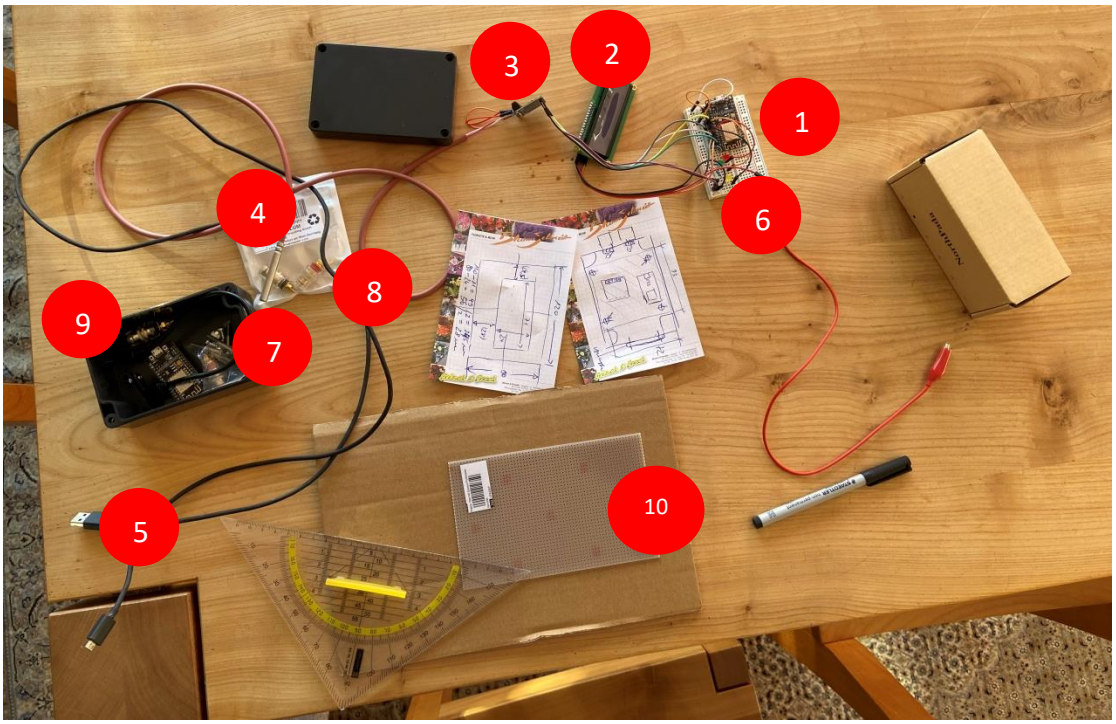
Measurement unit using PT1000 and ESP8266 nodeMcu

This project describes the making of how to read PT1000 sensor analogue data and displaying them on a LCD display.



Minimum items needed for a demo set-up on a breadboard:

1. ESP 8266 nodeMcu (for having later the option to use Wifi e.g., over Smartphone hotspot and send the data as well to a cloud. In this project not realized)	https://amzn.eu/d/4TakPEy
2. LCD blue display 16 x 2, type: HD44780 16x02. See datasheet attached.	https://amzn.eu/d/hfEh3zI
3. MAX31865 interface for PT1000. Note: there is a similar type available for PT100.	https://amzn.eu/d/faUMT1I
4. PT1000 sensor, 2-wire, silicone coated:	https://amzn.eu/d/1PCvCDE
5. USB micro cable for ESP8266 power & data connection.	Example: https://amzn.eu/d/f1JwiUg
6. Breadboard, Jumper wire cables	

Extended items for integration into a housing:

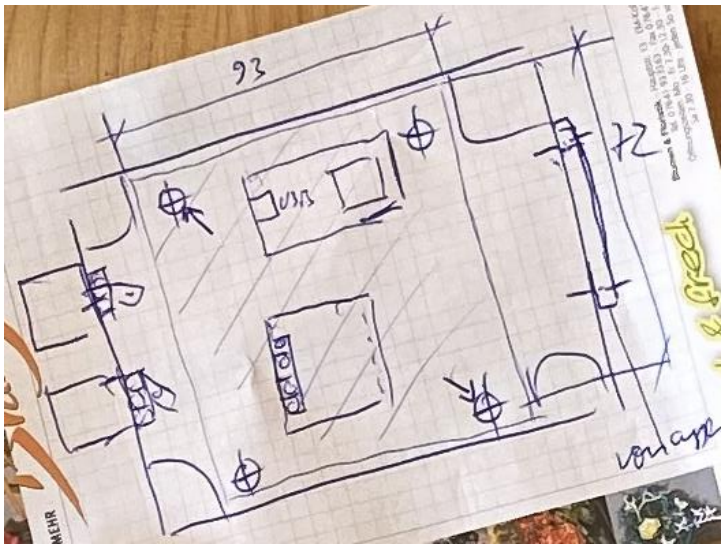
7. Delock Cable USB 2.0 Micro-B Female for Installation > USB 2.0 Micro-B Male 25 cm Used for build-in at housing and easy USB cable connection.	https://amzn.eu/d/fBeKFYv
8. Loudspeaker connectors for easy connection and de-connection of PT1000 to unit:	https://amzn.eu/d/emOS0JA
9. Housing:	https://amzn.eu/d/0O0unNu or for a bit more space: https://amzn.eu/d/emOS0JA
10. PCB Euroboard 160 x 100 mm, perforated grid board,	https://amzn.eu/d/55XhWMN

	coated with copper (adapt size yourself by cutting. Do not forget to coat with protection spray against corrosion)	
11.	**Female Connector Strips** 40 Pin Header Single Series Female PCB Header 2 mm 2.0 mm Single Row Pin Strips for Arduino Connectors	https://amzn.eu/d/3JGmW9z
12.	Miscellaneous srews, soldering station, soldering wire, etc.	

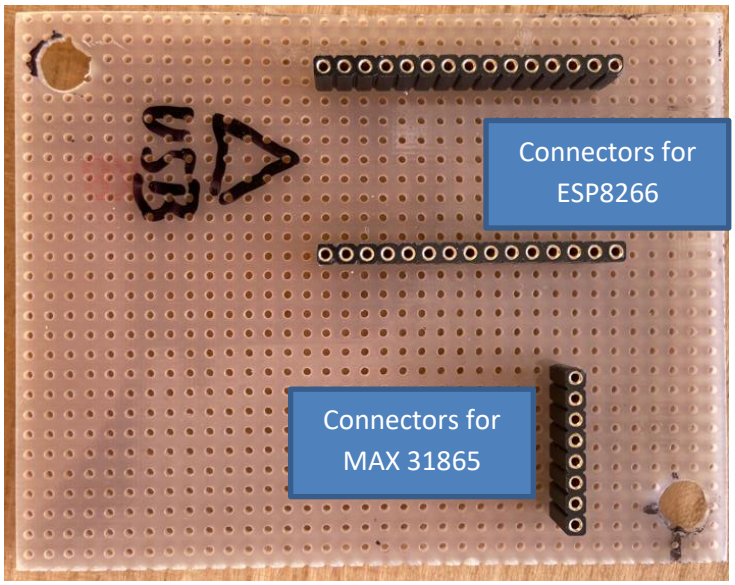
Mechanical:

- The opening for the LCD display must be cut-out by e.g., a Fretsaw for Woodworking. Measure yourself the opening to have the LCD in the centre of the housings cover. Fixation by two screws minimum.

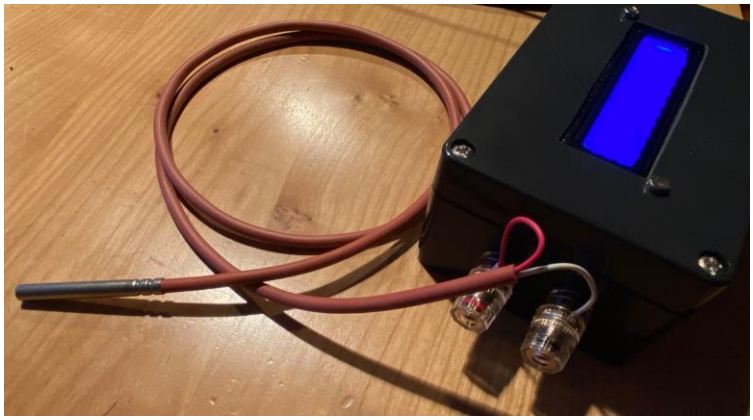
- Layout scribble for placing the PCB inside the housing. Left side connectors for PT1000, right side the USB connection. Centre will be the Euroboard with ESP8266 and the MAX interface.



- The female pins soldered to the PCB Euroboard allow easy and fast exchange of electronic components in case of damages.



- Drill holes at the side to mount the Loud-speaker connectors to housing.



- As well make an opening for the Micro-B Female connector. This will enable to exchange the cable to another one.

Software:

This programme version has **five sub-routines** for easy reading and extension to read values of the PT1000, gives warnings in case the values are not within the nominal operation value of the PT1000. As well if PT1000 is not well connected, the system gives an error warning. To indicate constant processing of the code in the loop, a heartbeat symbol (*) is blinking on the display. Important, if for example the values do not change due to constant temperature.

(copy the code in the left column and insert into the Arduino IDE. Use google translator to translate German comments)

Arduino code	***comments***
/********** By Manfred Koch, December 2022 Following sub-routines are included: 1. **Welcome(); Only executed once after power-up** 2. **PT1000_Wert();** reading the analogue value for temperature 3. **Grenzwertwarnung_auf_LCD();** Warning if values exceed measurement boundaries 4. **Temperaturwert_auf_LCD();** displaying the value on LCD 5. **Heartbeat(); Displays a blinking * to indicated running programme** **********/	

#include <SPI.h>
 	
#include <Adafruit_MAX31865.h>
#include <LiquidCrystal_I2C.h>
 	
// set the LCD number of columns and rows
int lcdColumns = 16;
int lcdRows = 2;
int Heartbeat_flag;
int Messung_Pause = 1000;
 	
// set LCD address, number of columns and rows
// if you don't know your display address, run an I2C scanner sketch
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);
 	
float Temperature_for_LCD;
 	
// Logische Verknüpfung von MAX31865 mit ESP8266 NodeMCU
// Use software SPI: CS, DI, DO, CLK
//NodeMCU pins connected to: D0, D7, D6, D3
Adafruit_MAX31865 thermo = Adafruit_MAX31865(16, 13, 12, 0); // https://randomnerdtutorials.com/esp8266-pinout-reference-gpios/
 	
// use hardware SPI, just pass in the CS pin
	SPI.h:
	https://www.arduino.cc/reference/en/language/functions/communication/spi/
	Library for amplifier must be installed
	Library for LCD Display must be installed
	
	Delay value to determine cycle time
	
	Logic connection of the interface with ESP 8266 inputs

//Adafruit_MAX31865 thermo = Adafruit_MAX31865(10);
 	
// The value of the Rref resistor. Use 430.0 for PT100 and **4300.0 for PT1000**
#define RREF 4300.0
// The 'nominal' 0-degrees-C resistance of the sensor
// 100.0 for PT100, 1000.0 for PT1000
#define RNOMINAL 1000.0
	Set the reference value to operate with PT100 or PT1000 sensor

void setup() {
Serial.begin(115200); // Baud rate
Serial.println("Adafruit MAX31865 PT1000 Sensor Test!");
 	
// *** **set PT1000 wire type**
//thermo.begin(MAX31865_3WIRE); // set to 2WIRE or 4WIRE as necessary
thermo.begin(MAX31865_2WIRE); // set to 2WIRE
	Define connection baud rate in IDE tools
	
	Set-up of two wire sensors

// *** **initialize LCD**
lcd.init();
// turn on LCD backlight
lcd.backlight();
	Note: on the backside of the LCD I2C interface is a potentiometer to adjust contrast !!!

Welcome(); // einmaliger Aufruf bei Start **} // end of set-up**	Call sub-routine Welcome only once after power-on

void loop() {
 	
Serial.print("Neuer Messwert in: ");
 	
delay(Messung_Pause); // Verzögerung neuer Messwert
Serial.print(Messung_Pause / 1000);Serial.println(" s");
 	
// call subroutines
PT1000_Wert(); // Werteermittlung
	Messages only for showing in serial monitor!
	
	Call sub-routine to read value from PT1000 sensor

// *** Grenzwerthinweis
if ((Temperature_for_LCD < -40.0) || (Temperature_for_LCD > 150)) {
 	
Grenzwertwarnung_auf_LCD(); // rufe Warnmeldung auf
delay(1500);
lcd.clear();
}
else {
Temperaturwert_auf_LCD();
}
 	
Heartbeat();
	Print warnings on LCD if nominal values in negative or positive temperature range are exceeded
	
	Call sub-routine for warnings on LCD display
	
	Call sub-routine to display values on LCD
	
	Call sub-routine to alternate display of a “*”

```
  Serial.println("* * * * * * * * * * * * * * * * ");


} // end of loop


// *** Untergrogramme / Sub-routines

void Welcome() {
  Serial.println(" Hello, I am ready ");
  lcd.setCursor(0, 0);
  // print message to LCD
  lcd.print("* Hello, I am ready *");
  delay(3000);
  lcd.clear();
}


// *** Ermittlung Temperaturwert PT1000

void PT1000_Wert() {

  uint16_t rtd = thermo.readRTD();

  Serial.print("RTD value: "); Serial.println(rtd);
  float ratio = rtd;
  ratio /= 32768;
  Serial.print("Ratio = "); Serial.println(ratio,8);
  Serial.print("Resistance = "); Serial.println(RREF*ratio,8);
  Temperature_for_LCD = (thermo.temperature(RNOMINAL,
RREF));
  Serial.print("Temperature = ");
Serial.println(Temperature_for_LCD);



  // Check and print any faults
  uint8_t fault = thermo.readFault();
  if (fault) {
    Serial.print("Fault 0x"); Serial.println(fault, HEX);
    if (fault & MAX31865_FAULT_HIGHTHRESH) {
      Serial.println("RTD High Threshold");
    }
    if (fault & MAX31865_FAULT_LOWTHRESH) {
      Serial.println("RTD Low Threshold");
    }
    if (fault & MAX31865_FAULT_REFINLOW) {
      Serial.println("REFIN- > 0.85 x Bias");
    }
    if (fault & MAX31865_FAULT_REFINHIGH) {
      Serial.println("REFIN- < 0.85 x Bias - FORCE- open");
    }
    if (fault & MAX31865_FAULT_RTDINLOW) {
      Serial.println("RTDIN- < 0.85 x Bias - FORCE- open");
    }
```

symbol

if (fault & MAX31865_FAULT_OVUV) {
Serial.println("Under/Over voltage");
}
thermo.clearFault();
}
Serial.println();
//delay(800);
}	

// *** Anzeige auf LCD Display	
void **Grenzwertwarnung_auf_LCD()**{	Showing temperature value warnings

if (Temperature_for_LCD < -40.0) {
lcd.setCursor(0, 0);
lcd.print("PT1000 Grenzwert:");
lcd.setCursor(0,1);
lcd.print("-40");
lcd.print((char)223); // degree symbol für LCD 16x2
lcd.print("C erreicht");
}	

if ((Temperature_for_LCD > 150.0) && (Temperature_for_LCD < 988.0)) {
lcd.setCursor(0, 0);
lcd.print("PT1000 Grenzwert:");
lcd.setCursor(0,1);
lcd.print("+150");
lcd.print((char)223); // degree symbol für LCD 16x2
lcd.print("C erreicht");
}	

if ((Temperature_for_LCD > 988.0)) {
lcd.setCursor(0, 0);
lcd.print("PT1000 Anschluss:");
lcd.setCursor(0,1);
lcd.print("pruefen");
}	

}	
void **Temperaturwert_auf_LCD()** {	Showing temperature value on LCD

Serial.print("I am sending to LCD: "); // auf serial Monitor
Serial.println(Temperature_for_LCD);
Serial.println();
//lcd.clear();	

// set cursor to first column, first row	
lcd.setCursor(0, 0); // print message	

lcd.print("Temperatur:");
delay(50);
// clears the display to print new message
//lcd.clear();
// set cursor to first column, second row
// set the cursor to column 0, line 1
// (note: line 1 is the second row, since counting begins with 0):
lcd.setCursor(0,1);
lcd.print("PT1000");
 	
// Fixe Positionierung der Temperaturwerte bei wechselnden Vorkommawerten im Zehnerbereich oder Minuszeichen (PT1000 Werte kann -40 bis +150)
 	
if (Temperature_for_LCD >= 0 && Temperature_for_LCD < 10) {
lcd.setCursor(7,1); // minus weg
lcd.print(" ");
lcd.setCursor(8,1); // 10 Zahl weg
lcd.print(" ");
lcd.setCursor(9,1); // fixe Position Zahlenwert
}
 	
if (Temperature_for_LCD >= 10 && Temperature_for_LCD < 100) {
lcd.setCursor(7,1);
lcd.print(" ");
lcd.setCursor(8,1); // fixe Position Zahlenwert
}
 	
if (Temperature_for_LCD >= 100) {
lcd.setCursor(7,1); // fixe Position Zahlenwert
}
 	
if (Temperature_for_LCD < 0 && Temperature_for_LCD > -10) {
lcd.setCursor(7,1); // fixe Position Zahlenwert
lcd.print(" ");
lcd.setCursor(8,1); // fixe Position Zahlenwert
}
 	
if (Temperature_for_LCD < -10) {
lcd.setCursor(7,1); // fixe Position Zahlenwert
}
 	
lcd.print(Temperature_for_LCD);
lcd.setCursor(14,1); // fixe Position von degree & C symbol
lcd.print((char)223); // degree symbol für LCD 16x2
lcd.print("C");
delay(50);
//lcd.clear();
 	
}
 	
void Heartbeat() {
	Conditions to fix the cursor position and compensate digit numbers

```
// *** soll wechselnden den * oder nix zeigen

  if (Heartbeat_flag == 1) {
    lcd.setCursor(15,0); // fixe Position Heartbeat, damit bei
konstanten Termperaturwerten eine Prozessoraktion gezeigt
wird
    lcd.print("*");
    Serial.print("Hearbeat sign: ");Serial.println("*");
    Serial.print("Hearbeat flag: ");Serial.println(Heartbeat_flag);
    Heartbeat_flag = 0;
  }

  else if (Heartbeat_flag == 0) {
    lcd.setCursor(15,0); // fixe Position Heartbeat, damit bei
konstanten Termperaturwerten eine Prozessoraktion gezeigt
wird
    lcd.print(" ");
    Serial.print("Hearbeat sign: ");Serial.println(" ");
    Serial.print("Hearbeat flag: ");Serial.println(Heartbeat_flag);
    Heartbeat_flag = 1;
  }

  Serial.println();



}
```

This project was designed and software was written by Manfred Koch based on following sources:

https://learn.adafruit.com/adafruit-max31865-rtd-pt100-amplifier/arduino-code

https://learn.adafruit.com/adafruit-max31865-rtd-pt100-amplifier/rtd-wiring-config

https://www.esp8266.com/viewtopic.php?t=17166#

https://randomnerdtutorials.com/esp8266-pinout-reference-gpios/

https://lastminuteengineers.com/arduino-1602-character-lcd-tutorial/

Idea for extension:

- Integrate “OTA” (Over the air, wireless) programming
- Integrate Thingspeak connection to show values as graph on a widget and store values in the cloud