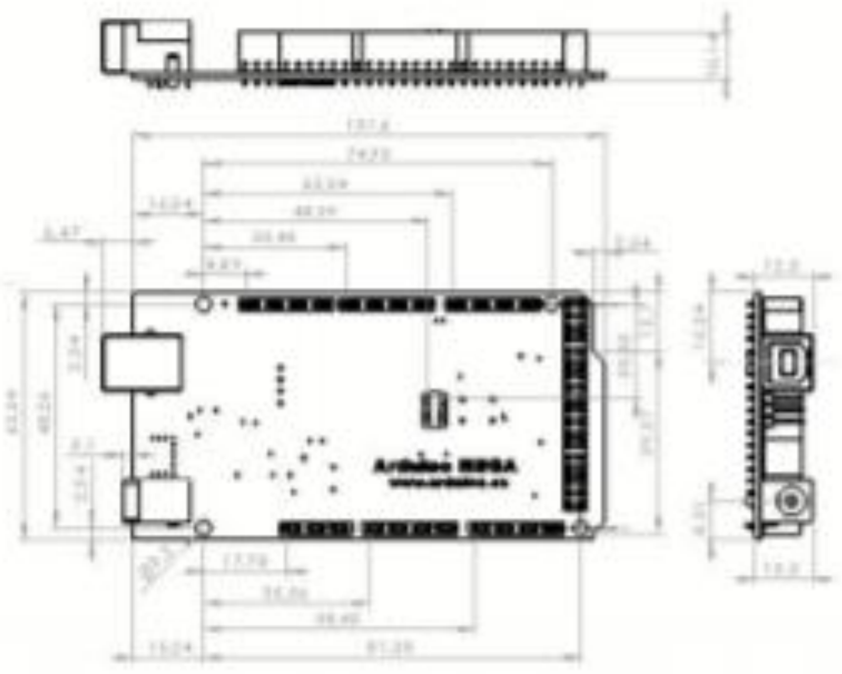


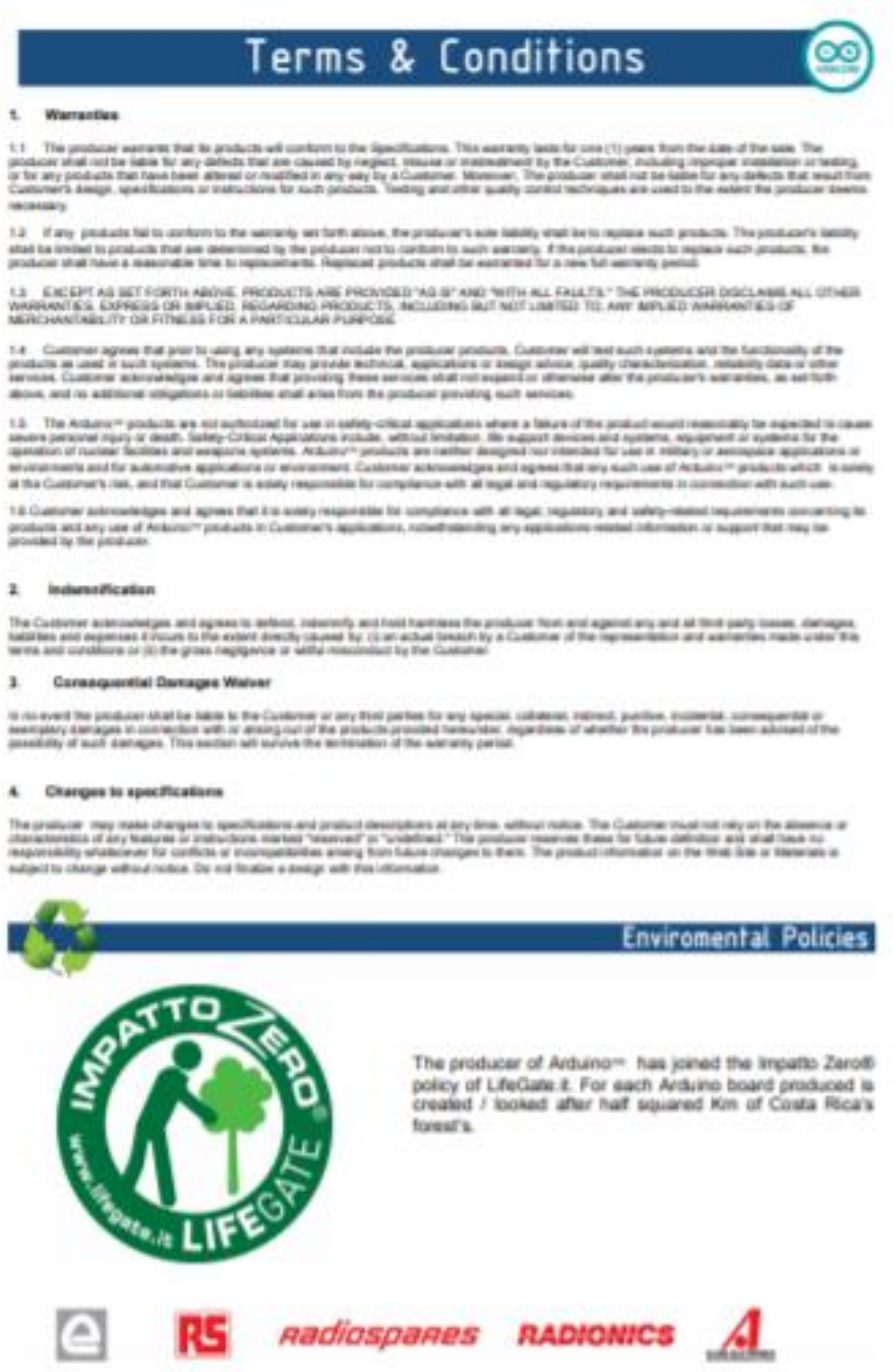
```
[illegible]
```



RS radiospares RADIONICS




Arduino MEGA
www.arduino.cc



Terms & Conditions

1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino™ products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino™ products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino™ products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino™ products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

4. Changes to specifications

The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

Enviromental Policies

The producer of Arduino™ has joined the Impatto Zero® policy of LifeGate.it. For each Arduino board produced is created / looked after half squared Km of Costa Rica's forest's.

RS radiospares RADIONICS