Boiler Control Project

Nick Piscitello, Joe Piscitello: 2014 - 15

Design Specifications

What is the device's desired operation?

- If the stove is above a user-defined threshold, turn the circulators on for a user-defined amount of time, then rest for a user-defined amount of time. Also, if the stove is hot, operate on the I/O device (encoder with button, 3 digit LCD, 4 indicator LEDs).

What are the device's main tasks?

- I/O: more of a function library than a true task - presents inputs when requested by meta-management (encoder value, thermistor value, etc), then outputs data to the display when told to by meta-management
- meta-management: adjust and monitor applicable variables (stove temperature, temperature threshold, circulate time, delay time), keep track of periodic tasks (temperature measurement, I/O output)
- circulator control: cycle the relay controlling the circulator loops

Software design:

- 3 modules, each pertaining to a main task (I/O, meta-management, circulator control)
  - each module is completely separate from the others, plug-n'-play
    - write each as a library? Read up on C++ classing...
  - there should be no code for any module in the loop() function

Pseudocode - *all code assumes stove temp is over threshold*

Module 1: I/O

control

- listen for a start/stop command from meta-management
- low-power listener mode if stopped

input

- deals with dirty work of reading the encoder and the thermistor
- processes and presents an array (or matrix) of values to meta-management

output

- will receive an array (or matrix) of the 4 variables and the index variable
- reads matrix, displays the appropriate information
- WILL NOT handle updating the variables or index; that's for meta-management

Module 2: meta-management

acquisition

- ask I/O module for relative encoder reading and thermistor reading

processing

- test for thresholds
- update variables

export

- package and send display data to I/O module
- package and send timing data to circulator control module

Module 3: circulator control

reciept

- will listen for an array of timing data (int[circ, rest] for example) and a start/stop boolean value from meta-management

control

- operate with any received timing data if the start bit is set; run a very low-power listener mode if stopped