Jim Koehler, VE5FP

2258 June Rd., Courtenay, BC V9J 1X9, Canada; **jark@shaw.ca**

Reflow Soldering for the Radio Amateur—Revisited

Automate a simple toaster oven for reflow soldering.

In a previous article[1] I described how a simple toaster oven could be used for reflow-soldering by radio amateurs. There have been a number of similar descriptions on the internet but, in my opinion, the secret of doing it successfully is to monitor the oven temperature. In the previous article, I described how to do this using a thermocouple-based electronic thermometer. It worked nicely but it quickly becomes tedious with manually turning the oven on and off to get the correct temperature profile. In this article, I describe how to automate the process so that it is necessary only to push a button to start and a simple controller does the rest.

The simple controller I used is the Raspberry Pi Zero W (R-Pi). This wonderful little computer costs only $10 yet it includes a Wi-Fi interface so that program development is very easy and the controlling program can be written in a high-level language. It is a gross example of over-kill to use such a powerful computer to control something as simple as an oven but it is hard to beat the price and the ease of development. Total parts cost will depend on what you have in your junk box but, even in the worst case, can't cost you more than about $100. For that price, you get a reliable reflow-solder oven, which can be controlled by just three push-button switches.

Description

For this project you will need a toaster oven, a solid-state switch capable of handling 20 A, an R-Pi, a small dc motor with a fan blade, a tiny temperature sensor, a temperature sensor interface, and a few LEDs and push-button switches. Suitable toaster ovens can often be found for under $25 new, or less for used ones at garage sales. Adafruit[2] sells the temperature sensor interface ($15) as well as the R-Pi ($10). The rest of the small parts and a dc power supply can be bought from Digikey or Mouser, or may be found in your junk box. For the dc power, I use an old laptop computer power supply. These are ubiquitous and supply around 18 V at up to 2 A, but any dc supply of 12 V or so at 1 A is sufficient. The dc supply goes to two TO-220 linear regulators that power the dc motor and the R-Pi.

The small dc motor is mounted on the back of the oven with the fan blades inside. It is not meant to create a strong wind inside the oven but rather to just stir the air so as to keep the temperature more or less constant across the volume of the oven. I found a 9 V dc motor with a fan blade already on it in my junk box. Similar small motors can be found in surplus stores or on the internet.

I used a Tyco SSRT-240D25 solid-state switch. It is rated to control up to 240 V ac at 25 A. Since my toaster oven takes only about 1 kW from 120 V ac, this switch is operated conservatively. Similar switches can be found on eBay for around $10 postpaid.

The circuit diagram is shown in Figures 1 and 2. Part of the circuit (Figure 2) is built on a pc-board[3], a "Pi-hat" that plugs onto the R-Pi. The rest (Figure 1) is just wiring on a chassis. The chassis (Figure 3) is constructed partly from sheet aluminum and partly from 0.1" thick ABS sheet. It is not totally metal because the R-Pi communicates with your local area network (LAN) via Wi-Fi, so it should not be inside a shielded enclosure. The metal part holds the ac receptacle that your oven plugs into as well as the dc power input to the circuitry.

After you have constructed the instrument, you must connect to your LAN in order to control the R-Pi when the oven is being calibrated. You will not need to have a separate keyboard, mouse and monitor to operate the R-Pi; it can all be done through your LAN from your computer. After calibration, the network connection is no longer required and you can control your oven with just the push-button switches mounted on the instrument. So, you can use the oven in your shop even if you do not have Wi-Fi or a computer there.

A very good description of the reflow-solder process and a discussion of the various parameters can be found in a paper by R.C. Lasky[4]. For the toaster oven running at 1 kW, a solder profile similar to Lansky's Figure 1 is easily achieved. On my oven I was able to get a temperature rate of increase of about 1° C per second. This is about the middle of Lansky's recommended heating range. The desired temperature profile is to steadily increase temperature until the solder melts and flows, hold at this temperature for 10 or 20 seconds and then cool. To do this, the controller turns the oven on and when the temperature reaches the desired peak temperature, the oven is switched off and an LED is lit to alert the operator to open the oven door to allow the pc-board to cool. This sounds too easy, and it is! When the oven is switched off, the temperature will continue to 'coast' upward and the actual peak temperature might be greater than what is desired. To compensate for this, the oven must be calibrated so that the controller switches the oven off before reaching the desired peak temperature. From that point

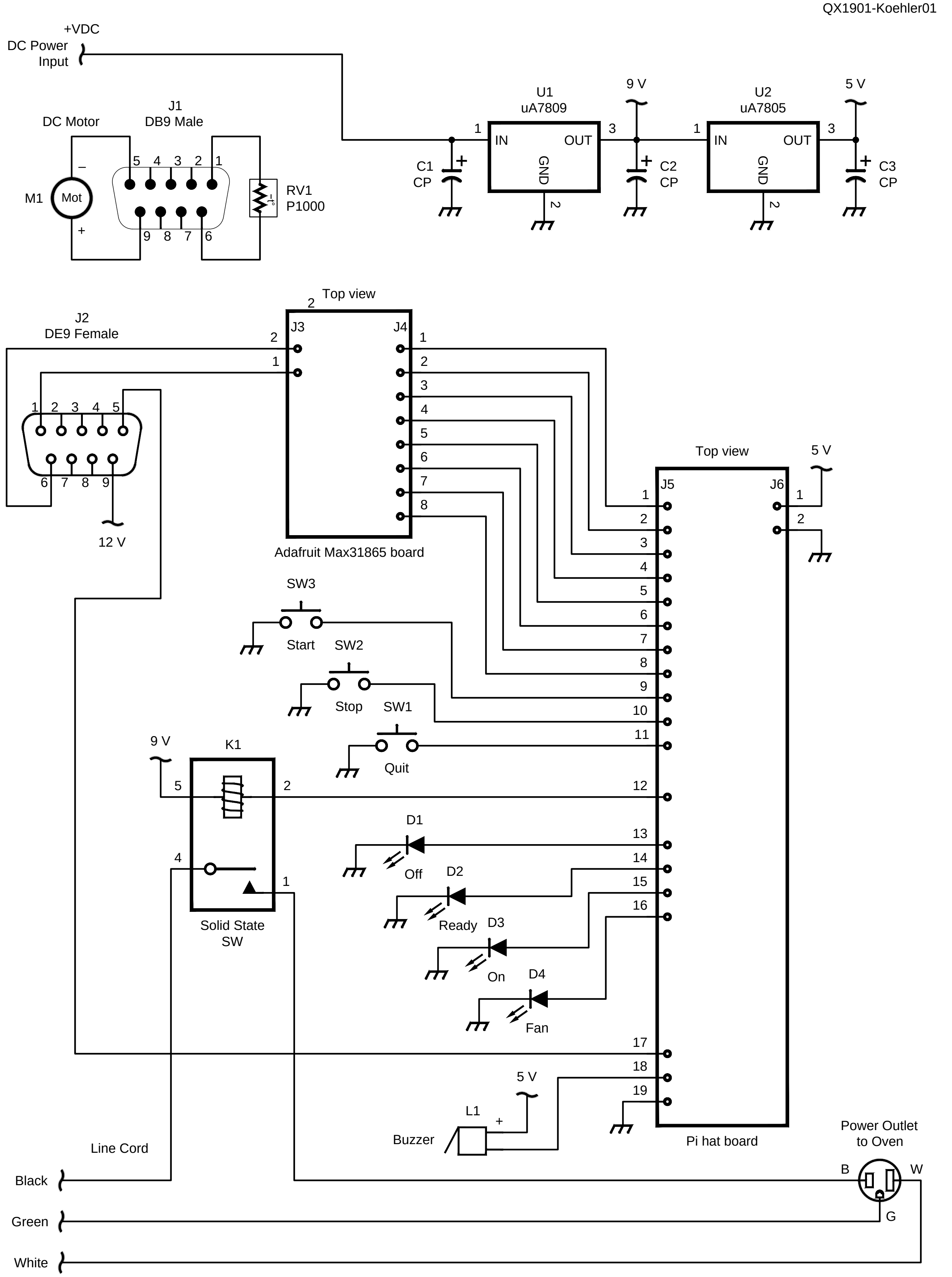

Figure 1 — Schematic portion of the point to point wiring on the chassis.

it will just coast to the desired peak. I have written a software routine — described later — to do this calibration very simply. The desired peak temperature will depend on the type of solder you use. The commonly used is 60/40 solder (60% lead, 40% tin) has a melting point of 183° C and a 'liquidus temperature'[5] — the temperature at which it flows freely — of 188° C. We want to be sure that the pc-board temperature reaches at least this temperature, so normally the oven is set to raise the air temperature to something greater than this value; say 210 to 215° C, and to hold it at this temperature for 10 or 20 seconds or so. Most integrated circuits can easily withstand these temperatures for tens of seconds. Indeed, in Europe, lead-bearing solder is no longer allowed in production electronics, so modern integrated circuits must tolerate much higher temperatures than what non-leaded solders require.

Construction

The metal bottom and back plate were

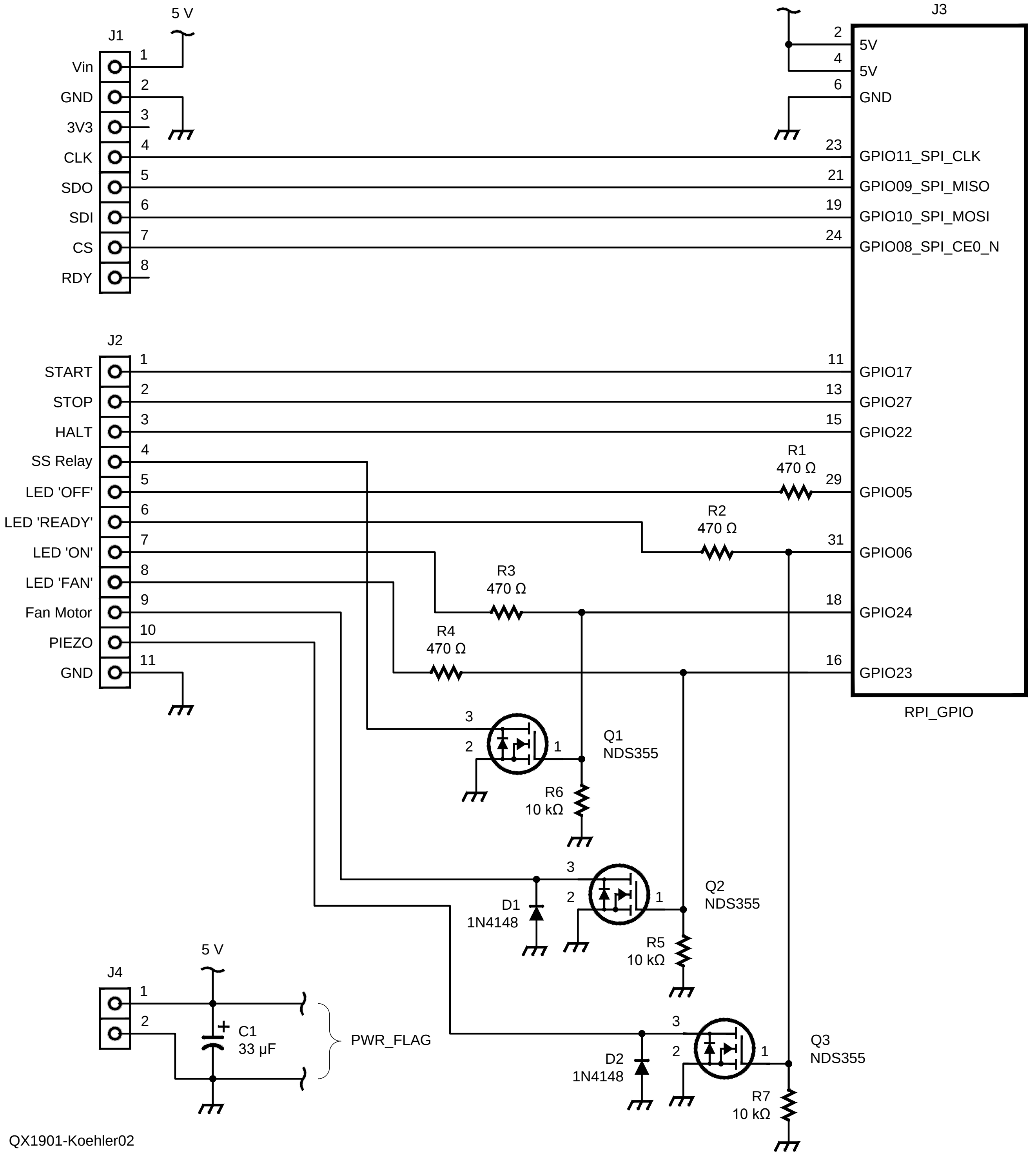

Figure 2 — Schematic portion that is built on a pc-board.

folded from a single sheet of aluminum (Figure 3). The 120 V ac receptacle that the oven plugs into is mounted on this back surface. Also mounted there is the dc jack compatible with the dc power supply, in my case the laptop power supply. Two wires go to the dc motor and another two wires go to the temperature sensor mounted in the oven. These four wire connections from the controller to the outside were made using four pins of a DE-9 connector. The male connector is on the four conductors going to the motor and the sensor, and the female connector is on the chassis. Also on the back plate of the chassis is a 120 V ac input cable that goes to the oven receptacle via the solid-state switch. I cut a piece of 0.1" ABS sheet to cover the bottom of the chassis and to make the front panel, and used more 0.1" sheet ABS to make an outside cover. ABS is nice to work with and it is easy to glue together. You may use either Krazy Glue® or ABS cement. The ABS sheet I had was a bilious yellow so I spray-painted it with a less garish pale yellow color and used water-slip decals to make the labels. The LEDs used in the project were just press-fitted into holes drilled in the front panel.

I had previously used thermocouple sensors and their interfaces for sensing temperature but for this project I decided to use a platinum temperature sensor and the MAX31865 as the interface. PT1000 platinum temperature sensors use a thin film resistor made from a platinum alloy that has a resistance at 0° C of exactly 1000 Ω. The resistance change with temperature is very well known and the MAX31865 uses a ratio-metric method that results in remarkably stable, accurate and well defined temperatures. In general, the accuracy surpasses that of thermocouples. Inside the oven, we want the sensor temperature to be very close to that of the air surrounding the target pc-board. This means that the sensor should have very little thermal mass. I chose US Sensor PPG102C1 (DigiKey part number 615-1045-ND). This is a small chip measuring about 0.08" by 0.12". It must be mounted inside the oven so that it is close to the pc-board being soldered, and it needs to be insulated electrically from the oven. I used an approximately 6" piece of 1/4" Teflon™ rod that I drilled with a 1/8" bit through its length. I used a ¼-28 die to thread one end of the Teflon rod. Figure 4 shows the rod (top image), the sensor soldered to a twisted pair of Teflon-insulated stranded wire with the solder joints covered with small heat-shrink tubing (middle image), and (bottom image) the sensor pulled into the end of the rod.

The Teflon rod acts as a heat insulator so the inside solder joints to the sensor should not actually melt when the air in the oven reaches solder melting temperatures. The threaded end of the rod is used to fasten the rod to the back surface of the oven using a ¼" hole. This hole should be placed so that the sensor end of the rod is only about ¼" to ½" above the grill where the pc-board will be placed.

The small dc motor is mounted on the outside back surface of the oven using long hex spacers. The back surface of the oven gets very hot and the spacers somewhat insulate the motor from it. The motor was mounted so that the fan blade was just an inch or so away from the inside back surface of the oven. Figurer 5 shows the motor mounted on the back surface of the oven (upper image), an inside view showing the fan and the temperature sensor (middle image), and an outside back view of the assembly (lower image) showing the motor and sensor mounting.

I had replaced my badly rusted oven grill with a thin aluminum plate used as a shelf. I made a small table from wire mesh that sits on this shelf. The pc-board to be soldered is put on this little mesh table, that way both the top and bottom surfaces of the pc-board are exposed to the heated air inside the oven. If your grill is clean, you can dispense with the aluminum shelf and the mesh table, and just place your pc-board on the grill.

The 'Pi-hat' pc-board is assembled using 0805-sized surface-mount passive components. This size is not too difficult to solder by hand. There are also three SOT-23 sized HEXFET transistors. For those who

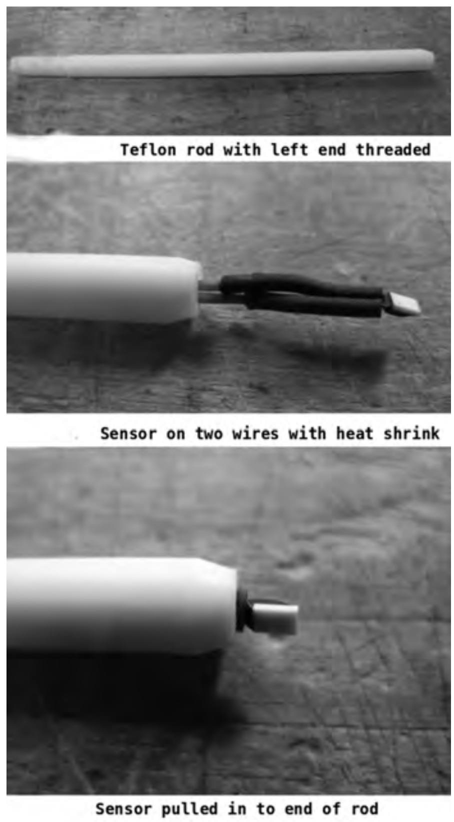

Figure 4 — Teflon rod (top image), the sensor soldered to a twisted pair of Teflon-insulated stranded wire with the solder joints covered with small heat-shrink tubing (middle image), and (bottom image) the sensor pulled into the end of the rod.

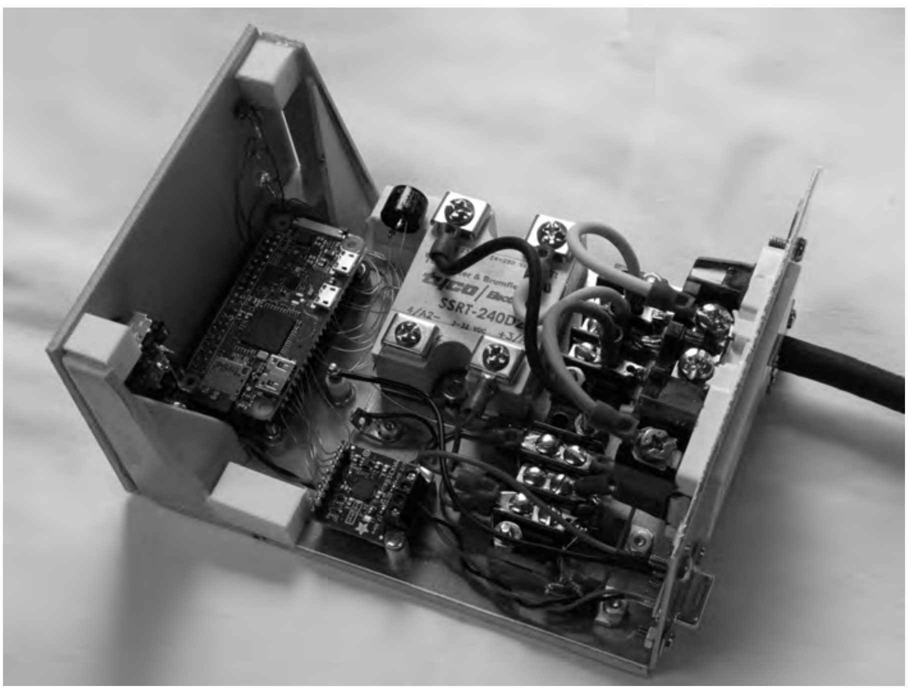

Figure 3 — The chassis is partially constructed from sheet aluminum.

Figure 5 — The motor mounted on the back surface of the oven (top image), an inside view showing the fan and the temperature sensor (middle image) and an outside back view of the assembly (bottom image) showing the motor and sensor mounting.

Figure 6 — The Pi-hat mates to the R-Pi via the female 40-pin header on top of the Pi-hat and the male 40-pin header mounted on the bottom of the R-Pi.

prefer not to work with SMD components, the assembled board is available, see Note 3. To mate the Pi-hat to the R-Pi, I have put the female 40-pin header on top of the Pi-hat and the male 40-pin header is mounted on the bottom of the R-Pi (Figure 6). This way the R-Pi sits on top with its Wi-Fi antenna clear of obstructions. The normal way of mounting would have obscured the antenna of the R-Pi. The Pi-hat on the bottom has two single-row male headers on it for connection to the rest of the circuit.

Referring to Figures 2 and 6, the eight connections of J1 connect to the Adafruit MAX31865 board. For J1, pin 1 is on the right side of the left-most 8 pins. The right-most 11 pins, J2 in Figures 2 and 6 connect to the LEDs, switches, the motor and optionally a buzzer. J2 pin 1 is adjacent to J1 pin 1. I placed these two headers adjacent to one another so that a single un-cut 19-pin piece of male header can be used to make both headers J1 and J2. To interface between the Pi-hat and the Adafruit MAX31865 board you can use jumpers, solder wires or wire-wrap. Similarly, from J2 to the LEDs and switches the end going to J2 can be jumpered or soldered or wire-wrapped. I used wire-wrap for both. To prevent the pins of the header on the Pi-hat from contacting and shorting to conductors on the bottom side of the R-Pi, I glued a strip of plastic on the bottom of the R-Pi and I also nipped off the tops of the Pi-hat header pins. Instead of using straight male headers as I did, you could also use a row of right-angle male headers. That way the male pins would point away from the Pi-hat board instead of sticking up from it.

There is a pin on the Pi-hat to connect an optional buzzer. This buzzer is activated at the same time as the READY LED so it will sound when the reflow-solder process controlled by the R-Pi has finished.

Power to the oven comes into the enclosure via a strain relief. Be sure to use an appropriate size of line cord and the wires from the solid state relay to the ac outlet. They must handle about 10 A if your oven needs 1 kW. The oven is turned off and on via the solid-state relay. Please be very careful about this portion of the wiring!

Setting up the R-Pi

The R-Pi is normally used with a separate keyboard, mouse and monitor. However, it is also possible to use it ‘headless’. That means you can connect to it using your own computer with its keyboard mouse and monitor through your LAN using the R-Pi Wi-Fi capabilities. To do this, you must connect automatically to your LAN when it is turned on. The first step is to load the pre-configured Linux operating system onto a micro-SD card. It is available as a file you can download from the Raspberry web page[6]. The needed image file is *Raspbian Stretch* with *Desktop*. This image file is more than 4 GB in size. You then edit a few files on it, according to the instructions in an Adafruit tutorial[7] and then ‘burn’ this image file into a micro-SD card using a free program called *Etcher*[8]. There are free versions of *Etcher* available for Linux, Windows and Mac OS. The micro-SD card should be at least 8 GB in size and fairly fast. There is no maximum size limit. These days it is hard to find micro-SD cards smaller than 16 GB and the speed of these is sufficient for the R-Pi operating system. I used the most inexpensive 16 GB card I could find. To transfer the image file to the micro-SD card you must access the card from your computer. The easiest way to do this is to put the micro-SD in a USB adapter. Then, when you plug in this adapter your operating system will detect it. You can then burn the image onto the card using *Etcher*.

If you then plug the micro-SD card into the socket on the R-Pi and power it up, either with the R-Pi just by itself using a mini-USB power supply or by putting it into the circuit you have just built, you will see some activity on the little LED close to the mini-USB socket on the R-Pi. After 20 seconds or so, it will have booted up and connected to your LAN. Your router will assign the R-Pi an address on your LAN. But what is that address? There are programs to find out and I recommend the *Angry IP Scanner*[9] available for free for Windows, Mac and Linux operating systems. When you do a scan of your LAN, you will see the R-Pi appear with a URL. On my home LAN it appeared at **192.168.1.26**.

Use *SSH*[10] to connect to the R-Pi. *SSH* is a secure protocol for connecting from one machine to another on a network. If your main machine is Linux or a Mac, you will already have *SSH* on it and you can access it by opening a terminal. If you have Windows 10, then it is already part of the operating system but it is not enabled by default. You can find instructions on the internet to do so[11]. In the terminal, to connect to the R-Pi, you simply type: **ssh pi@192.168.1.26** where you replace my address with the address that you got from the *Angry IP Scanner*. You will have to answer a question and then you will be connected to the login screen of the R-Pi asking for a password. The password is 'raspberry'. After logging in, you will be the user named 'pi' sitting at a terminal with full control of the R-Pi.

To power down the R-Pi — don't do it just now — just type the command: **sudo halt** and the R-Pi will disconnect itself from the network and power down. You can tell when it is done as the little green LED will go out.

We can also access the R-Pi by using a protocol called Virtual Network Computing (VNC). This is all described in another Adafruit tutorial[12]. These instructions tell you how to install a VNC server in the R-Pi, how to install the VNC client in your computer and how to set up the R-Pi so that the VNC server starts up automatically on boot-up.

Figure 7 — The R-Pi on the computer screen is a generic GUI interface.

From now on, you can use the VNC client installed on your computer to log in to the R-Pi and you will then have a GUI interface to the R-Pi. In the material that follows, I assume that you are interacting with your R-Pi using VNC. The R-Pi appears on your computer screen as a window as shown in Figure 7. This is a generic GUI interface. If you left-click on an icon, it does the default thing and if you right-click on an icon, you get a list of possibilities. To start with, if you left-click on the third-from-left icon on the top row, you are invoking the file displayer and it will open a window and show you the contents of the home directory on the R-Pi. If you then close the file displayer window and left-click on the fourth-from-left icon on the top row, you will invoke the terminal and if you now type '**ls**' on the terminal line, you should see something like the listing shown in Figure 7. Notice that the response is not anything like that of your main computer. I recommend that you change the host-name of the R-Pi and the default password especially if you operate the oven within range of your LAN.

Finally, we must install some software that is used in the various oven programs. Python, a high-level language is already installed. We need to also install *gnuplot*, a program for plotting data and *feh*, a program for displaying graphics files. We install these from the internet by first invoking a terminal and then typing,

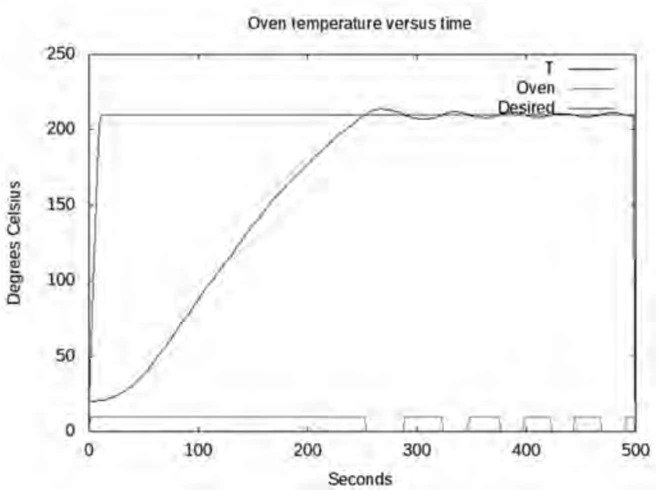

Figure 8 — Temperature variation vs. time with Kd equal to zero.

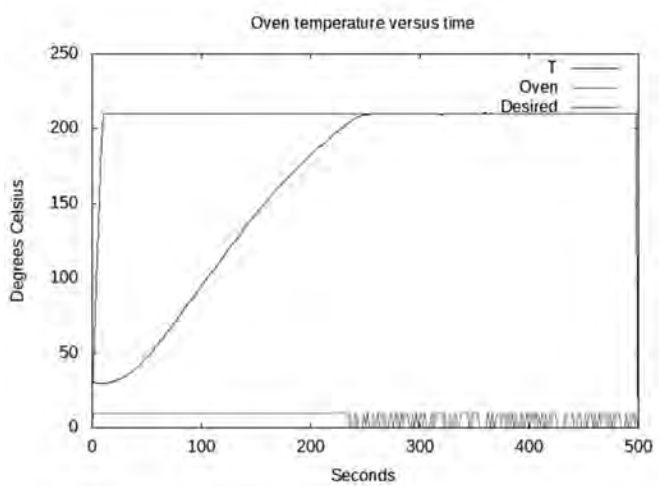

Figure 9 — Temperature variation vs. time with Kd set -9.

sudo apt-get install gnuplot feh

We also need the software I've written to handle the oven. This is available in the file, '*pi_oven.zip*', from the **www.arrl.org/QEXfiles** web page. You can access using the R-Pi web browser and download it into the R-Pi. It will appear in the R-Pi download directory. Using the R-Pi file handler, extract all the files in it into the R-Pi **/home/pi** directory. The R-Pi is now completely set up.

Setting Up the Oven

The oven operates as a negative feedback system. The controller determines the error between the desired temperature and the actual temperature and switches the oven on and off so as to minimize the error. The particular algorithm is called the *proportional integral and differential* (PID) method. In the case of this oven controller, because the entire solder-reflow process takes just a few minutes, we dispense with the integral of the error so, properly, this algorithm is the PD method. The temperature error e is the desired temperature Td minus the actual temperature Ta at any time,

$$e = Td - Ta$$

so the value s used to turn on the oven is,

$$s = Kp \cdot e + Kd \cdot \frac{de}{dt}$$

If s is greater than zero, the oven is turned on. If s is zero or less, the oven is turned off. Here Kp is a constant of proportionality and Kd is the differential constant. The differential de/dt is the rate of change of error per second. We can arbitrarily assign a value of 1 to Kp and make some measurements to determine the value to use for Kd. Its value will depend on your particular oven.

In the program used to control the oven the value of Kd is read from a file named "k_d.txt". This file consists of one line holding one number. Click on the file from within the VNC window to open it with a text editor and you will see that is just the number -9.0. This is the value that works for my oven. I have written a test program, called "test_oven" that you can use to determine the appropriate value for your oven. Using a text editor, open "k_d.txt" and replace the number with zero (0.0). Then, with the oven connected, run the test_oven program by double-clicking its icon in the VNC window and then clicking on "Run in terminal". This will run the oven with a desired temperature of 210° C for five minutes. When the program has completed, you will see a graph displayed in the VNC window showing how the temperature varied with time. Figure 8 shows what I saw with Kd equal to zero. The "Desired" is the flat trace at 210°, the "Oven" is the trace at the bottom, and the "T" is the sloping trace of oven temperature. The temperature overshot the desired value by about 5°, then oscillated back and forth around the target temperature. Change the value in the k_d.txt file by trial and error so that the temperature comes up to the desired temperature and just overshoots a fraction of a degree. Kd will be some negative number. If it is too large a negative number, the temperature will approach the desired temperature too slowly. If it is too small, it will overshoot too much. Figure 9 shows the graph when the value in k_d.txt is -9.0, the value that works best for my oven.

The oven is ready to go once the correct value of Kd has been determined. I have written a Python program *oven.py* that does the job of turning on the oven and going to an appropriate temperature and then turning the oven off. The program is designed to have the oven follow a temperature 'profile' that is just a file containing a time sequence, one second at a time, of what the desired temperature is. Since the whole profile might last over many minutes, this file will contain many lines of data with one new line every second. It would be tedious in the extreme to have to write out this file for every new situation. I wrote a second program that takes an abbreviated profile which just has an entry every time the desired temperature changes and produces the final temperature profile that the *oven.py* program requires. My abbreviated short profile file for eutectic or 60/40 solder contains the lines in Table 1.

At time zero, the desired temperature is 20° C, at 1 second, the desired temperature is 210° C, this is desired until time 301 seconds when it drops to 20 again and finally, the whole process ends at 360 seconds. This is the normal file that I use for normal boards using 60/40 solder. I wrote a Linux script file, named *oven* that first invokes the program to read the short eutectic solder file and write the desired profile file and then invokes *oven.py* to follow that profile.

Doing the Reflow-Soldering

Once you have gotten the best value for Kd and written that value into the k_d.txt file, the oven is ready for use. After you have prepared the pc-board by applying the solder paste and placed the components, you just need to put it in the oven and then run the program. Double-click on the "oven" icon, select "Run from terminal" and from then on the process will be controlled by the three push-buttons. Start (see Figure 10) begins the solder-reflow process. The oven will come up to 210° C. Then the Ready LED will start to flash and the oven will turn off. At this time, you should open the oven door to allow the pc-board to cool. If you built in the optional buzzer, it will sound simultaneously with the Ready LED. That

Table 1.
Abbreviated short profile file for eutectic or 60/40 solder.

0	20
1	210
300	210
301	20
360	20

Figure 10 — Front panel of the Pi-Oven controller. Push Start to begin the solder-reflow process.

is all that there is to it. The Stop button will stop the process at any time that you press it. The Quit button is used when you are done for the day. It makes the R-Pi go through its shut-down process. Note that you must Start the process first for the Quit button to be active. When the program has gone through its sequence, a graph of temperature *vs*. time will be displayed in the VNC window.

Although it is nice to see the graph of temperature *vs*. time for each solder process, it is not very convenient to have to control the oven using a computer. It is very simple to have the oven program run every time that the R-Pi boots up. That way, you can still control it using the three push-button switches and you do not need an internet connection. To make the program run by default upon boot up, you just edit a file in the home directory called **.bashrc**. Notice the period before the name: it is important. From a terminal we can use the simple text editor built into the R-P nano and open the file for editing by typing,

sudo nano .bashrc

This brings up a window showing the existing contents of the file. Simply scroll down to the bottom and add the following line followed by an *Enter*,

./oven&

Then save the file by typing *control-x*, and answering yes. Now when the R-Pi is rebooted it will automatically run the oven program. This means that you can control it using just the push-button switches.

Figure 11 shows the oven next to the controller. When you're done using the oven program, it is considered good practice to shut down the R-Pi gracefully by pressing first the Start button and then when the oven comes on, hold the Quit button for a few seconds. Give it ten seconds or so to shut down and then you can unplug the power to the unit.

Figure 11 — The oven shown next to the controller.

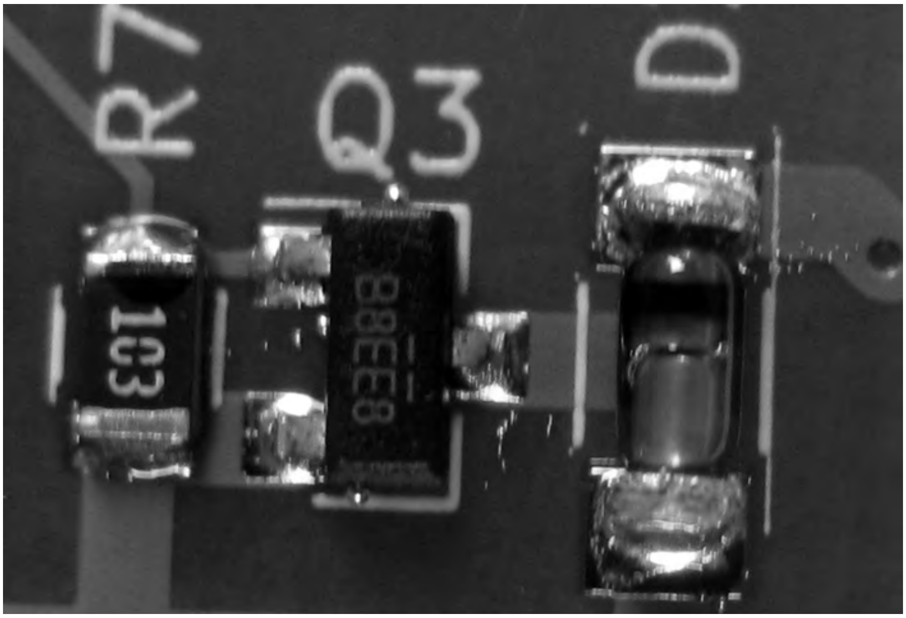

Figure 12 — Portion of a solder re-flowed pc board.

Conclusion

The modified toaster oven does an excellent job of reflow-soldering. Preparing boards for reflow-soldering is relatively easy and it does not matter too much if every pad does not have a precisely correct application of solder paste. Figure 12 shows a portion of the Pi-hat board, used in this project, and with the solder paste applied by hand, as it comes out of the oven. The solder joints are bright and clean. I normally wash a completed board in isopropanol using a tooth-brush to gently brush the components to dissolve and remove any solder flux. Do not use common rubbing alcohol which is isopropanol diluted with water.

Jim Koehler, VE5FP, earned his first ham license at age 15 in 1952. He completed undergraduate university, then was awarded a post-graduate degree in Australia. He returned to Canada in 1966, and became Professor of Physics and Engineering Physics at the University of Saskatchewan, and did research in upper-atmospheric physics. After retiring in 1996, he and his wife moved to Vancouver Island to enjoy Canada's best climate. Jim shares a hobby interest in electronic design with friend Tom Alldread, VA7TA. Both have extensive "junk boxes" and so are able to scrounge parts from one another freely, and cooperate with each other in taking advantage of all the new integrated circuits, which make life so much easier than when they were much younger.

Notes

[1]Jim Koehler, VE5FP, "Reflow Soldering for the Radio Amateur", *QST*, Jan., 2011, pp 32-35.

[2]**https://www.adafruit.com/product/3648**.

[3]The "Pi-hat" bare board is available from Jim Koehler, email **jark@shaw.ca**, for $15 postpaid. The board with all the surface-mount components already assembled on it is available for $25 postpaid.

[4]**https://kicthermal.com/wp-content/uploads/2012/11/Best-Practices-Reflow-Profiling-98675.pdf**.

[5]**www.farnell.com/datasheets/315929.pdf**.

[6]**https://www.raspberrypi.org/downloads/raspbian/**.

[7]**https://learn.adafruit.com/raspberry-pi-zero-creation/overview**.

[8]**https://etcher.io/**.

[9]**angryip.org/**.

[10]**https://en.wikipedia.org/wiki/Secure_Shell**.

[11]**https://www.howtogeek.com/336775/how-to-enable-and-use-windows-10s-built-in-ssh-commands/**.

[12]**https://learn.adafruit.com/adafruit-raspberry-pi-lesson-7-remote-control-with-vnc/overview**.