

## **Soldering station for PTC based soldering irons**

This project is released under GNU General Public License as published by the Free Software Foundation either version 2 of the license or (at your option) any later version and the CERNOSHW Version 1.1 License.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

### **Credits**

The LCD routines of the firmware are written by Erik Häggström <xpress@xpress.mine.nu>

### **Purpose of the project**

The soldering station project realizes a soldering station for soldering irons that measure their temperature at the tip by using the heating element's PTC characteristic.

Some time ago I needed a soldering iron for soldering SMD parts with 0.5mm pin pitch. I decided to solder the parts using the well soldering procedure, where some solder is held into a well within the soldering tip. The Ersa Micro Tool was one of the first to offer these kind of tips. The iron itself is not cheap at all. But it offers good quality for the money. The Ersa soldering station for the iron is very expensive. So I build my own.

As I do not like complicated user interfaces for soldering irons, I designed the station so that it has only one rotary encoder to set the temperature and one LED to show its operation. An LCD is used to show the desired temperature and the real temperature.

Please note that I build my station using a piece of prototyping PCB. I never build it using the PCB included in this project. I only designed the PCB for my projects website. But there are no DRC errors found by Altium-Designer. So the board should work. Please report any problems to me.

The station should also work for other 24V soldering irons if you change some calibration values within the firmware.

V. Besmens, January 2013

### **Included in the project**

- This PDF document
- Part list in PDF
- Firmware source and HEX file for the ATmega8 written in GNU C (WinAVR package and AVRStudio4. Refer to main.c file for details on optimization-, linker- and fuse-settings
- Schematic, Layout (Altium designer and PDF files) and extended-Gerber files for the display board and encoder board

### Technical data

- Up to 80W at 24V for solder irons with PTC heating element based temperature control
- Simple user interface with only one rotary encoder and 2 line LCD panel to set the temperature. Last temperature set is stored
- PID based temperature control
- Ability to detect connected solder iron (although not implemented in firmware)

### Circuit description

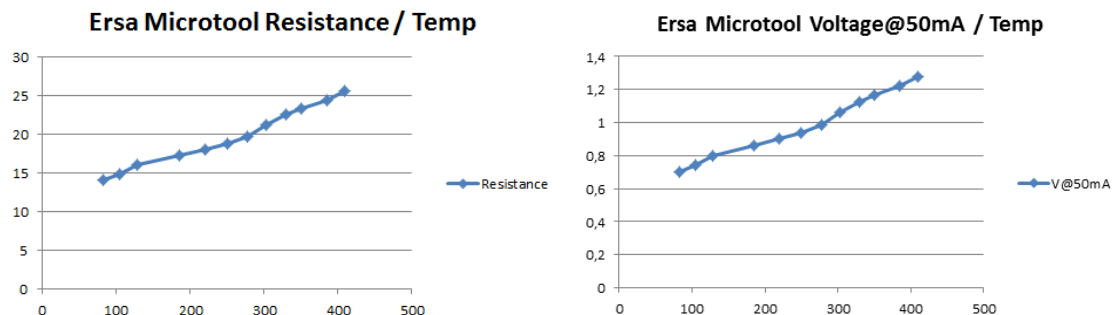
The circuit and firmware are relatively simple:

A P-Channel driver FET (Q1) that is driven by a NPN bipolar transistor (Q2) switches the solder iron on and off. The PWM frequency is 1 Hz with a resolution of 1ms. It is controller via the ATMEL ATmega8. Every one second, the driver FET switches off to allow temperature measurement. A constant current source (U3) always puts a current of 50mA thru the iron. The voltage drop on the irons PTC is measured in the off state using the AD converter of the ATmega8 that uses its internal 2.56V reference. The value measured is fed into a PID loop to calculate the next PWM value. A Watchdog timer is used for safety reasons.

To calculate the slope and offset of the resistance versus temperature characteristic, I measured some values for the ERSA Micro Tool:

Temp °C	Resistance of PTC	Voltage @ 50mA	ADValue @ 2.56VRef
82	14	0.700	280
105	14.8	0.740	296
128	16	0.800	320
185	17.2	0.860	344
220	18	0.900	360
250	18.7	0.935	374
278	19.7	0.985	394
303	21.2	1.060	424
330	22.5	1.125	450
350	23.3	1.165	466
385	24.4	1.220	488
410	25.6	1.280	512

The PTC shows an overall linear characteristic:



### What you need to build the board

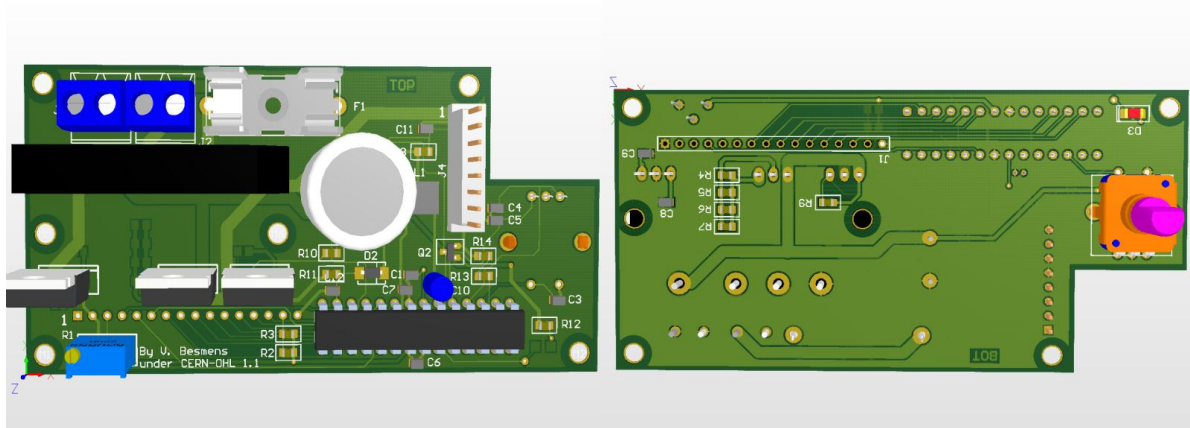
- A PCB-manufacturer that accepts extended Gerber files to create the dual layer PCB's. The PCB's are based on 6mil / 6mil (0.15mm / 0.15mm) technology for structures and gaps with a minimum hole diameter of 0.4mm.
- The ability to solder SMD parts with 0805. A programmer that is capable of programming the ATmega8. A parallel programmer is suggested because the serial programming pins are used by the circuit.

### The PCB

The PCB is populated on both sides. The LCD, rotary encoder and LED and some other components are mounted on the bottom side. The Heatsink I used is available from Reichelt (a German component distributor) and must be cut to a length of around 46mm. The parts must be mounted electrically isolated onto the heatsink. You can of course use a different kind of heatsink.

The LCD mounts onto the bottom side of the PCB using a 16 pin single row 2.54mm pitch pin-strip. Put some foam between the main PCB and the LCD as it is only held by the pin-strip. The LCD I used is available from Electronic Assembly. If you are using a different model you will probably have to connect it using single wires.

The pictures below show the populated board's top- and bottom side without the heatsink.



The board needs an external power supply of 24V AC or DC with 60W or 80W depending on the soldering iron. You can either use a separate 24V supply (some 3<sup>rd</sup> party notebook supplies can deliver 22V or 24V and are pretty cheap. Make sure that you get a power supply that has a separate earth ground pin) or a (toroid-) transformer. The mains switch must be wired externally. Earth ground must be connected to GND and the soldering iron using J3.

The soldering iron is connected via J4. Please refer to the schematic for details.



The light of the LED can be displayed on the front using a short light pipe.

### What you need to rebuild the firmware sources

A pre-build HEX-File for programming the controller is included within the project files. If you want to build the firmware from the sources, you will need WinAVR and AVRStudio4 which are both free. Make sure that you use the right fuse settings also described in main.c.

### Changing the Firmware

It will probably be necessary to reverse the direction of the encoder in the software depending on the encoder type used. To do this, simply change the sign of the following values in the main routine:

```
TempEncoder=Encoder();  
if (TempEncoder!=0)  
{  
  if (TempEncoder==1)   
  {  
    EncoderChanged=true;  
    EncoderSave=false;  
    DesTemp+=1;  
    if (DesTemp>450) DesTemp=450;  
  }  
  if (TempEncoder==1)   
  {  
    EncoderChanged=true;  
    EncoderSave=false;  
    DesTemp-=1;  
    if (DesTemp<100) DesTemp=100;  
  }  
}
```

If you want to use another iron, calculate the offset and slope of the PTC heating element by doing some measurements at different temperatures of the iron. Then change the values of the "Slope" and "Offset" constants.

The ATmega8 also measures (at its pin 28) the value of a resistor within the connector of the soldering iron. I think it is used to determine, which iron is connected to the station. You can change the firmware by adding irons with different sense resistors to the "GetSolderIronType" routine and by using different values for "Slope" and "Offset" for specific irons.

You will probably have to measure the value of the reference voltage at pin 21 of U1 and change the value of the "VRefMV" as the internal reference of the ATmega8 is not very accurate in its absolute value.

## References

ERSA GmbH (MicroTool): <http://www.ersa.de>  
Electronic Assembly (LCD): <http://www.lcd-module.de>  
Reichelt Elektronik GmbH: <http://www.reichelt.de>