Bernburg
Dessau
Köthen

**Hochschule Anhalt**
Anhalt University of Applied Sciences

emw
Fachbereich
Elektrotechnik, Maschinenbau
und Wirtschaftsingenieurwesen

# Projektarbeit

**Jian Song**

Vorname Nachname

Master Elektro- und Informationstechnik,
4055229

Studiengang, Matrikelnummer

Thema:

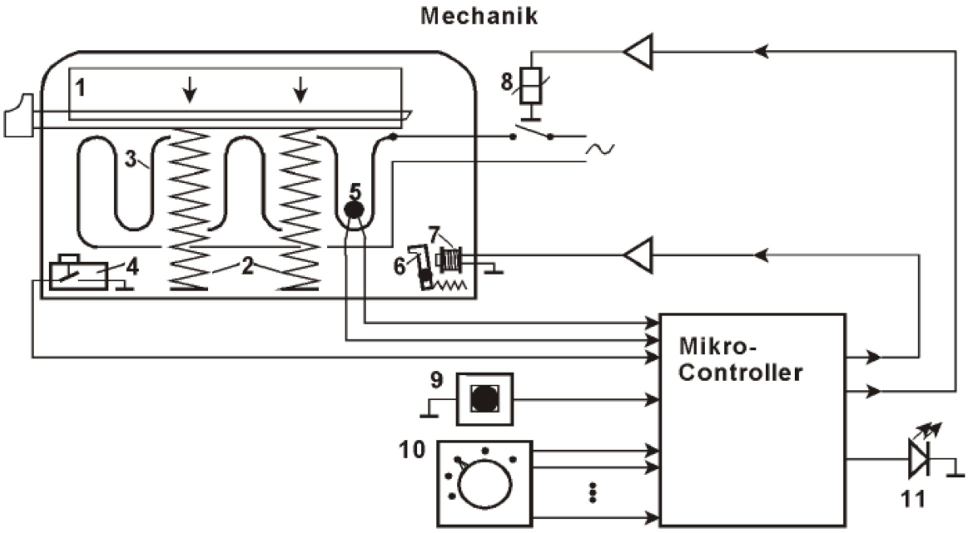
## Aufbau eines Praktikumsversuches “Toastersteuerung” mit einem Arduino-Board

Prof. Dr. Ingo Chmielewski

Betreuer

# Inhaltsverzeichnis

## 1. Schema der Funktionsweise eines Toasters



### Die zu steuernde Hardware

1 – Korb 2 – Druckfeder 3 - Heizwendel
4 – Endlagenkontakt 5 – Temperatursensor 6 – Klinke
7 – Auslösemagnet 8 -Schaltrelais (oder Triac) 9 – Stoptaste
10 -Drehschalter zum Einstellen des Bräunungsgrades 11 – Kontrollanzeige (Leuchtdiode).

In dem Fall gibt er kein Endlagenkontakt 4 sondern zwei metallische Blatt. Wenn man Korb druckt, kontaktieren die beiden einander. Mechanische Relay 8 wurde durch ein Solid State Relay ersetzt, damit Relay PWM-Signal erkennen kann. OneWire-Temperatursensor wurde in dem Fall als Rückführung hinzugefügt, damit eine Überhitzung zu Unterbrechung des Systems führt. Arduino-Board wurde mit ein AC/DC-Wandler verbunden und AC/DC-Wandler wurde mit Netzkabel des Toasters verbunden, damit 230V-AC direkt Arduino-Board versorgen kann.

***Bevor Programmierung in Arduino-IDE soll man folgende Bibliotheken installieren:***

***PWM frequency library:***

http://code.google.com/p/arduino-pwm-frequency-library/downloads/list

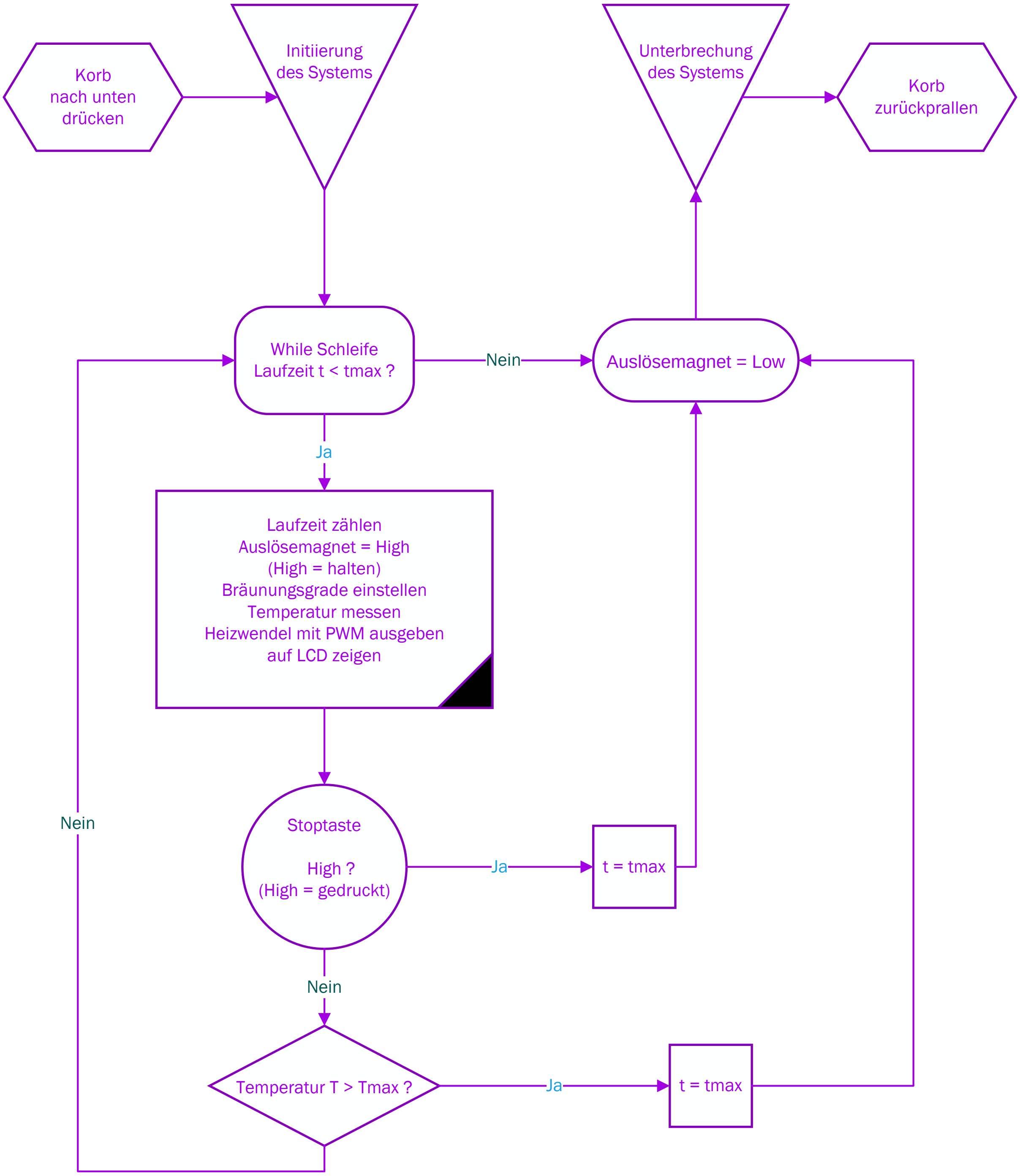
***OneWire library:***

https://github.com/PaulStoffregen/OneWire

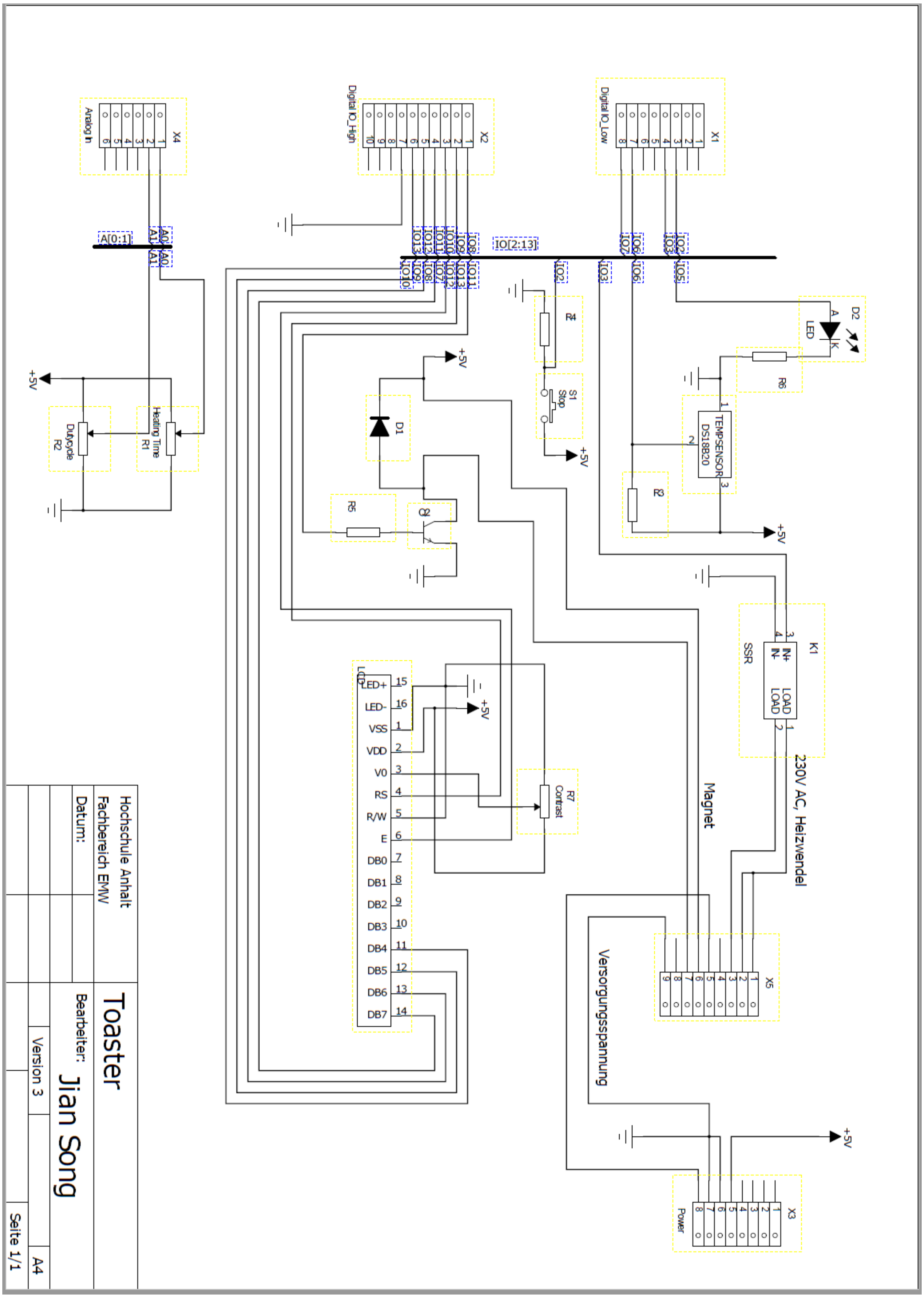
***DallasTemperature library:***

https://github.com/milesburton/Arduino-Temperature-Control-Library

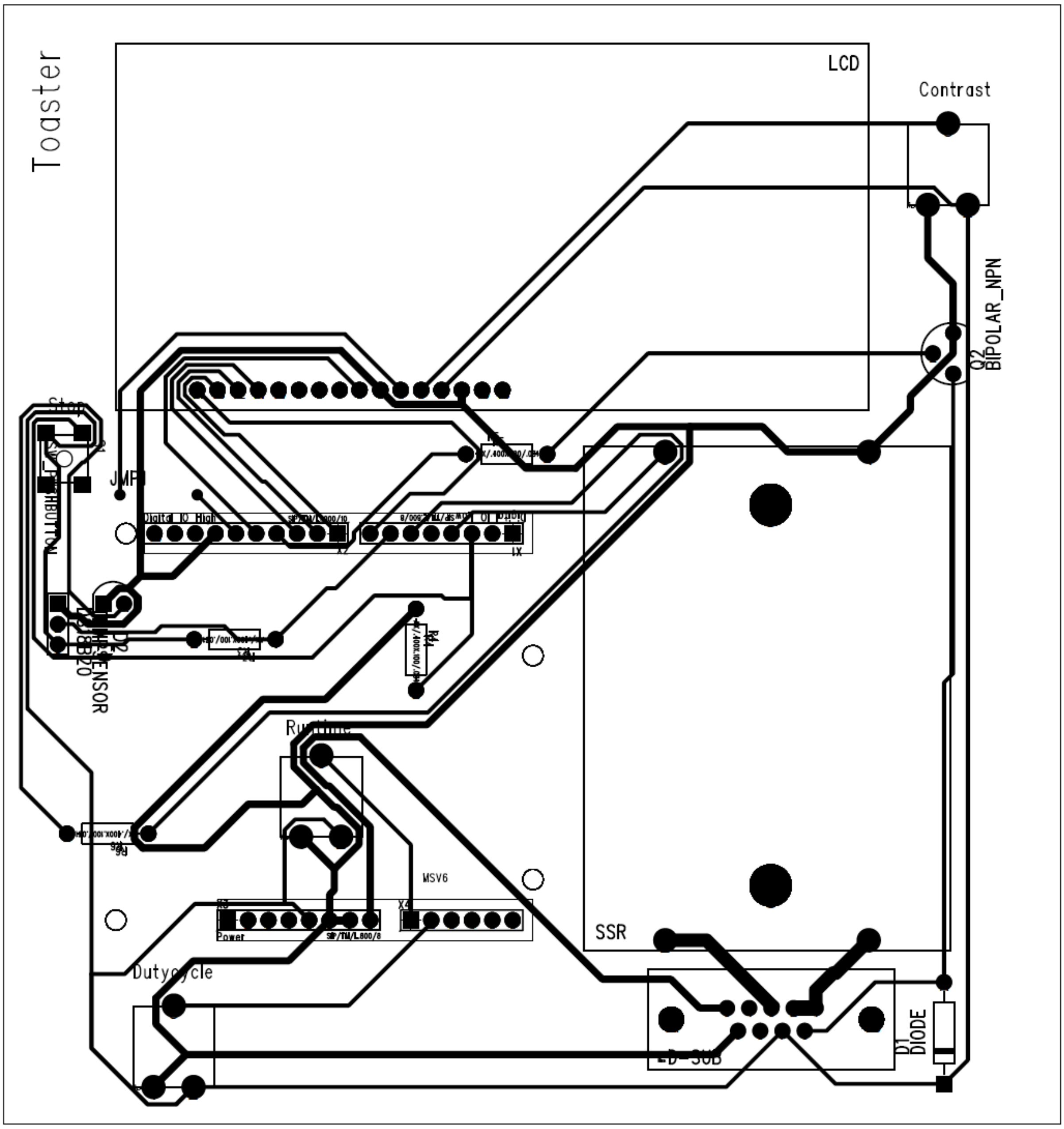
## 2. Ablauf-Plan des Programms



## 3. Strom Lauf-Plan



## 4. PCB Layout


Toaster
LCD
Contrast
BIPOLAR_NPN
Q2
Stop
PUSHBUTTON
JMP1
Digital IO High
X2
X1
R1
R3
R4
D2
DS18B20
TEMPSENSOR
Running
MSV6
X3
X4
Power
SP/TM/L800/8
R6
Dutycycle
SSR
SD-SUB
D1
DIODE

## 5. Arduino-Code mit getrennter Funktionen

```
/********************
Toaster Project
Rev 3
Jian Song

Functions:
1. Drehschalter
2. Auslösemagnet
3. Temperatursensor
4. Heizwendel
5. Stoptaste
6. LCD
7. Kontrollanzeige
*********************/

void Drehschalter() {
//function for calibration of the heating time

      int Grades;
//define the degrees of browning
      unsigned long tmax = 60000;
//define the max. heating time
      Grades = analogRead(0);
//read the data from pin A0 of analog input
      if (Grades >= 0 && Grades <= 205) {
//input range for A0 is from 0 to 1024(because of 10 bit ADC),
define potentiometer as 5 steps
        tmax = 10000;
            }
      if (Grades > 205 && Grades <= 410) {
        tmax = 20000;
            }
      if (Grades > 410 && Grades <= 615) {
        tmax = 30000;
            }
      if (Grades > 615 && Grades <= 820) {
        tmax = 40000;
            }
      if (Grades > 820 && Grades <= 1024) {
        tmax = 50000;
            }

}

void Ausloesemagnet() {
//function for magnet trigger

      int Magnet = 11;
//define the magnet
      unsigned long t = 0;
//define the runtime
      pinMode(Magnet, OUTPUT);
//set pin 11 as digital output
      while (t < tmax) {
//run the loop if the runtime is less then tmax
        t = millis();
//returns the number of milliseconds since the Arduino board began
running the current program
        digitalWrite(Magnet, HIGH);
//turn on the magnet
        }
        digitalWrite(Magnet, LOW);
//turn off the magnet, if the loop is broken

}

void Temperatursensor() {
//function for reading temperature in celsius

      #include <OneWire.h>
//use onewire library
      #include <DallasTemperature.h>
//use dallastemperature library
```

```
      int tempsensor = 6;
//define temperature sensor
      double temperature;
//define the temperature
      double Tmax = 50;
//define the max. temperature
      OneWire onewirepin(tempsensor);
//setup a onewire instance to communicate with any onewire devices
      DallasTemperature sensors(&onewirepin);
//pass our oneWire reference to dallas temperature
      sensors.begin();
//start the library up
      sensors.requestTemperatures();
//send the command to get temperatures
      temperature = sensors.getTempCByIndex(0);
//get the temperature from the first sensor only
      if (temperature > Tmax) {
//compare the current temperature and max. temperature. if higher
than max. , set t as tmax, then that while loop condition in
function of ausloesemagnet is not true anymore, so the while loop
will be broken until reset.
        t = tmax;
      }

}

void Heizwendel() {
//function for controlling the heater

      #include <PWM.h>
//use PWM library

      #define PIN_OUTPUT 3
//define pin 3 as output of PWM
      int D;
//set the dutycycle for PWM(8 bit timer can provide a range from 0
to 255 steps for PWM)
```

```
      int32_t frequency = 10;
//set the frequency of PWM in hertz
      InitTimersSafe();
//initialize all timers except for 0, to save time keeping
functions(for UNO timer0 and timer2 are 8 bit, timer1 is 16 bit;
pins 5 and 6: controlled by timer0; pins 9 and 10: controlled by
timer1; pins 11 and 3: controlled by timer2)
      bool success = SetPinFrequencySafe(PIN_OUTPUT,frequency);
//sets the frequency for pin 3
      D = map(analogRead(1), 0, 1023, 0, 255);
//re-maps the range of analog input A1 from 0 - 1023 to 0 - 255
      pwmWrite(PIN_OUTPUT, D);
//use this functions instead of analogWrite on pin 3

}

void Stoptaste() {
//function for the stop button


      int S;
//define the push button as stop button
      S = digitalRead(2);
//read HIGH or LOW digital signal from pin 2
      if (S == HIGH) {
//if the button is pressed, set t as tmax, then that while loop
condition in function of ausloesemagnet is not true anymore, so the
while loop will be broken until reset.
        t = tmax;
      }

}

void LCD() {

      #include <LiquidCrystal.h>
//use LCD library
```

```
      LiquidCrystal lcd(13, 12, 10, 9, 8, 7);
//initialize the library with the numbers of the interface pins
      lcd.begin(16, 2);
//set up the LCD's number of columns and rows
      lcd.setCursor(0, 0);
//set the cursor to column 0, line 0(line 0 is the first row, since
counting begins with 0)
      lcd.print("T=");
//print "Temperature = "
      lcd.print(temperature);
//print temperature in celsius
      lcd.print("`C ");
      lcd.print("t=");
//print "Time = "
      lcd.print(millis()/1000);
//print current runtime in seconds
      lcd.print("s");
      lcd.setCursor(0, 1);
//set the cursor to column 0, line 1
      lcd.print("D=");
//print "Dutycycle = "
      lcd.print((D*100)/255);
//print current dutycycle in percent
      lcd.print("% ");
      lcd.print("tmax=");
//print "max. runtime = "
      lcd.print(tmax/1000);
//print preset of max. runtime in seconds
      lcd.print("s   ");

}
void Kontrollanzeige() {
//shows the status, if PWM is working

     #define LED 5
     analogWrite(LED, D);
//put a LED and resistor between pin 5 and GND
}
```