***ROUGH DRAFT***

**Graduation Project:**

**Reflow Oven: Design and Prototype**

By:

Joshua Irwin

April 10th, 2014

**Reflow Oven (Rough Draft)**

**Current Soldering Process at North Penn:**

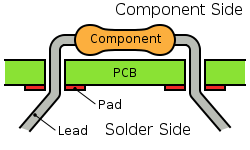
North Penn’s Electronics program does a lot of work with designing their own circuits and making projects. Currently the highest level of soldering that the Electronics courses partake in is through-hole soldering. This method involves taking an averaged size component and inserting the leads through holes in the printed circuit board, then making a solder joint between them through the use of a soldering iron. This is a good process to learn the basics of soldering and how circuits work, but it comes with downsides. Some of which include bulky boards, leads on components that can easily snap, and components that can take up lots of surface area, leading to larger circuit boards. While through-hole soldering is a good skill to learn, just about all electronics used today use a type of component classified as a Surface Mount Device (SMD). These components are much, much smaller than the traditional component and allow circuit boards to become much smaller while still doing the same functions.

Diagram 1: Standard through-hole soldering to a circuit board.

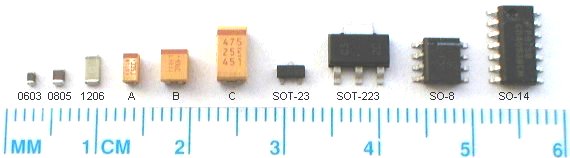
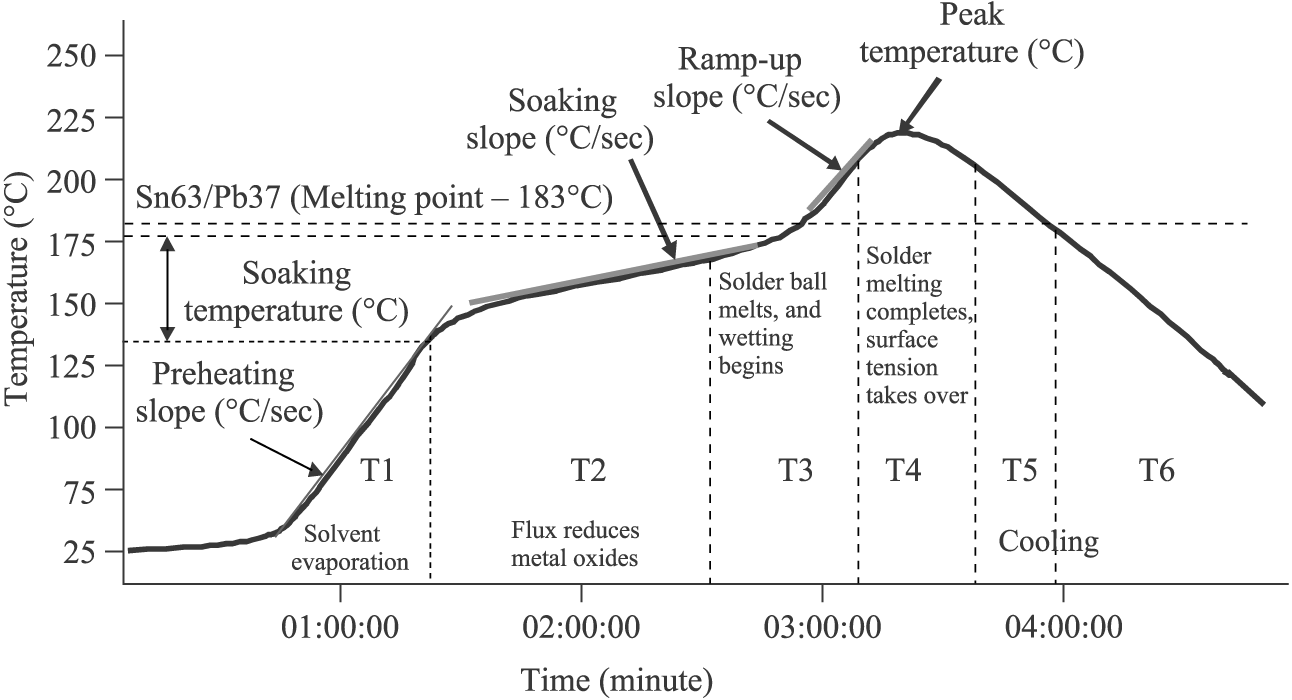
While these new Surface Mount Devices provide many benefits, North Penn never really used them due to the fact that they are very small and are a great challenge to solder to a board with a standard soldering iron, without burning the component or losing it in a solder joint. These components are usually soldered to a circuit board through the use of a reflow oven. These ovens control the internal temperature so that these components make a full solder joint on to the board without damaging the component. Professional reflow ovens can cost thousands of dollars, but with the knowledge gathered from a year and a half in the electronics course and through personal research, we were able to manufacture one for our own use at a fraction of the cost.

Diagram 2: Size of standard Surface Mount Devices (SMD).

**What Is Reflow Soldering:**

When working with Surface Mount Technology, the normal soldering that North Penn utilizes would not be sufficient enough to make circuit boards with surface mount components efficiently. This caused use to realize that we needed to utilize a device called a reflow oven. What a reflow oven does is monitor the temperature inside and adjusts it as necessary so that the solder melts fully to form a solid solder joint and at the same time making sure that the oven doesn’t increase or decrease too rapidly, causing thermal shock. When compared to through-hole soldering, reflow soldering requires less work once you have a functional oven, but much more precision due to the fact that both the solder and the flux are contained in a single mixture called solder paste. This paste has to be carefully applied to the circuit board so that is covers the component leads, but not too much so that it will bridge between the two leads, causing a short.



On top of monitoring the internal temperature of the reflow oven, the oven must also follow a specified temperature curve so the solder joint is of high quality. This is done by ensuring that the solder alloy and the base metal reach the minimum soldering temperature for a sufficient amount of time. This temperature curve is called a thermal profile and consists of five stages.

Diagram 3: Standard Reflow Temperature Profile.

The first stage is called the Ramp-to-Soak (RTS) stage. In this stage, the oven goes from ambient temperature to about one hundred and fifty degrees Celsius, plus or minus twenty degrees. Once the temperature is around one hundred and fifty degrees, the second stage, called the Soak stage begins. It is here that the board and solder alloy rise in temperature for about forty-five seconds so that they are around the same temperature. Once the time has passed, the third stage, called the Ramp-To-Peak stage begins. This stage is essentially the same as the Ramp-To-Soak stage as it simply raises the temperature to about two-hundred and seventeen degrees Celsius, at about one to three degrees per second. Once the temperature reaches two hundred and seventeen degrees Celsius, it holds the temperature for about forty-five seconds as well to ensure that the solder bonds evenly with the joint. Once the given time has elapsed, the cooling stage begins. The cooling stage essentially just turns off all heat output and limits the temperature drop to about one to three degrees per second. This limit is imposed so that the board and component don’t suffer from thermal shock and cracking. If every stage goes well, there should be a high quality solder joint between the board and the component.

**How We Plan To Make Our Own Oven:**

To make a reflow oven of our own, there are many thoughts to take into consideration. Before we start to build anything, we need a clear outline of what we want to do and just how automated we want this to be. We needed to plan out what components we required to complete this project and how they would eventually all interact with each other in the end. We did a lot of research on how reflow ovens worked and what temperature specifications they had to reach in order to follow a successful temperature profile. Once we had a general of what we were doing, we started looking at parts we would need. We knew we would need a DC to AC relay, an Arduino, a K-Type thermocouple, a MAX13855 Thermocouple Amplifier from

The 25 Amp relay we used. Pin 1 was connected to one side of live AC; pin 2 was connected to one side of the toaster oven wiring; pin 3 was connected to pin 6 on the Arduino; and pin 4 was connected to ground.

Adafruit, and a standard toaster oven. Thankfully we already had a toaster oven, Arduino, and thermocouple with amplifier, so that just left obtaining a relay,

Our goal was to have the Arduino control the relay, which would in turn control the toaster oven based on inputs received from the thermocouple. We wanted to have the Arduino constantly reading the temperature inside the toaster oven, and when a push button was pressed, it would begin the pre-heat stage. Once the pre-heat stage was started, the relay would turn on, allowing the toaster oven to turn on. When the oven reached about one-hundred-forty-five degrees Celsius, the relay would constantly toggle on and off, keeping the temperature stable for about forty-five seconds. After this time has passed, the relay would turn back on and the Ramp-to-Spike stage would begin, increasing temperature to about two-hundred degrees Celsius. When the temperature reached this point, it would hold for about fifteen seconds and then the cooling stage would begin. Once in the cooling stage, the relay would turn off, turning the oven off and a fan mounted on the back vent of the toaster oven would turn on. This would help extract the hot air out of the toaster oven, allowing for a faster cool time, and when the temperature also dropped to about seventy degrees Celsius, we could then prop open the door, allowing more cold air in and at this point we wouldn’t have to worry about thermal shock. Now that we had this basic plan for the reflow oven, our next step was to start programming the Arduino so that it could accomplish the tasks we wanted.

**Our Biggest Obstacle and Objective (Arduino):**

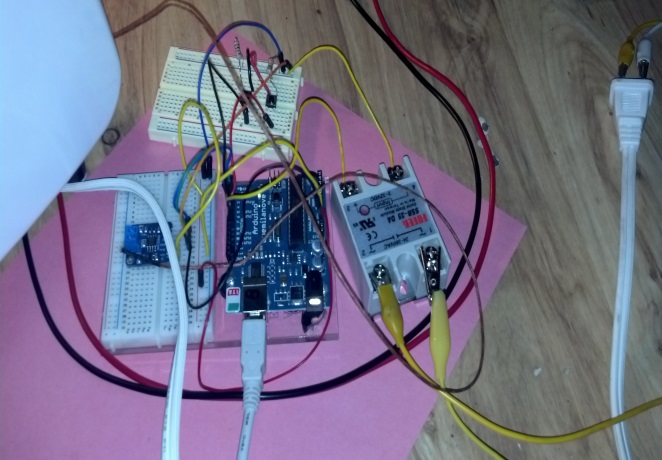
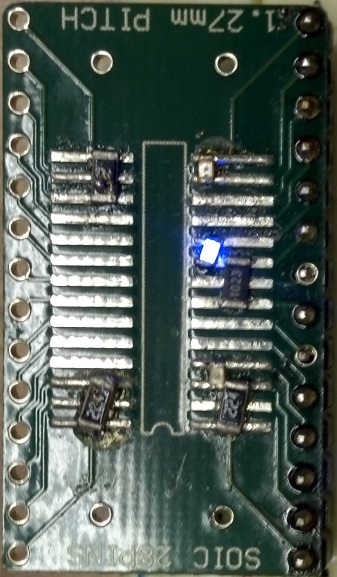
Planning the reflow oven and figuring out how it was going to work was the easy part of this project. The most difficult part was taking the Arduino and learning how to code in Arduino C so that we could use it to accomplish what we wanted in regards to the reflow oven. The Arduinos were brought up briefly in the first year electronics course, but after not working with one for over a year; we had to pretty much reteach ourselves how they worked and how to code again. We started out by relearning basic Arduino codes such as making a LED blink, and getting simple readings from an analog input device and then displaying them on to the Arduino’s serial monitor. From there the next step was taking the input readings that were displayed on the serial monitor and then using them to control the relay. This in essence would control the relay depending on what the temperature reading was and determine if the heating element should be on or off at this point. This was done through the use of an if else loop. In an if else loop, the Arduino is looking for a certain integer, if that integer meets certain criteria, then the Arduino preforms a function, if that criteria isn’t met, then the Arduino preforms a different function that is specified in the else loop. This was a basic bit of code said that if the thermocouple read less than a certain temperature, then turn the relay on. When the temperature was then reached, the relay would then turn off due to the else loop. This loop, while simple, was a major stepping stone to getting the rest of the project to work. Once we proved that the Arduino could control the power going into the reflow oven, we then knew we could complete our goal, it was just a matter of how much we needed to add to the code to refine what was going on and to create a stable temperature profile. Once we had the basic on/off loop set up, our next task was making each stage its own statement in the code. By making each stage a statement, we were then able to edit each one individually so that we could edit the times and temperature values for each. This also made the actual program loop a lot easier to understand since it became much smaller by splitting everything up. It took us awhile to figure out how statements worked in Arduino, but we soon figured it out so that each stage was its own statement and could be called in the program loop. Once we had everything set we had to figure out how to have everything timed so that each stage would stay at its max temperature for a set amount of time. This was probably the hardest part of all the programming we had to do, we at first had a delay set to 45000, which is equal to forty-five seconds, we soon realized how this would start counting the second the stage began, which in turn meant that no stage ever reached completion and the entire process was done in about a minute and a half, without the oven ever reaching temperatures greater than seventy degrees Celsius. We needed to reevaluate how we were going to time this so our next revision tried having the stage reach its max then beginning the timer, we soon realized this wouldn’t work either since if we tried this approach, the heating element would either stay on or turn off once the max was reached and the temperature wouldn’t remain stable, after more thought we ****looked into researching for loops. The basic setup of a for loop is “for (i=0;i<45;i++)”, what this loop does is set an integer, i, equal to zero and declare a statement underneath it, then while i is less than forty-five, it will continue running the loop until i reaches forty-five. We quickly realized we had to move the i++ statement to the end of the loop where the relay was turned off so that it would increment only if the max temperature was reached. Once this was done we realized we had all the basic code compiled that we needed to make this work and we moved into our testing phase.

Diagram 4: Everything connected for testing.

**Testing the Reflow Oven:**

Once our code was compiled and uploaded into an Arduino, we connected everything together so that we could begin testing. Before we started putting components on a board and tossing them in the oven, we decided that we needed to do a few dry runs first to see what our temperature profiles looked like, and edit values in the code if needed to fix it. We ran our first test and saw that the preheat stage was nearly non-existent. Where it was supposed to hold for about forty-five seconds, it held for maybe one. There were also a lot of fluctuations in the data table due to placement of the thermocouple; this would cause issues when the temperature reading is what is telling the Arduino what to do. Another thing we noticed was that our cooling stage was way too steep, if we left it like this, we would run the risk of causing thermal shock, which could damage the board and the components. We went back and fixed some values in the code, and moved the thermocouple around so that it was placed right above where the circuit board would be, without touching anything else. We then ran our second test and the curve we got was much better. When we plotted the data points, we were able see how the oven held the preheat temperature for a while before it began its Ramp-to-Spike stage, we also noticed that the cool down curve was not as steep and we could pin point where we opened the oven door due to the temperature drop. Once we determined that the temperature profile was sufficent for what we needed to do, our next step was to actually take a surface mount LED and solder it onto a printed circuit board. We used a material called solder paste, which is a combination of both solder and flux, and placed a smalled amount of it onto the metal leads on the board. We then placed a small LED inbetween both points using a pair of fine tweezers and made sure that the LED was in solid contact with both of them. We put the board in the oven and let it go through its stages, once the cooling stage was over we pulled out the circuit board and tested to see if there was a solid solder joint between the two leads. We tested the joint with a pair of tweezers and it was stable so we moved on to testing it it would light up or not. We applied 5 volts to the LED but we learned fast though that surface mount LEDs, and most likely all surface mount components, have a much smaller operating voltage comapared to the through hole componets. While we managed to get the LED to light, it was light by the milivolts of voltage output by a continuity teser andwe wanted to have it run off a normal power supply we have in class. This requiered us to set up a led and a resistor in series that that way there would be a voltage drop and we could use the LED without fear of having it burn out. This lead to our second reflow test, we used a 12000 ohm reisitor and put it in series with a new LED and put the board back into the oven. When the reflow cycle was complete we pulled the board back out and this time wired it up to a ten volt power supply, this time we were able to get the LED to light up and see that it wasn’t going to burn out.

**Future Improvements:**

While the reflow oven we have just built is indeed functional, at this point it is really more of a proof of concept, showing that we are able to reflow solder by modify an existing toaster oven with an Arduino. From here we have recognized many ways that we and future classes can improve upon this device to make it more functional and increase it autonomy. One item we thought about adding was a LCD screen. This screen could show what the temperature reading inside the device is at all times, removing the need for a separate computer to look at the serial monitor, this screen could also show what stage the reflow oven is at and approximately how much time is left. At first this would seem like a daunting task due to having to add more code to the Arduino to make this work, but in reality, after looking at a few example codes, the LCD screen seemed really easy to implement, and our current and future revisions actually has the code already in it, just waiting to be wired to a screen. The next improvement that we thought of was adding a servo onto the door of the reflow oven. What this servo would do it prop open the door as the cooling stage has reached a certain point, allowing for more air to escape and decrease the cool down time. This would be fairly easy to implement since it would only be involved in one part of the code. We also thought about adding a buzzer along with the servo, once the temperature had reached a certain point, we could then program the buzzer to go off, and indicating the reflow cycle was over. This would be as easy to implement as the servo and both would help with the automation of the reflow oven. These would both allow the person doing the soldering to be able to walk away and do something else and then be informed once the cycle was over so that they continue with what they were working on. One more addition that would be nice to implement would be a cancel button. This button would allow the reflow cycle to be canceled at any time without hitting the reset button on the Arduino itself. This would be a nice addition so that we could put everything into a case without having to worry about hitting the button, but the only downside is that the cancel button would probably be the hardest part to implement. This would be difficult due to the fact that we would have to code the Arduino to constantly read the cancel pins state no matter what so that when it was actually pressed, it would turn off all power to the oven.

Our next goal once we get all the components set in place would be to make this reflow oven its own self-contained unit. We plan on doing this by making our own circuit board for the circuit we have developed and then placing it all inside a box that will have an outlet mounted onto it. This would allow us to just plug in any toaster oven we have, and it would be able to act as a reflow oven with the Arduino controlling it. This would greatly increase the lifetime of this unit since if anything happens to the one toaster oven we do have, the reflow circuit will remain unharmed, and be able to work again once a new oven was found.

**Final Thoughts:**

All in all, this is probably the most involved I’ve been with a project. A lot of time was spent researching about reflowing and reading papers detailing the process. I have learned a lot about Surface Mount Devices during this time and a lot about the reflow process. This project also has taught me a lot about coding and how multiple components work. Even though this project was tough and is still currently in progress, it has been a fun project and every step felt like a small accomplishment. I plan to continue work on this reflow oven until the school year is over and try to improve it as much as possible in an effort to make reflow soldering a straight forward process for the electronics classes.

Bibliography:

* Vivari, John. "First Principles of Solder Reflow." Nordson.com. EFD Inc., n.d. Web. 10 Apr. 2014.
* Bergenthal, Jim. "Reflow Soldering Process Considerations for Surface Mount Application." KEMET Electronics Corporation. KEMET, Nov. 1997. Web. 11 Apr. 2014.
* Tarr, Martin. "Reflow Soldering." Reflow Soldering. University of Bolton, n.d. Web. 10 Apr. 2014.
* Mounting of Surface Mount Components." Texas Instruments. National Semiconductor, Aug. 2000. Web. 9 Apr. 2014.