University of British Columbia
Electrical and Computer Engineering
ELEC291/ELEC292 Winter 2017
Instructor: Dr. Jesus Calvino-Fraga
Section 201

Project 1 – Reflow Oven Controller

Group #: B4

Student Name
Adrian, Clarence
Gao, Jian
Lee, Matthew
Luo, Edward
Windsor-Lewis, JoJo
Yu, Jacky

Date of Submission: February 28, 2018

Table of Contents

1.0 Introduction

This project report provides documentation on the architecture and design of a reflow oven controller prototype by students of Electrical Engineering at the University of British Columbia (UBC). Entailed is a detailed account of our team's design process, including idea generation, data collection, problem specification, solution assessment, and full illustrations of our design and how we applied appropriate engineering knowledge to do so. This document is written with the criteria of being addressed towards a reasonably expert reader in the field of engineering such as a project manager.

1.1 Design Objective

The objective of the project entails the software and hardware design of an oven controller accurate enough to conduct reflow soldering, a process commonly used to temporarily attach electrical components on a printed circuit board (PCB) by the use of solder paste. Design stages include the building, programming, and testing of the prototype, along with the implementation of extra functionalities, with an allocated work time of a few weeks. Below are the further requirements for the project:

- Programmed in assembly language
- Able to measure temperatures between 25oC and 240oC using a K-Type Thermocouple
- Able to operate a standard off-the-shelve 1500W toaster using a Solid State Relay (SSR)
- Basic User Interface and Feedback Functionality:
    - Selectable reflow profile parameters (soak temperature, soak time, reflow temperature, reflow time) using pushbuttons
    - Operation of EEPROM using switches

    - Temperature, reflow state, and running time displays using an external LCD
    - Oven temperature display using 7-segment LED displays
    - Start/stop pushbutton
    - Beeper feedback to produce necessary signals
    - Temperature strip chart plot through serial port connection to a personal computer in python
- Error/exception handling: automatic cycle termination
- Temperature validation data to show ±3°C temperature accuracy for the given range

Additional learning objectives:

- Acquire knowledge of assembling PCBs with SMT components by reflow soldering
- Understand the components of a reflow oven controller
- Implement a thermocouple as a method of measuring temperature
- Control and load AC currents throughout different electrical components using a Solid State Relay

The list above is a summarized requirement list obtained from the course-provided lab manual which covers each requirement in more detail (see Appendix I)

1.2 Specifications

1.2.1 Hardware Specifications

- Atmel AT89LP51RC2 8-bit Flash Microcontroller IC 80C51 MCU 32K FLASH 40-DIP
- LCM-S01602DTR/M Liquid Crystal Display Module 4/8-bit HD44780

- General Purpose Thermocouple
- TI OP07CP Op Amp
- FT93C66A-ITR-T IC EEPROM 4KBIT 2MHZ
- BO230XS Serial USB adapter
- 500W Lab Oven
- SSR box
- 0.1uF capacitors
- 330Ω resistor
- LED 5MM GREEN
- 22.1184MHZ Crystal
- EFM8 boards

1.2.2 Software Specifications and Functionality

- AT89LP52 Assembly Software Specifications:
  - Finite State Machine
  - Interrupt Service Routine Timers
  - Error safety measures

- ADC/output conversion
- Application of EEPROM
- External LCD display

- PuTTy (SSH and telnet client)
  - Password-protection interface through serial port connection
  - Display of the temperature value

- Python
  - Temperature Strip Charts
  - Temperature Validation Table

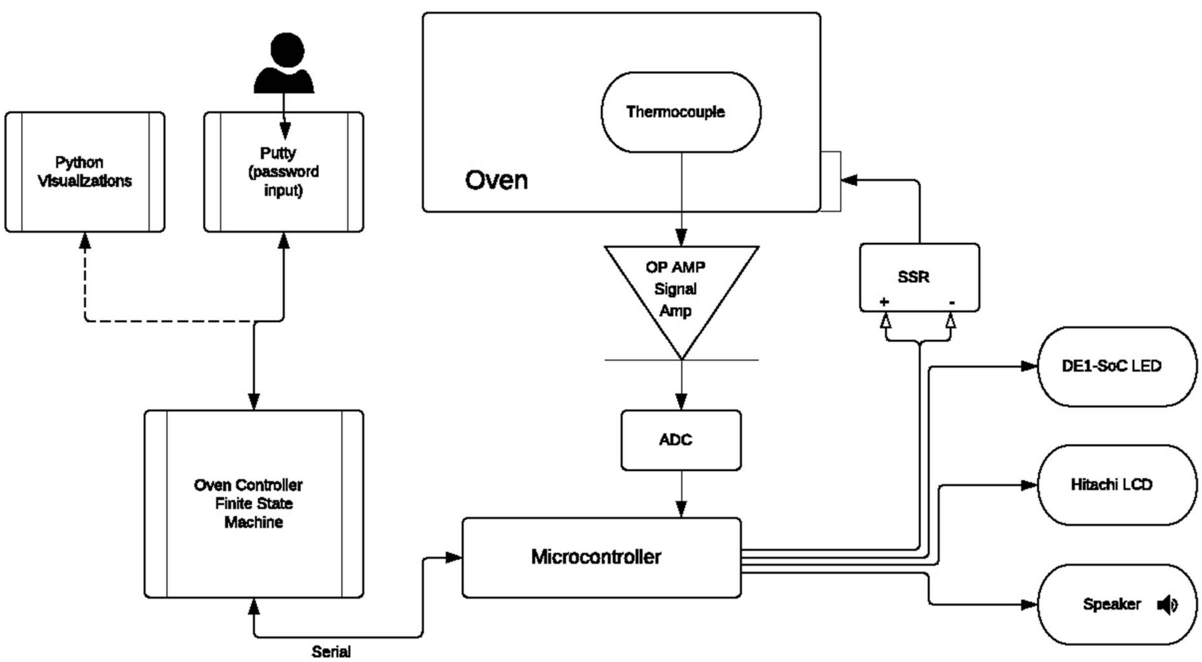

Figure 1: Full System Block Diagram

2.0 Investigation

2.1 Idea Generation

Our first task in this project was to identify the stakeholders involved in this project and to form a clear idea of their needs. With this in mind, we could begin generating ideas for our solution towards this task. Designs were evaluated and narrowed down, with our best contenders being modelled to allow direct comparison between designs. Through elimination, we selected what we felt would be the most effective design for our Reflow Oven Controller - and through iterative design processes, we were able to revise this design into our final prototype and move forward into the next stage of the project.

2.2 Investigation Design

As part of investigation design, we identified ant noted the most important pieces of information to collect and observe throughout the entire design process. Prior to the start of the project, steps we took include gathering information on the stakeholders of the project and the design requirements needed in order to meet their needs, collecting data on the equipment and tools available, including their detailed specifications and limitations from data sheets and lab manuals, and finally formulating a detailed design plan based on the information collected. A detailed list of the apparatus available to us was compiled for our project can be seen in the hardware and software specifications in Section 1.2 of this document.

In order to maximize efficiency and safety experimentation and analysis of the performance of our device, we determined that there were components that were particularly important to keep track of, especially during testing. These include:

- Temperature readings from each test conducted, as they are needed to compare with the design requirements and past/future tests
- Group-wide documentation of errors and iterations made toward the design

2.3 Data Collection and Synthesis

Data collection was conducted and analyzed to fulfill the criteria set. We used the relevant tools and devices available as part of the project. Below are the different components of data collected, and the procedures we took towards collecting them.

Temperature Readings

Temperature readings of the oven were taken regularly and made an imperative part of the project as it is the main factor contributing towards the success of reflow soldering. In order to maximize ease of access to this data, temperature readings were displayed in a variety of ways.

i. Temperature readings obtained from simple voltage reading measurements were shown in real-time on the DE1-SoC's 7-segment LED display during operation of the device (Figure 2a). Measurements were taken using the lab multimeter and converted to display temperature values.

ii. Temperature readings obtained from a K-type thermocouple were shown in real-time on any personal computer connected, by a serial cable, to the microcontroller. A feature of this interface is to record the real-time data into a Temperature-time plot graph. The cold junction temperature is also fully adjustable, at any time of operation, to reflect the real temperature of the surroundings. This interface is fully implemented and run by the Python programming language.

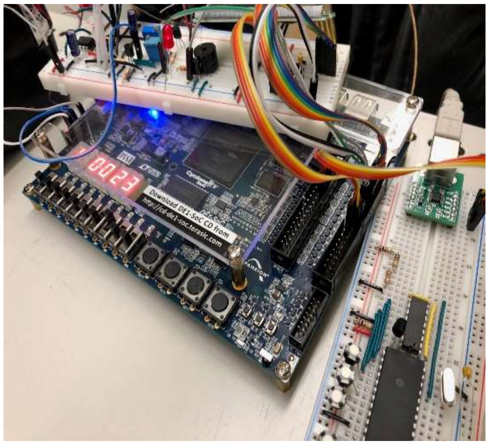

Figure 2a: Temperature Display (DE1-SoC 7-segment LED)

Figure 2b: Temperature Display (Python UI)

State Machine Functionality

As part of the design plan, a state machine was implemented to provide an intuitive way of operating the oven through the reflow process. In order to test the accuracy of the state machine, an external LCD display (LCM-S01602DTR/M) was used to display the current oven/reflow stage and total runtime by transmitting data with the microcontroller through serial cables (Figure 3). Functionality of the state machine was analyzed by ensuring that the current state complements the supposed state as decided by the temperature of the oven.

Figure 3: Oven State Display

Errors and Iterations

Due to the nature of the project, a lot of the tasks were delegated to avoid overlapping. Therefore, file versions were tracked and stored for reference during a lot of iterations throughout the project. Simple file management and sharing used to communicate iterations within the project.

Further Data Collection

- Lab tools such as the oscilloscope were used for debugging the circuit and inspect individual components of the circuit.
- Built-in and external program compilers were used to debug code files
- Additionally, cross-checking with other members and groups, as well as reference to data sheets and online databases were used to investigate inaccuracies and errors.

Synthesis of the data was done cohesively throughout its collection. Data was constantly inspected to identify errors and to ensure that it runs as expected. Graphs, tables, and diagrams were used to represent some of the data to increase ease of evaluation.

2.4 Analysis of Results

The validity of our conclusions was assessed through rigorous testing and constant evaluation of the results obtained throughout the entire project. Our circuit is tested manually by observation every test run, along with programs that automatically record, store, and display the data obtain. This data is then graphed and plotted for easier viewing and kept being re-evaluated against the design requirements and criteria to see if they improve after each iteration or testing.

With regards to temperature measurements, our team utilized a python script to compare the measurements taken by the multimeter and by the thermocouple to determine their difference, or error. These error values are then recorded at intervals in the form of a table in an excel sheet. As per the criteria, none of the error values should exceed ±3ºC. To further ease the analysis of the data, conditional formatting (Microsoft Excel feature) was implemented to mark any values that are over ±3ºC (see Appendix II). From three full trials, the data showed minimal error values that were outside the allowed constraints.

3.0 Design

3.1 Use of Process

We applied knowledge of general design processes obtained from previous lectures and past courses to design our circuit. From lab manuals, we identified needs and stakeholders, and understood the purpose and requirements of the project. We then read some slides which gave us some ideas of what our circuit should include. Afterwards, we built the circuit and tested it with example codes. Once the circuit is working, we began to write our code. We started by writing C codes as it was more straightforward and later, we converted them into assembly. To expedite our work, we divided ourselves into two groups working on the temperature measurement and on the finite state machine. We then merged two parts of code together and performed final tests. We changed parameters and baked paper sheets, trying to find the appropriate soak and reflow temperatures. During this period, Python was used to graph the temperature changes, and we wrote our own Python code for error calculation at the end. Note that datasheets were also used throughout the design process to help us configure the circuit. Lastly, multiple adaptations were made to improve the functionality of our project. (use of timers, pushbuttons, switches, etc.)

3.2 Need and Constraint Identification

Our main stakeholders are Dr. Jesús Calviño-Fraga, the course instructor, TAs of the course, engineers who will solder components onto circuit boards, and ourselves. Thus, our need is to make a reflow oven controller. The controller must have certain functionalities that enable automatic soldering and reflowing process.

Some constraints that we also identified were the specifications of the equipment and tools that were available for us, such as the power limitations of the oven, which was imperative in helping us determine the parameters of our circuit. Furthermore, other electrical components, such as the resistors, microcontrollers, and capacitors also have their own limitations. Close inspection and constant reference to the existing data sheets were necessary to ensure that their usage is in proper accordance with their recommended usage conditions.

3.3 Problem Specification

As mentioned in Section 2.2 Need and Constraint Identification, most of the design requirements had been decided by the stakeholders. We established the need to amplify the signals, stabilize the output from the ADC, and create an interactive user interface on the LCD. These displays also needed to be made as clear as possible to minimize error in its usage. The oven parameters should also be easily adjustable so that our controller can be made compatible with different kinds of ovens. In terms of safety, a safeguard was installed in the controller allowing it to automatically cut power to the oven when it encounters an unhandled exception as well as sound an alarm to warn users around the oven. This feature bypasses and will run regardless of the state the oven is operating in. Through the project, we also realized several additional functionalities that were required. We considered the fact that since the projects were being conducted in an open lab, where most engineering students in general have access to, we needed to add a feature that enables the protection of our equipment and others from the potential dangers of it being operated by individuals who do not have the experience/authorization to do so. This factor has led us to identify another design requirement: password protection.

3.4 Solution Generation

- Iterations in Design:
    - Altered the resistors' values to make the voltage gain more accurate and consistent at high temperatures.
    - Added password protection
    - Using potentiometers to match room temperature to the cold junction of the thermocouple (offset nulling)
- Failed Designs:
    - Using timers for the finite state-machine
    - A set of resistors with relatively high resistance which results in a fluctuating voltage gain and unsteady temperature readings
    - Time counter by adding up delays; Timer 2 was used instead for keeping track of the time counter, and is independent of the finite state machine
- Final Solution Specifications:
    - Timers used to count the runtime
    - EEPROM was implemented
    - Delays utilized for letting in different amount of power (0%, 20% or 100%)

3.5 Solution Evaluation

Our microcontroller design met the needs of the stakeholder using the oven-controller. Temperature measurements were ensured to be accurate, and the safety of the oven-controller is ensured through multiple testing of the safety guard feature. Safety is a core feature of APEGBC Code of Ethics guidelines[2], and we ensured that our solution is safe. The password mechanism accurately deters individuals whom do not possess the predetermined password. The UI and interface are intuitive, and the oven-controller correctly reaches desired reflow temperature parameters, as thorough testing has revealed.

3.6 Detailed Design

Finite State Machine

As illustrated above, our Finite State Machine (FSM) includes 7 stages, which are STOP (an extra stage), Reset (stage 0), Ramp to Soak (stage 1), Preheat (stage 2), Ramp to Peak (stage 3), Reflow (stage 4), and Cooling (stage 5) respectively. There is generally a beep in the transitions between stages. In addition, it also includes an auto abortion feature in case of running into exceptions to satisfy the safety criteria. Pressing KEY.2 and exceptions both lead it to STOP stage where resetting is necessary to control the oven again.

To be more specific, stage 0 is the place where we, as operators, can adjust the FSM profiles: temp_soak, time_soak, temp_refl, and time_refl. Due to the successful implementation of EEPROM, we are also able to store and retrieve those values from the memory chip accordingly.

Key.3 on DE1-SOC is pressed then released to activate the entire operation. As soon as it gets into stage 1, the clock for runtime starts to count up using the timer 2 with the oven running at 100% of power. FSM gets onto next stage when temperature reaches exactly our very first parameter-- temp_soak. As a safety feature, the microcontroller cuts of power if the oven is not heated to the acquired temperature within 60 seconds on state 1, jumping to the STOP stage.

Having arrived at stage 2, the power decreases to 20% expecting that the temperature plateaus around or slightly higher than temp_soak, so the soldering paste has time to flow back in desired place. After time_soak of seconds elapsed, stage 2 ends, and the oven gets back on its full power.

Similar to stage 1, temperature goes up dramatically on stage 3 until it hits temp_refl, again, one of our profile parameters. Staying at temp_refl for seconds of time_refl by 20% power on stage 4, temperature finally starts to drop (stage 5) from the ceiling to the ambient. We uncovered the oven to help it cool down quickly. At the end of stage 5, system reaches the safety threshold (60C) accompanied by a series of beeping from the buzzer.

Meanwhile, the FSM loops back to stage 0 where we can modify the profile and prepare for the next round of operation.

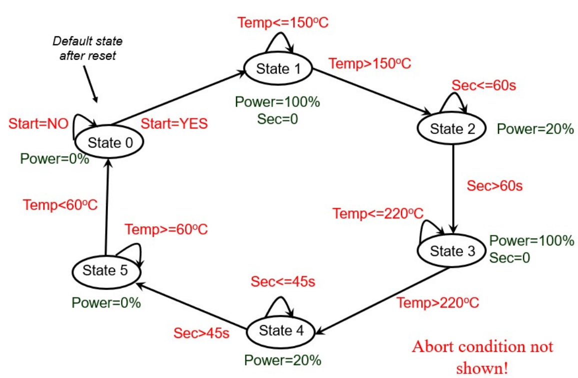

Figure 3: Oven State Display

EEPROM

Included in the package is an Electrically Erasable Programmable read only memory (EEPROM) that was implemented to store and provide the FSM profile. FT93C66, as the EEPROM we used, is initialized at the initialization stage and transmitting data with FSM at stage 0 taking inputs from DE1_SOC switches-- SW.6 for activation, SW.5 for Read/Write switching, and SW.4 for location selection.

Oven-Controller Interface

To use the Oven-Controller, the user must first turn on the DE1-SOC. After pressing the first pushbutton (sequence from left to right) on the DE1-SOC, the LED displays a reminder to enter the password, and you must enter the password through PUTTY on the computer through the Cross-IDE. Reflow parameters are then displayed following the password being confirmed, and the oven-controller is ready to be initialized. The oven controller commences the finite state machine by pushing the second push button, but if you want to alter from the default/previous reflow profile you must alter the reflow profile before commencing the finite state machine. The first switch switches between displaying both time_refl and temp_refl and both time_soak and temp_soak. Depending on what displays, the third push button changes either time_refl or time_soak, and the fourth push button changes either temp_refl or temp_soak. The third switch decides between incrementing and decrementing.

3.7 Solution Assessment

Our microcontroller design was tested extensively to ensure it met design requirements. Temperature measurements from the thermocouple were validated using a multimeter attached to the thermocouple and a python script which compared the values read from the multimeter and the values acquired by the microcontroller. The spreadsheet generated from the script confirmed that the temperatures were within the 3 degree accuracy range for all operating temperatures. The cold junction temperature measurement from the LM335 temperature sensor was verified using a mercury thermometer.

Our also team tested every scenario that our controller could encounter during operation to ensure it operated as expected. Abort conditions were checked at every stage and safety features were verified. The timer running the entire operation was checked using an external stopwatch to ensure accuracy. The plateau temperatures of the soak and reflow temperatures were verified using a python strip chart.

Finally, we ran tests using pieces of paper to optimize reflow parameters and to ensure the oven was working properly. We first placed a large sheet of paper in the oven during operation to determine hot and cold spots within the oven to determine the optimal location to place the boards to be soldered. We then tested several times to ensure that the paper came out golden, as this indicates that the boards will be soldered properly.

4.0 Life-Long Learning

In this project, our team referred to many technical concepts and skills we had gained in the prerequisite courses such as ELEC 201 Circuit Analysis and CPEN 211 Introduction to Microcomputers as well as current courses such as ELEC 221 Signals and Systems. We found ELEC 201 was of particular use when we were designing the circuit. When we moved onto designing the software for this project we used the knowledge gained in ELEC 221 which allowed us to understand the pulse width modulation signals we needed to create.

We found that this project highlighted our inexperience with finite state machines and the difficulty of creating programs that utilize sequential logic. We took this as an opportunity to build up our knowledge of the different types of state machines, implication tables and combinational logic. As a result of our online research, we were able to use these new skills to optimize our design and create a functional state machine. In consideration of safety, the system can always go back to the reset stage when running unexpectedly.

Through this process, we have gained experience in hardware design and construction, finite state machine design and optimization and circuit analysis. We feel the skills we have developed over the course of the project will be useful not only for the continuation of our education but for our future careers as electronic engineers.

5.0 Conclusion

In summary, our design controls the oven mainly in two ways based on our Assembly code: temperature controlling and time controlling. During the Ramp to Soak (stage 1) and Ramp to Peak (stage 3) stages, it keeps the oven at full power, reads the temperature from the thermocouple wire, and compares it with the parameters that we set. When the temperature reaches the Soak and Reflow temperatures, it advances to the Preheat (stage 2) and Reflow (stage 4) stages respectively where it turns on and off the oven periodically (for every second, the oven is on for 0.2 second and off for 0.8 second) to maintain steady temperatures, and time is calculated to determine when to go to the next stages. After stage 4, the oven enters Cooling (stage 5) stage where our controller simply turns off the oven. When the oven is safe to touch (i.e. below 60 degrees), it informs us and returns to stage 0 in preparation for the next operation. Our oven parameters can be adjusted by using pushbuttons, and we enable the memory to record the last parameters for frequent use. Our alarm beeps when an exception is captured and automatically shuts off the oven. In consideration of extra safety features, we have implemented password protection to prevent anyone else from using the oven.

Over the course of this project, we encountered challenges with our circuit design and software implementation. One such problem was that the resistors we chose for detecting temperature resulted in huge inaccuracy and fluctuation in readings. We resolved this issue by using lower value resistors, as we found that these gave more stable and accurate readings. We also encountered some issues with a constant offset on our final temperature reading, which we tested to ensure it was constant across all temperatures before calibrating using a potentiometer.

Moreover, during stage 2 and 4, time went faster than it should normally be, resulting in insufficient stays in soak and reflow processes. We solved this problem by switching to timer which provided us with much more accurate results.

6.0 References

[1] Smith, J, and F. Jones, “Designing an universal logic circuit”, Journal of Impossibly Wonderful Electronic Circuits, v.3, n.1, pp. 21-35, March, 1910.

[2] Jones, F and J. Smith, “Why universal logic circuits are impractical” , …
http://srdata.nist.gov/its90/download/type_k.tab
https://www.apeg.bc.ca/getmedia/e8d858f5-e175-4536-8834-34a383671c13/APEGBC-Code-of-Ethics.pdf.aspx

7.0 Bibliography

Calviño-Fraga, J., Dr. (2019, January 25). *Project 1 – Reflow Oven Controller*. Lecture presented at Electrical and Computer Engineering ELEC291/ELEC292 in University of British Columbia, Vancouver.

Department of Electrical and Computer Engineering. (2019, Winter). Project 1 – Reflow Oven Controller Manual. Vancouver, British Columbia: The University of British Columbia. Retrieved from class website at http://canvas.ubc.ca.

IPC, 2215 Sanders Road, (2001). *Guidelines for Temperature Profiling for Mass Soldering Processes (Reflow & Wave)* (IPC-7350, p. 1). Northbrook, IL: IPC.

Reflow soldering. (2018, December 11). Retrieved January 25, 2019, from http://en.wikipedia.org/wiki/Reflow_soldering

8.0 Appendices

Appendix I – Project Requirements

1. **Programmed in assembly language.** The software of the Reflow Oven Controller must be written in assembly language.
2. **Temperature Measurement.** Your reflow oven controller should be capable of measuring temperatures between 25°C and 240°C. For this purpose you MUST use a K-type thermocouple with cold junction compensation. About 2 meters of K-type thermocouple wire is included in the parts kit for project 1. Please do not cut the thermocouple wire; it needs to be long to reach the toaster oven from your circuit.
3. **Toaster Oven and Solid State Relay Box.** The Reflow Oven Controller must operate a standard of-the-shelve 1500W toaster oven using a solid state relay (SSR). Toaster ovens and SSR boxes are installed in MCLD 322 and MCLD 303 for this purpose. The controller must regulate the amount of power delivered by the oven using any method (Pulse Width Modulation (PWM), for example, seems to work well) on the control signal of to the SSR.
4. **User Interface and Feedback.** The Reflow Oven Controller must have the following functionality:
    a. Selectable reflow profile parameters such as soak temperature, soak time, reflow temperature, and reflow time. These or equivalent parameters should be selectable using pushbuttons.
    b. Display of temperature(s), running time, and reflow process current state using an LCD attached to the system.
    c. Display the oven temperature in °C using three 7-segment LED displays. If you are using a soft processor in the DE1-SoC board, you can use the 7-segment LED displays included with the board.
    d. Start/Stop pushbutton. A pushbutton will be used to start the reflow process. A switch or pushbutton will be used to stop the reflow process and turn off the toaster oven at any moment of the reflow process. The stop function must be independent of the reset button of the system.
    e. Beeper feedback. A short beep will indicate that the reflow process has been started. A short beep will indicate the transition from one state of the reflow profile to the next. A long beep will be used to indicate when to open the toaster oven door. Six intermittent beeps will be used to signal that the PCB is cool enough to handle.
    f. Temperature strip chart plot using the serial port of the system and a personal computer. The Reflow Oven Controller must send the current oven temperature, in degrees Celsius, through the serial port of a personal computer at a rate of one reading per second. Software running in the computer will read the information from the serial port and plot the temperature in real time to provide feedback about the reflow process.
5. **EFM8 boards**. Included in the project 1 parts kit for ELEC291/ELEC292 are six EFM8 PCBs and all the components required to assemble them. As part of the work for project 1 you must assemble at least three (3) of these boards by applying solder paste using the stencil included in the kit, place the SMD components using the non-magnetic tweezers and magnifying loupe also included in the kit, and solder the components using the toaster ovens, SSRs, and your Reflow Oven Controller. These boards will be used for labs 4, 5, and 6, as well as the second project of course. If you feel confident enough about your controller you can reflow one of these boards during the project demonstration.
6. **Automatic cycle termination on error**. If the reflow process is started without the thermocouple wire inside the oven, it is possible to reach temperatures high enough to burn (as in catch fire!) any PCB placed inside the oven. As a safety measure, the reflow process must be aborted if the oven doesn't reach at least 50°C in the first 60 seconds of operation.
7. **Temperature validation data**. Use the lab multimeters to validate your controller temperature measurements. The maximum acceptable temperature error of your controller is ±3°C for the range 25°C to 240°C. You must include the procedure and the data in the project report. This data must be available during the project demonstration.

Picture taken lab manual given to the groups prior to starting the project.
(Dr. Jesús Calviño-Fraga, 2019)

Appendix II – Error Tables

Temperature													
20	-2.2	-2.2	-2.2	-2.3	-3	-2.9	-2.2	-2.2	-2.1	-2.1	-2	-2	-2
30	-2.4	-1	0.8	1.6	0.1	0.4	1.2	0.8	0.4	0	0.6	0.2	
40	-2.7	-0.4	2.1	2	1	1.1	2.2						
50	-2.7	1.8	2.4	3.1	2.1	1.2	0.2	1.2					
60	-1.2	2.3											
70	-1.5	-1.9	1.4	1.2	2	0.3	0.6						
80	-1.5	-1.2	0.9	1.2	0.7	0.1	-1.3						
90	-1.5	-1.4	-1.6	-2.2	1.9	1.3	1.5	0.7					
100	-1.4	-1	-0.4	2.1	0.4	-2.3	-2.5	-1	-0.1				
110	-1.5	-1.3	-1.5	-1.5	-1.2	0.5	2.5						
120	-1.8	-0.8	-2.4	-1.3	-0.7	-0.6	-2.4	-1.8	-0.9	1.7			
130	-1.9	0.9	1.3	1.5	1.9	1.2	0.5	0.3	0.2				
140	-1.2	1.2	1.2	2.6	2.5	0.2	-1.2						
150	-1	-0.2	-2	-2.3	-1.2								
160	-0.7	-1.2	-0.7										
170	0.2	2.9	-0.7	1.5	2.7								
180	-0.7	-0.6	-2.3	-1.3	1.4	0.7	0.2						
190	-0.8	-0.8	1.8	1.2	0.4								
200	2	2.1	-1.3										
210	-1.4												
220													

The colorized error table above shows the data taken during the first oven reflow test. There were very significantly less values due to the strict criteria that the values had to be checked before undergoing error evaluation.

Temperature															
20	-1	-1	-1.1	-1.1	-1.2	-1.2	-1.2	-1.2	-1.2	-1.2	-1.2	-1.2	-1.2	-1.2	-1.2
30	0.1	-1.5	-0.1	-0.5	-2.1	-1.3	-1.8	-0.5	0.7	-1.4	1.1	-1.4	0.3	0.1	-0.1
40	-1.6	-2.9	-1.7	-1.6	-1.4	-0.4	-0.2	0.3	0.4	0.7	-0.4	-0.2	0.2	0.1	-0.8
50	-2.6	-2.3	-2.5	-2.9	-0.4	-1.4	-0.4	-0.2	-0.1	0.4	0.5	0.7	0.1	-0.2	-0.8
60	-2.8	-2.9	-2.5	-2.4	-2.1	-1.2	-1.4	0.1	-1.4	-0.1	-0.1	-0.4	-1.4	-0.8	1.1
70	-2.9	-2.8	-2.2	0	0.1	0	-0.3	-0.4	-0.6	-0.2	-0.2	0.1	0.2	-0.1	-0.2
80	-2.1	-1.9	-2.2	0	-0.1	-0.2	-0.3	-0.3	-0.4	-0.4	0.5	-0.7	-0.8	-0.9	0
90	-2.6	-2.1	-2.4	-0.1	-0.2	-0.4	0.5	0.4	-0.7	0.1	-0.1	-0.2	-0.3	-0.4	0.5
100	-3	-2.3	-2.3	-0.1	-0.3	-1.4	-1.4	-0.5	-0.6	0.4	-0.7	0	-0.2	-0.4	-0.6
110	-3	-2.3	-2.3	0	-0.1	-0.2	-0.3	-0.5	-0.6	0.2	0	-0.2	-1.4	-0.5	-0.6
120	-2.2	-2.6	-2.6	-1	-0.2	-0.3	-0.6	0.2	-0.9	-0.1	-0.3	-0.5	-0.6	0.2	0.1
130	-0.9	-1.8	-1.2	0	-0.2	-0.4	0.4	0.2	0.1	-0.1	-0.2	0.6	-0.6	-0.7	0.1
140	-0.9	-1.9	-1.4	-1.4	-0.1	0.6	0.3	0.1	0.7	-0.4	0.2	0.9	0.7	0.4	0.2
150	-1.2	-0.8	-1.5	0.8	0.6	0.2	1	0.5	0.4	0.2	-0.1	0.7	0.4	0.2	-0.1
160	-0.6	0.3	-0.1	-0.4	1	0.6	1	0.8	1.6	1.5	1.2	1.1	1.8	1.6	0.4
170	-0.4	-0.9	0	-1.2	-0.6	-0.2	0.6	-0.8	-0.1	0.5	0.1	-0.4	-0.7	-1.2	-0.5
180	-0.5	-0.9	-0.9	-0.9	-0.3	1	0.5	0.1	0.6	0.1	-0.3	0.1	0.8	0.3	
190	-0.6	-0.6	0	-1	0.8	0.2	0.7	0.1	0.8	0.3	1	1.4	0		
200	-0.3	0.3	-0.6	-0.9	0.8	1.4	0.8	1.3	0.7	0.3	1.7	0			
210	1	-0.5	0.4	0	0.8	0.3	2.8	1.3	1.9	1.4	2	0.4	1		
220	0.7	0.2	0	-0.3	1.6	0.9	1.5	1.3	1.9	1.6	1.1	1.8	1.5	1.1	1.8

The colorized error table above shows the values taken for the second trial of fully operating the reflow oven

Both tables were fully printed out by a python script which simultaneously read and computed the values transmitted by lab multimeter and the thermocouple through the serial port.

Appendix III – Source Code

The complete source code can be viewed on gist and github.
We have distributed the code into different parts:

- Assembly Controller
  https://gist.github.com/ad2969/b151c8feb8e94b5c0679a81dc4b5b404

- Python Visualizations
  - Strip-chart
    https://github.com/ad2969/at89-python-temperature-validation/blob/3f76fb16f86859b29f766ddc274934fedcfe9c87/newest-stripchart.py
  - Error Validation
    https://github.com/ad2969/at89-python-temperature-validation/blob/d0fe928d25436034425c50cef3e4ef0b6edc53a7/main-validation%20-%20lenient.py

Appendix IV – Video Demo

A video recording a demo of the oven at work can be found at this link
https://youtu.be/HV-fJrIxkjY