

Smart BBQ

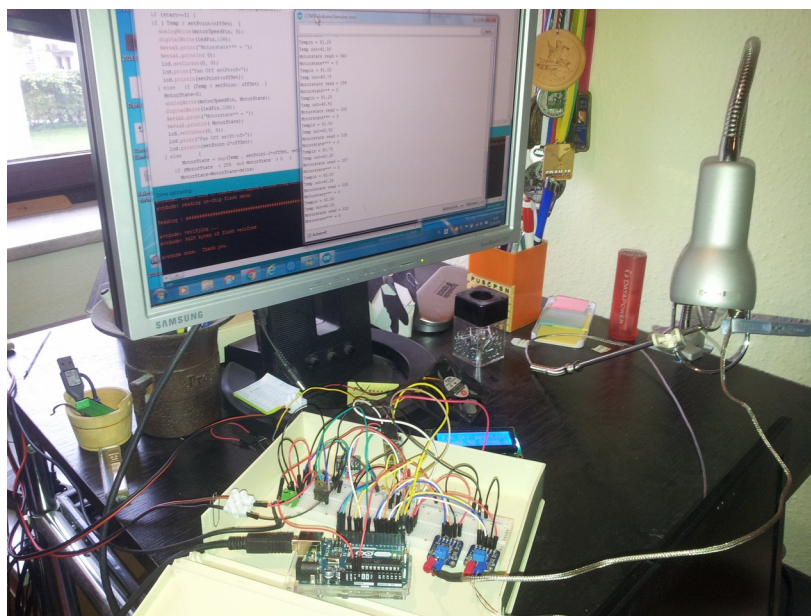
Android app controll Arduino Uno over BlueTooth and so fan output.



BBQ Project

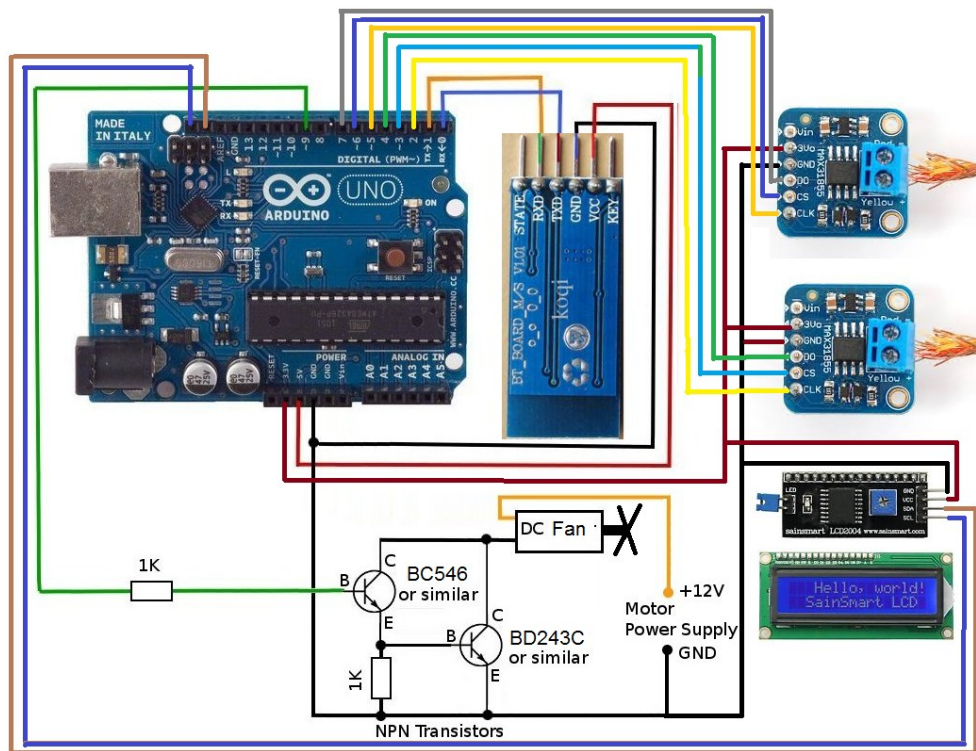
Smart BBQ consists of Arduino Uno R3, two termocouple senzors (termocouple probes), two MAX31855 chips, LCD display ,12 V DC fan, BD243C – NPN and BC546 – NPN transistors, two 1K resistors, three 0,250K resistors and LED diodes and 9-12 V power supply.

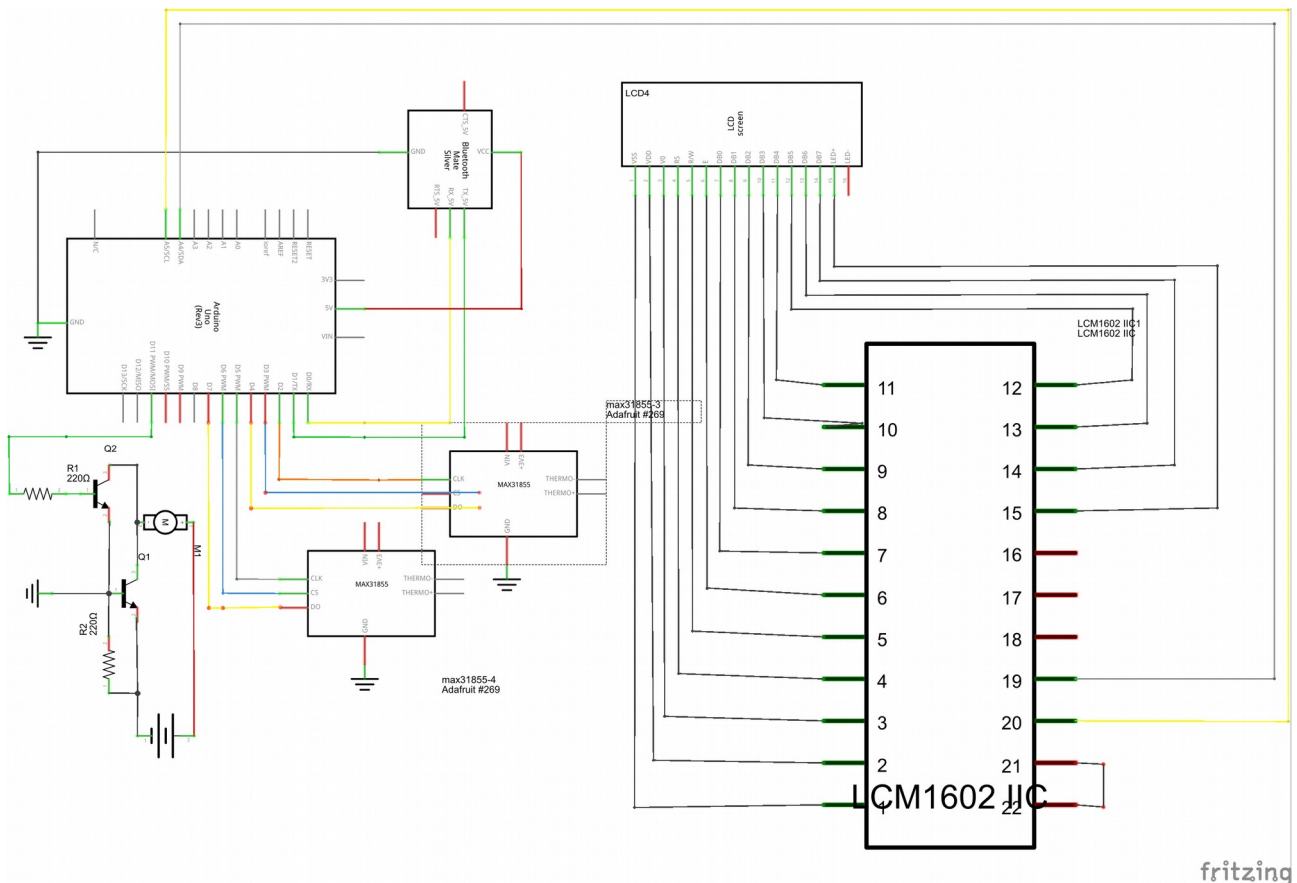
Develpments



I used Arduino on Windows, Adafruit library for MAX31855 and library for PWM control App Inventor ver.2 , Fritzing and soldering equipment.

Schematic





fritzing

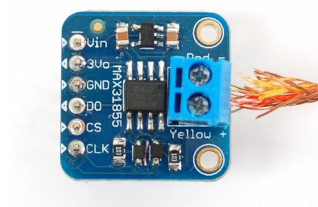
Using a Thermocouple

If you're planning to use the MAX6675/MAX31855, there's a little more work to be done. First off, Vin and GND must connect to a 3-5V supply. Then the three data pins must connect to digital IO pins:

- **CLK** (clock) is an input to the MAX6675/MAX31855 (output from microcontroller) which indicates when to present another bit of data
- **DO** (data out) is an output from the MAX6675/MAX31855 (input to the microcontroller) which carries each bit of data
- **CS** (chip select) is an input to the MAX6675/MAX31855 (output from the microcontroller) which tells the chip when its time to read the thermocouple and output more data.

In the beginning of our sketches, we define these pins. For our examples **DO** connects to digital 3, **CS** connects to digital 4, and **CLK** connects to pin 2

```
#define MAXDO  4
#define MAXCS  3
#define MAXCLK 2
//Tempout - long sonde
#define MAXDO1 5
```



```
#define MAXCS1 6
#define MAXCLK1 7
```

If you are using the MAX31855 v1.0 in a noisy environment, you may need to add a 0.01uF capacitor across the thermocouple leads.

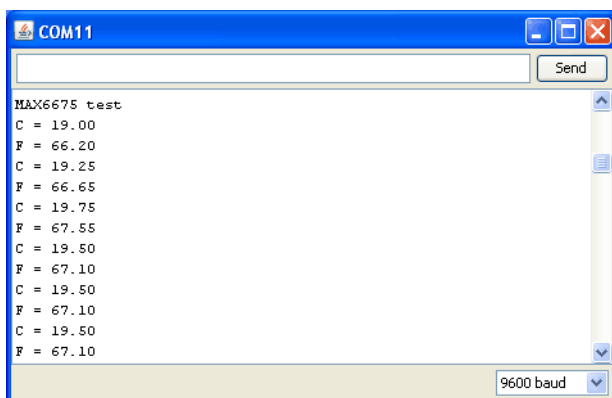
The MAX31855 does not support grounded thermocouples - if the sensor touches ground the chip will return an error

Arduino Library

If you have an older MAX6675 breakout, [download the MAX6675 Arduino library code](#) by going to the github page and clicking Download Source. Then uncompress the folder and rename it **MAX6675** and [install it into the library folder according to our handy tutorial](#).

If you have the newer MAX31855 breakout, [download the MAX31855 Arduino library code](#) by going to the github page and clicking Download Source. Then uncompress the folder and rename it **Adafruit_MAX31855** and [install it into the library folder according to our handy tutorial](#).

Restart the Arduino IDE and open up the **File->Examples->MAX6675/Adafruit_MAX31855->serialthermocouple** sketch and upload it to your Arduino. Once uploaded, open up the serial port monitor to display the current temperatures in both Celsius and Fahrenheit





As you can see, its pretty simple to use the library, simply tell the sensor object what the clock, chip select and data pins are, then call **readCelsius()** or **readFahrenheit()** to get a floating point result.

M K Type 800 Degree Celsius Thermocouple Sensor 5mm Probe Dia

Package List:

- 1 x Adafruit MAX31855 Module + K Type Thermocouple Thermocouple Sensor For Arduino

<http://www.sainsmart.com/sainsmart-max31855-module-k-type-thermocouple-sensor-module-for-arduino.html/>

 	Product Name Type Temperature Range Probe Diameter Probe Length Thread Diameter Internal Insulation External Shielding Total Length Fork Terminal Spacing Color Weight	Thermocouple K Type 800°C(32-752°F) 5mm / 0.2" 15cm / 5.9" (Not Included Flexible Section) 7.7mm / 0.3" Fibreglass Metal Shield 1M / 3.28ft 4mm / 0.16" Silver Tone 30g

Adding a Display

A common request is to have the temperature output onto [a 'classic' character LCD such as the ones in this tutorial](#).

For this wiring, we connected **CLK** to digital **3**, **CS** to digital **4** and **DO** to digital **5**. Once you get it working, you can change the pin connections in the sketch

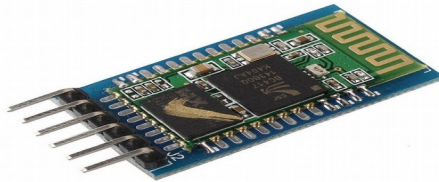
We have an example sketch for this as well. [First get the LCD working by following our tutorial](#). Now load up the new sketch **File->Examples->MAX31855>lcdthermocouple** and plug in the thermocouple module as we did in the serial thermocouple test, you'll see the internal temperature and the thermocouple temperature displayed in Celsius

Bluetooth connection

The HC-05 Bluetooth module is the most economical and easiest way to go wireless (via Bluetooth).

<http://randomnerdtutorials.com/arduino-control-dc-motor-via-bluetooth/>

HC-05 is a serial port module which makes it very easy to use. If you see the pin configuration of HC-05, there are total 6 but we only need 4 middle ones for our set-up.

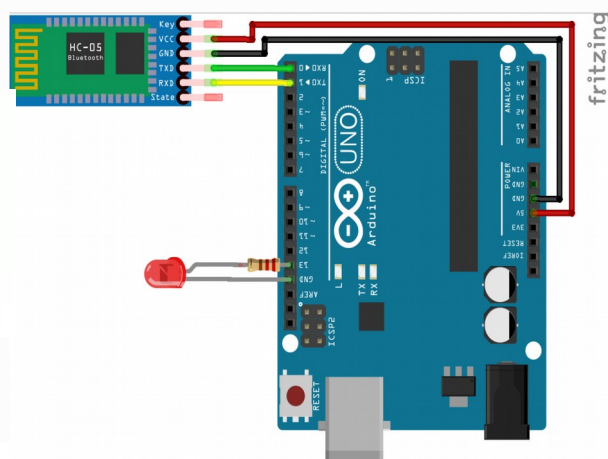
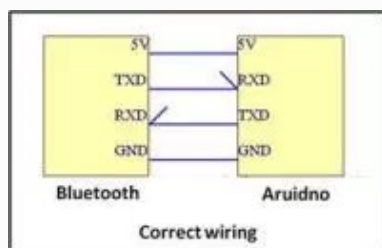


1. Connect VCC with 3.3V of Arduino, please do not connect it with 5V as that can cook the module
2. Connect GND with any GND of Arduino
3. Connect Rx pin with Tx of Arduino
4. Connect Tx pin with Rx of Arduino

Now power-up the Uno using USB cable, a red light LED on HC-05 will start blinking, means we are ready to go forward to the next step!

1pcs HC-05 Wireless Bluetooth RF Transceiver Module serial RS232 TTL for arduino

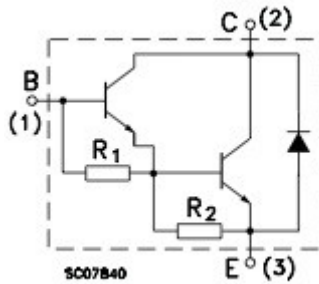
<http://www.ebay.com/itm/Wireless-Serial-4-Pin-Bluetooth-RF-Transceiver-Module-HC-06-RS232-With-backplane-/370999421767>



Controlling fan or motor speed with PWM

On the first part, I talk about switching any 12V DC or higher electronic components ON or OFF. This can be easily done using an optoisolator and a 12V reed relay. In this article, I will be using a different component to control the speed of the 12V fan or motors using Pulse Width Modulation (PWM).

The TIP-122 are Darlington transistors that can support voltage up to 100V. It consist of two transistors with resistors and diode all inside a TO-220 package. You can use TIP-120 that support up to 60V. Please refer to the TIP-120/TIP-122 datasheets for details specifications [here](#).



TIP-122

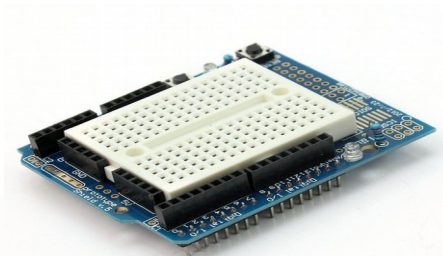
ARDUINO PROTO EXTENSION KIT, ARDUINO A000083

https://www.galagomarket.com/index.php/item/display/891/2470_shields_arduino_arduino-proto-extension-kit,-arduino-a000083Features & Benefits of the Proto Shield

- Connections for all Arduino I/O pins
- Space for through-hole and surface mount ICs
- Components can be soldered to the shield (a solderless breadboard is needed for non-soldered prototyping)

Proto Shield Kit Contents

- Prototype Size: 7cm x 5.5cm x 2cm
- Mini breadboard size: 4.4 x 3.4 x 1cm



Connection to IOT

There are many more way to connect uno to internet to send , receive data and do some interesting IOT project.

1.GPRS modem(Old and for remote area use)

2.ESP8266 wifi chip (Cheap and Easy To use)

<http://dalpix.com/blog/connecting-your-arduino-wifi-esp-8266-module>

3.enc28j60 Ethernet Module

4. Bluetooth to smart phone, and smart phone to internet.

[http://androidexample.com/How_To_Make_HTTP_POST_Request_To_Server_-](http://androidexample.com/How_To_Make_HTTP_POST_Request_To_Server_-_Android_Example/index.php?view=article_discription&aid=64)

[_Android_Example/index.php?view=article_discription&aid=64](http://androidexample.com/How_To_Make_HTTP_POST_Request_To_Server_-_Android_Example/index.php?view=article_discription&aid=64)

<https://trinitytuts.com/post-json-data-to-server-in-android/>

<https://www.quora.com/What-are-some-iot-based-projects-using-arduino-uno>

Arduino Code

```
//Author:Boris Milikic
/*Resources
  http://www.electroschematics.com/9540/arduino-fan-speed-controlled-temperature/
  source: http://www.electroschematics.com/9540/arduino-fan-speed-controlled-temperature/

  * http://www.circuitstoday.com/pwm-generation-and-control-using-arduino
  * https://www.arduino.cc/en/Tutorial/SecretsOfArduinoPWM
  *
BBQ temperature range:
  * HIGH
2 to 3 seconds; 450 to 650Â°F ,232 - 343Â°C
Right after the coals are dumped from the chimney, the heat is
extremeâ€”too hot to cook almost anything. You may be ranging more around
the 1-second mark here, which is why this is the time to let the coals
heat up that dirty cooking grate for about 5 minutes, then scrub it off
and test again. Now you should be more in the area of 2 to 3 seconds, and
also playing with a temperature that won't be sustained long. If there's
a tuna steak or thin pork chop that's going benefit from a quick and hot
cook, now's the time to get that done.
  * MEDIUM-HIGH 4 to 5 seconds; 375 to 450 Â°F ,190 -232Â°C
The second the charcoal is lit and spread out, it starts losing heat. It
will only take about 5 to 10 minutes to go from high heat to medium-high,
and a lot of your direct grilling will most likely be done in this
range.Â This is the temperature that makes things like burgers, sliced
veggies, and fish really shine.
  * MEDIUM 6 to 7 seconds; 325 to 375Â°F, 162- 190 Â°C
  * Totally middle of the road, offering everything the name impliesâ€”a
fire low enough to gently cook, while still hot enough to brown outsides
nicely during longer cooking times. This is the king of the indirect
cook: chickens, turkeys, roasts. They all love a soothing medium heat.
  * MEDIUM-LOW 8 to 10 seconds; 250Â°F to 325Â°F, 121-162- Â°C
Once you drop to around the 300Â°F mark, you're dealing with a fire that
isn't going to brown anything. If you're working with direct grilling,
it's time to replenish those coals and get the grill back up to
temperature.
  *LOW 11 to 15 seconds; 225 to 250Â°F, 107-162- Â°C no longer grilling,
but barbecuing (as long as you add smoke into the equation). At this
temperature the fat and connective tissue in ribs, pork shoulders, and
briskets slowly, but surely melts away, transforming the toughest cuts
into moist and tender delights that gives me purpose
  *
```



```

*   roka približno 5 do 10 minut nad kuhalo rešetko
*   It will only take about 5 to 10 minutes to go from high heat to
medium-high,
*   MEDIUM-HIGH 4 to 5 seconds; 375 to 450 °F, 190 -232°C:
temperatura primerna za hamburgerje, narezano zelenjavo in ribe
*   MEDIUM 6 to 7 seconds; 325 to 375°F, 162- 190 °C : *This is the king
of the indirect cook: chickens, turkeys, roasts. They all love a soothing
medium heat. A well-seasoned grill with the lid on can easily keep in the
medium range for 45 to 60 minutes, which is perfect for those longer
cooking items.
*   MEDIUM-LOW 8 to 10 seconds; 250°F to 325°F, 121-162 °C: you're
dealing with a fire that isn't going to brown anything. If you're working
with direct grilling, it's time to replenish those coals and get the
grill back up to temperature.
*   LOW 11 to 15 seconds; 225 to 250°F, 107-121 °C:you're no longer
grilling, but barbecuing (as long as you add smoke into the equation). At
this temperature the fat and connective tissue in ribs, pork shoulders,
and briskets slowly, but surely melts away, transforming the toughest
cuts into moist and tender delights that gives me purpose. You can
transform a kettle grill into a smoker
*/

// Serial Parameters: COM1 9600 8 N 1
// \r or \n to end command line
// Bluetooth is on Pin 0 & 1 @ 9600 speed

// Command structure
// CMD MODE=AUTO|MANUAL|STOP
// CMD TMAX|TMIN|SECONDS|SPEED=value
// CMD SECONDS=value
// CMD STATUS

// Status message structure
// STATUS AUTO|MANUAL|FAN%|TMIN|TMAX|SECONDS|TEMPIN|TEMPZAR|FANSPEED|
THIGH=value

//#include <SPI.h>
//#include <Wire.h>
#include "Adafruit_MAX31855.h"
#include <LiquidCrystal_I2C.h>

#include <math.h>

// Example creating a thermocouple instance with software SPI on any
three
// digital IO pins.
//tempin long sonde
#define MAXDO 4
#define MAXCS 3
#define MAXCLK 2
//Tempzar - short sonde
#define MAXDO1 5
#define MAXCS1 6
#define MAXCLK1 7

volatile int seconds = 0;

volatile boolean statusReport = false;

```

```

String inputString = "";
String command = "";
String value = "";
boolean stringComplete = false;

int maxSeconds = 11; // send status message every maxSeconds

//https://www.arduino.cc/en/Reference/AnalogWrite
//https://www.arduino.cc/en/Tutorial/SecretsOfArduinoPWM
//analogWrite(pin, dutyCycle), where dutyCycle is a value from 0 to 255,
and pin is one of the PWM pins (3, 5, 6, 9, 10, or 11).

int motorFanPin = 11;
// fast PWM on pins 3 and 11 (Timer 2). In the simplest PWM mode, the
timer repeatedly counts from 0 to 255.
//The output turns on when the timer is at 0, and turns off when the
timer matches the output compare register.
//The higher the value in the output compare register, the higher the
duty cycle. This mode is known as Fast PWM Mode.

// Initialize the Thermocouple
Adafruit_MAX31855 thermocouple(MAXCLK, MAXCS, MAXDO);
Adafruit_MAX31855 thermocouple1(MAXCLK1, MAXCS1, MAXDO1);

// initialize the library with the numbers of the interface pins
LiquidCrystal_I2C lcd (0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); // Set
the LCD I2C address

#ifdef ARDUINO_ARCH_SAMD
// for Zero, output on USB Serial console, remove line below if using
programming port to program the Zero!
#define Serial SerialUSB
#endif

int temp;
int minTemp= 60; // the temperature to start the fan
int maxTemp= 100; // the maximum temperature when fan is at 100%

int setPoint = 71;
int offSet = 5;
boolean state = true;
String mode = "MANUAL";
int fanSpeed = 0; //0-255
int fanLCD=0;
int ledPinauto = 8;
int ledPinmanual = 9;
int ledPinstop = 12;
int Tempin;
int Tempzar;
int delta=5;
boolean loopOK=false;

void setup() {

```

```

#ifdef ESP8266
    while (!Serial);    // will pause Zero, Leonardo, etc until serial
    console opens
#endif

inputString.reserve(50);
command.reserve(50);
value.reserve(50);
pinMode(ledPinstop, OUTPUT);
pinMode(ledPinauto, OUTPUT);
pinMode(ledPinmanual, OUTPUT);

    digitalWrite(ledPinstop, LOW );
    digitalWrite(ledPinauto, LOW);
    digitalWrite(ledPinmanual, LOW);

pinMode(motorFanPin, OUTPUT);
analogWrite(motorFanPin, 0);

/*
the following code is needed to initialize the timer interrupt and set it
to fire every second, the slowest that Arduino can do
for detailed information see: http://www.instructables.com/id/Arduino-Timer-Interrupts/step1/Prescalers-and-the-Compare-Match-Register/
*/

cli();          // disable global interrupts

// initialize Timer1 for interrupt @ 1000 msec
TCCR1A = 0;      // set entire TCCR1A register to 0
TCCR1B = 0;      // same for TCCR1B

// set compare match register to desired timer count:
OCR1A = 15624;
// turn on CTC mode:
TCCR1B |= (1 << WGM12);
// Set CS10 and CS12 bits for 1024 prescaler:
TCCR1B |= (1 << CS10);
TCCR1B |= (1 << CS12);
// enable timer compare interrupt:
TIMSK1 |= (1 << OCIE1A);

sei();          // enable global interrupts

// set up the LCD's number of columns and rows:
lcd.begin(16, 2);
lcd.clear();
lcd.print("Arduino BBQ");
Serial.begin(9600);
Serial.print("Max T: ");
Serial.print(maxTemp);
Serial.print("Min T: ");
Serial.print(minTemp);
Serial.print(" setPoint: ");
Serial.println(setPoint);

```

```

    // wait for MAX chip to stabilize
    delay(1000);
}

// timer interrupt routine
ISR(TIMER1_COMPA_vect)
{
    digitalWrite(ledPinstop, LOW );
    digitalWrite(ledPinauto, LOW);
    digitalWrite(ledPinmanual, LOW);
    Tempin = (int)thermocouple.readCelsius();
    if (isnan(Tempin))
    {
        Serial.print("T/C Problem");
    }

    Tempzar = (int) thermocouple1.readCelsius();
    if (isnan(Tempzar))
    {
        Serial.print("T/C Problem");
    }

    if (seconds++ >= maxSeconds) {
        statusReport = true;
        seconds = 0;
    }
}

void loop() {

    int intValue = 0;
    if (stringComplete) {
        state=true;
        boolean stringOK = false;
        loopOK=true;
        if (inputString.startsWith("CMD")) {
            inputString = inputString.substring(4);
            int pos = inputString.indexOf('=');
            if (pos > -1) {
                command = inputString.substring(0, pos);
                value = inputString.substring(pos+1, inputString.length()-1);//
extract command up to \n exluded
                value.toUpperCase();
                value.trim();
                Serial.print(command);
                Serial.print(":");
                Serial.println(value);
                if (command.equals("MODE")) { // AUTO|MANUAL|STOP
                    loopOK=true;
                    mode=value;
                    if (mode=="MANUAL") {
                        fanSpeed=0;
                        //      Serial.println(value);
                        Serial.println("SELECT SPEED");
                    }
                    stringOK = true;

```

```

    } else if (command.equals("SETPT")) { // TMAX=value
        intValue = value.toInt();
        if (intValue > 0) {
            setPoint = intValue;
            if (setPoint > 90 or setPoint < 60) {
                setPoint = 80;
            }
            stringOK = true;
        }
    } else if (command.equals("SPEED")) { // TMAX=value
        intValue = value.toInt();
        if (intValue > 0) {
            fanSpeed = intValue;
            stringOK = true;
        }
    }

    } else if (command.equals("TMAX")) { // TMAX=value
        intValue = value.toInt();
        if (intValue > 0) {
            maxTemp = (float) intValue;
            stringOK = true;
        }
        loopOK=false;
    } else if (command.equals("TMIN")) { // TMAX=value
        intValue = value.toInt();
        if (intValue > 0) {
            minTemp = (float) intValue;
            stringOK = true;
        }
        loopOK=false;
    } else if (command.equals("SECONDS")) { // SECONDS=value
        intValue = value.toInt();
        if (intValue > 0) {
            maxSeconds = intValue;
            stringOK = true;
        }
        loopOK=false;
    }
} // pos > -1
else if (inputString.startsWith("STATUS")) {
    Serial.print("STATUS MODE=");
    Serial.println(mode);
    Serial.print("STATUS setPoint=");
    Serial.println(setPoint);
    Serial.print("STATUS fanSpeed=");
    Serial.println(fanSpeed);
    Serial.print("STATUS TMAX=");
    Serial.println(maxTemp);
    Serial.print("STATUS TMIN=");
    Serial.println(minTemp);
    Serial.print("STATUS SECONDS=");
    Serial.println(maxSeconds);
    Serial.print("STATUS TEMPIN=");
    Serial.println(Tempin);
    Serial.print("STATUS TEMPZAR=");
    Serial.println(Tempzar);

    stringOK = true;
}

```

```

        } // inputString.startsWith("STATUS")
    } // inputString.startsWith("CMD ")
    stringOK ? Serial.println("Command Executed") :
Serial.println("Invalid Command");
    state=true;
    // clear the string for next iteration
    inputString = "";
    stringComplete = false;
} // stringComplete

if (statusReport) {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Tin:");
    lcd.print(Tempin);          // display the temperature
    lcd.print(" ");
    lcd.print("Tz:");
    lcd.print(Tempzar);        // display the temperature
    lcd.print(" C");
    lcd.setCursor(0,1);    // move cursor to next line
    lcd.print("FAN:");
    lcd.print(fanLCD);      // display the fan speed
    lcd.print("%");
    lcd.print(" Set:");
    lcd.print(setPoint);    // display the fan speed
    lcd.print("C");

    Serial.print("STATUS Tempzar=");
    Serial.println(Tempzar);
    Serial.print("STATUS Tempin=");
    Serial.println( Tempin);
    Serial.print("STATUS setPoint=");
    Serial.println(setPoint);
    Serial.print("STATUS fanSpeed=");
    Serial.println(fanSpeed);
    Serial.print("STATUS mode=");
    Serial.println(mode);
    Serial.print(":");
    Serial.print(fanLCD);
    Serial.print(":");
    Serial.print( Tempin);
    Serial.println(":");

    statusReport = false;
}

if (mode=="AUTO" and loopOK) {
    fanSpeed =0;
    if((Tempzar < minTemp) || (Tempzar > maxTemp)){    // if temp is
lower than minimum temp
        fanSpeed=0;          // fan is not spinning
        digitalWrite(ledPinauto, LOW);
        delay(100);
    }

    if((Tempzar == 0 or isnan(Tempzar)) && (Tempin < setPoint)) {    //
if temperature is higher than minimum temp
        fanSpeed = map(Tempin, minTemp, maxTemp, 255,32); // the actual
speed of fan 50, 130 ->0,255

```



```

        digitalWrite(ledPinstop, LOW );
        digitalWrite(ledPinauto, HIGH);
        digitalWrite(ledPinmanual, LOW);
        //fanSpeed=fanSpeed+delta*(setPoint-Tempin);
        if (fanSpeed > 255 )
            fanSpeed=255;
    }

    if((Tempzar >= minTemp) && (Tempzar <= maxTemp) && (Tempin <
setPoint)) { // if temperature is higher than minimum temp
        fanSpeed = map(Tempin, minTemp, maxTemp, 255,32); // the actual
speed of fan 50, 130 ->0,255
        digitalWrite(ledPinstop, LOW );
        digitalWrite(ledPinauto, HIGH);
        digitalWrite(ledPinmanual, LOW);
        //fanSpeed=fanSpeed+delta*(setPoint-Tempin);
        if (fanSpeed > 255 )
            fanSpeed=255;
        }
        fanLCD = (fanSpeed*100/255); //map(fanSpeed, minTemp, maxTemp, 0,
100); // speed of fan to display on LCD
    } else if (mode == "MANUAL" and fanSpeed>0 and loopOK) {
        loopOK=false;
        //fanSpeed = map(fanSpeed, minTemp,maxTemp,0, 255); // the
actual speed of fan
        fanLCD = (fanSpeed*100/255); //map(fanSpeed, minTemp,maxTemp, 0,
100); // speed of fan to display on LCD
        if (fanSpeed > 255 ){
            fanSpeed=255; // fanSpeed-delta;
            fanLCD =100;
        }

        if (fanSpeed < 0 ) {
            fanSpeed=0;
            fanLCD =0;
        }
        digitalWrite(ledPinstop, LOW );
        digitalWrite(ledPinauto, LOW);
        digitalWrite(ledPinmanual, HIGH);
    } else if(mode == "STOP") {
        fanSpeed=0; // fan is not spinning
        // analogWrite(motorFanPin, fanSpeed);
        digitalWrite(ledPinstop, HIGH);
        digitalWrite(ledPinauto, LOW);
        digitalWrite(ledPinmanual, LOW);
        fanLCD=0;
    } //end if mode

    if (state) {
        // Serial.print("fanSpeed:");
        // Serial.println(fanSpeed);
        analogWrite(motorFanPin, fanSpeed); // spin the fan at the
fanSpeed speed
        state=false;
        // Serial.print("fanSpeed2:");
        // Serial.println(analogRead(motorFanPin));
    }

    delay(1000);

```

```
}

/*
  SerialEvent occurs whenever a new data comes in the
  hardware serial RX.  This routine is run between each
  time loop() runs, so using delay inside loop can delay
  response.  Multiple bytes of data may be available.
  */
void serialEvent() {
  while (Serial.available()) {
    // get the new byte:
    char inChar = (char)Serial.read();
    //Serial.write(inChar);
    // add it to the inputString:
    inputString += inChar;
    // if the incoming character is a newline or a carriage return, set a
flag    // so the main loop can do something about it:
    if (inChar == '\n' || inChar == '\r') {
      stringComplete = true;
    }
  }
  Serial.println(inputString);
}
```