Embedded IOT Controller

Background and Proposal

Our project was to create a control system for an injection molding machine made by Minufacture. The current machine uses two separate temperature control systems that must be set manually. This caused user to have to know the melting point of their plastic and set each value individually. Our project will allow users to be able to choose the type of plastic they are using via a display. Based on this choice, the set points for both heaters will be set. They will also be able to create and edit these profiles. We will use a PID loop to keep the temperature within 10 degrees of the set point.

Design Requirements

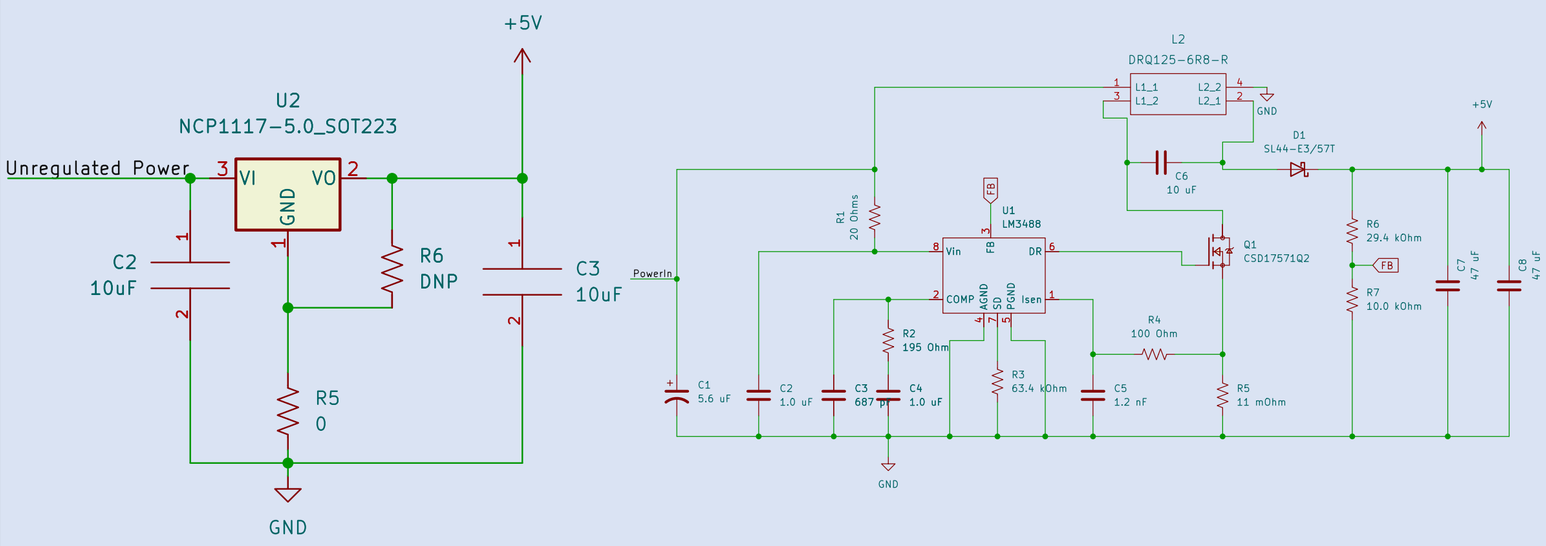
- Have a single UI that can control both heating elements
- Pull temperature from a database based on type of plastic
- Edit and create new profiles
- Keep temperature within 10 degrees of intended set point
- Cost $250 or less
- Ability to be open sourced with room for expansion
- Keep power to Beaglebone until shutdown is complete

Hardware Solution

Our solution is a PCB cape for a Beaglebone black. The PCB includes two Adafruit thermocouple circuits, battery charging circuit, power supply switching, and two mechanical relays to control the heating elements. We also designed a custom daughter board containing a 5V buck boost converter.

Hardware Design Evolution

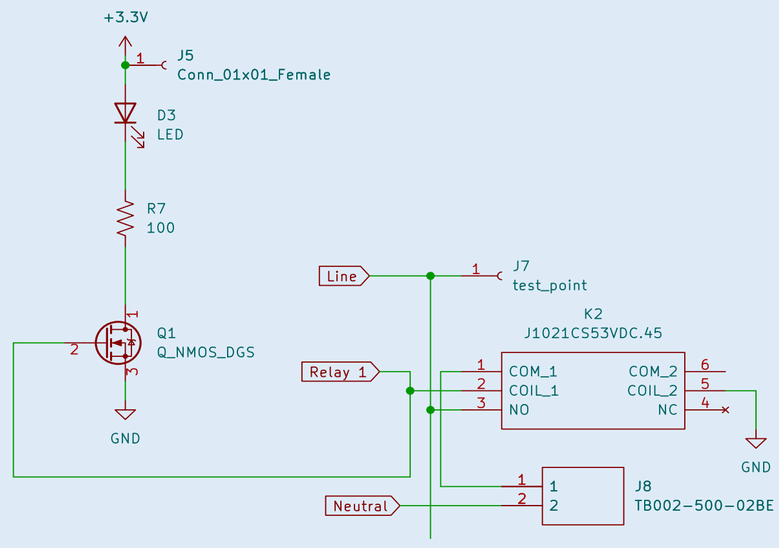
5 Volt Regulation



Our first implementation of the 5V regulator was an LDO shown above in conjunction with the first rev of the gentle shutdown circuit. It operated on the 15V rail.

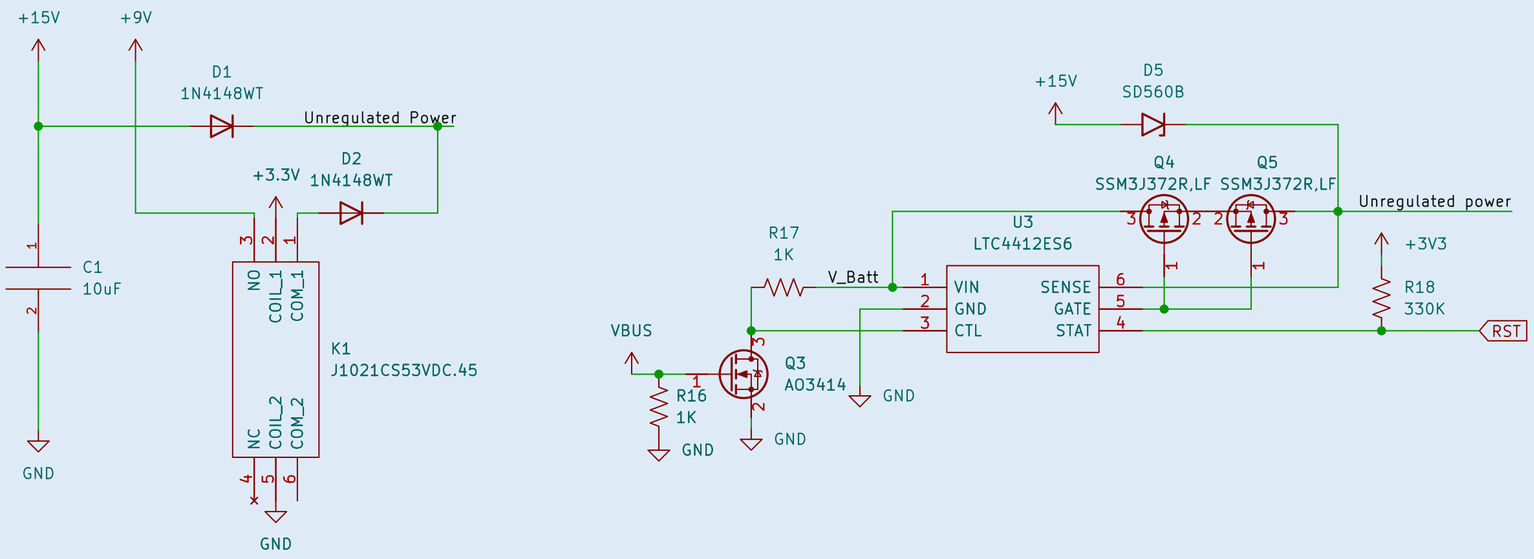
As our implementation of the gentle shutdown functionality evolved to include a battery, we moved to a 5V switching regulator on a daughter board to boost the battery voltage.

Relay Control



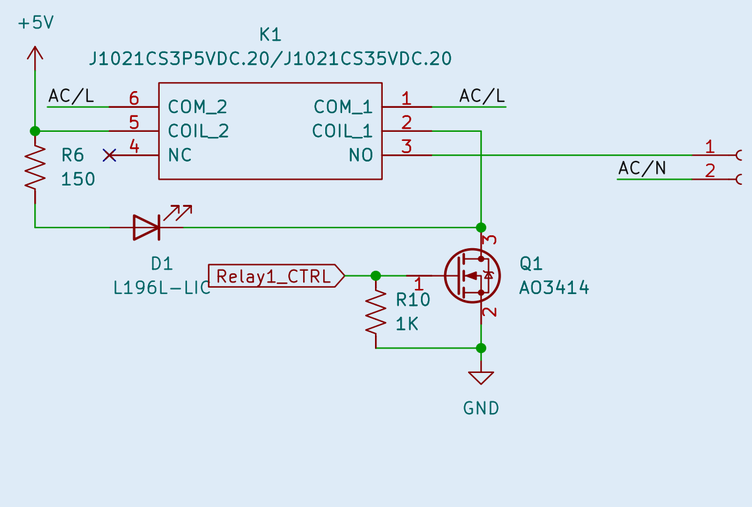
We need relays to regulate the heating element for the injection molder. In our first rev, we only used the µController to drive the relays, but this didn't work out, nor did our initial choice of relay parts.

Gentle Shutdown



If 5V rail drops from the BBB after shutdown begins, the OS will corrupt. This was our first attempt fixing this using a 9V battery which discharged into the 5V regulator if the 15V rail was shut off.

The final revision utilizes a LIPO battery charging circuit (not pictured) in tandem with the 5V regulator to provide the BBB with 5V when the 15V rail is down. We use MOSFETs to alternate where the BBB sources 5V.



Our current implementation of the relay sub system involves use of MOSFETS to provide the relays with the current they need for operation.

Software Solution

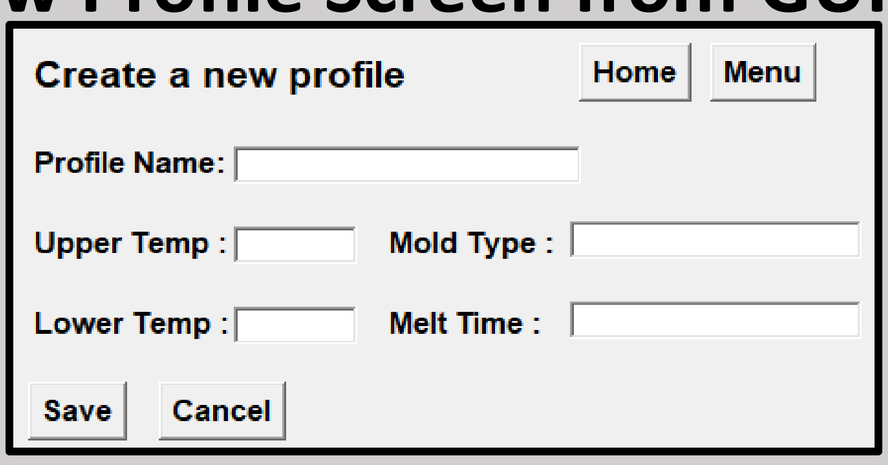
Final List of Implemented Features

- Device Tree Overlay to set up display
- GUI using the python tkinter library
- A python command line interface to support GUI functions
- SQL database to store profiles, PID values, and history
- GPIO configuration to read thermocouple temperature and turn on relays if needed

Security Concerns

- Exposed USB ports needed for mouse and keyboard
- During initial setup must be connected to internet

New Profile Screen from GUI



IEEE Standards

IEEE/ISO/IEC 29119-2-2021 These standards informed how we tested our software.

IEEE 15939-2008 These standards informed how we collected and used sensor data.

Lessons Learned

- Do more extensive parts research at beginning of project
- Order more parts than you think you will need
- Triple check PCB footprints for size and orientation
- Switching regulators are very layout dependent
- Include lots of easily accessible test points.

Advisor: Dr. Jones

Client: Mark Hansen

Group Members: Rachel Teberg, Stone Widder, Josh Baringer, Evan Pasero, Charles Sang