Steven Gassert

Embedded Linux

Final Report

1.Project description

Home brewers around the country have struggled for years with the challenge of keeping their fermenting beer at suitable temperatures through out the hot summers and cold winters. With the dawn of micro computing and affordable technology some hobbyist's have taken to using the raspberry pi to automate the control of heating and cooling elements. In this project I used my raspberry pi to turn a refrigerator and heat fan on and off to keep my wort within 1 degree of the temperature I set. I also used a temperature sensor to collect data and an Infrared Beam Break sensor to alert me when my beer was complete and create some surprisingly not so bad tasting beer.

2.Project Goals

A man by the name of Manuel Fritsch contributed to git hub some really nice software which controls the heating elements and even provides a dashboard which the user can view much of the information on his/her currently brewing beer. My goal in this project includes implementing Fritch's project plans, adding another element which collects data, and then creating my own dashboard which will provide additional information that is also helpful to the home brewer. The other element to which I am referring is a bubble counter. Before beer turns into beer, yeast sits in a bucket with dissolved sugars. Over the course of approximately 4 weeks the yeast turns the sugar into gas and alcohol. Home brewers place a “bubble” lock on top of their fermentation bucket which lets extra gas out without letting outside air in. as the bucket releases gas the air lock bubbles. As the beer approaches it's completion the bucket will release less and less bubbles and the home brewer knows that it is safe to package the beer and eventually consume because not as many bubbles are visibly passing through the air lock. My alternate dashboard will display three things that Manuel Fritshch does not: bubbles seen over the past hour, bubbles seen over the past 24 hours, highest temperature recorded over the past 24 hours, and lowest temperature recorded over the past hour.

3.Implementation
Hardware:

Exhaustive charts on the hardware wiring can be found on http://web.craftbeerpi.com/hardware

a short summary:

SSR relays are used to turn both a refrigerator and a heat fan on and off the SSR relay is connected to an extension cord. The power to the SSR relay which acts as a switch when the extension cord is plugged is wired to a GPIO pin on the raspberry pi. When the pi provides power to the GPIO pin it effectively turns on anything that is plugged into the extension cord on and visa versa when no power is being supplied through the GPIO.

To detect weather or not a bubble is passing through the air lock I used an adafruit infra red beam break sensor. The IR beam break sensor contained two parts, the transmitter and the receiver. The transmitter was wired to 5v power and a ground while the reciever was wired to a 3v power, ground, and a GPIO pin. A temperature sensor also was used and wired to 3v power, ground, and GPIO pin. Both the temperature sensor and the IR receiver used pull up resistors wired between the power and data wire.

Software:

As stated above the software used to turn on and off the fride and heat fan was cloned from Github at https://github.com/manuel83/craftbeerpi. As far as my own dashboard I used an appache server to serve web pages over the LAN. All the software that is necessary to run my implementation is found in /home/pi. I use a start up script startBubbleCounter.sh to run bubbleCounter.py on start up. Once bubbleCounter.py is running any time the beam break sensors are pointing at each other a bubble will be recorded in the bubbles table of mydatabase.db. In addition to using the Infra red Beam break to collect data about the beer a temperature sensor was also used. A cron tab is used to collect temperature and log it to the temps table in mydatabast.db every minuet. When the pi is used the user should request the IP-ADRESS-OF-PI/cgi-enabled/index.py file. This will call and execute index.py in the var/www/html/cgi-enabled folder. At the beginning of index.py you will find two functions defined first:. isWithin24(datetime) and isWithin1(datetime). These functions were necessary because when index.py goes to display the data to the user it fetches all entries in the bubbles and temps table and checks each date and time to see if that entry was recorded in the last 24 hours and 1 hour respectively. Some parsing was required in these two functionc because in the temps and bubbles tables of mydatabase.db the time.now() built in function was used to record temperatures and bubble sightings while I needed to get these dates and times into datetime.datetime format to compare to the current time and see if it was recorded within the last day or hour Next the getTemps() and getHighsAndLows() searches through the temperatures to display the highest recorded temperature and lowest recorded temperature within the past hour and day. Finally this information is formatted and displayed to the user.

4.Components:

I completed the hardware wiring first as this was the most daunting to me. Once the ssr relays were tested to be successful I worked on installing the craftbeerpi software and getting it to recognize the GPIO pins that I had wired my heat fan, fridge, and temperature sensor to. While the beer was fermenting I worked on creating the bubble counter and then finally the web server to display the information

5.Project evaluation:

Once of the more difficult aspects of this project included getting the SSR relays to work properly. Since SSR relays work with main voltage, it is dangerous to do testing if you are not sure of what you are doing, because of this much research was done at every step of the way during this aspect of the project. I also housed the SSR relays in plastic boxes attached to the side of the fridge to ensure safety. Another aspect I had never worked on and had to do a gair amount of research on during this project was setting up an apache server. Particularly challenging to me was getting the server to execute .cgi and .py scripts rather than just displaying the code. Another part of the project that gave me a fair bit of trouble was parsing the current time and the time stored in the database because I had used two different functions to store the time. On the other hand, one thing that was surprisingly easy was getting the temperature sensor and other elements hooked up to the craftbeerpi software as Fritshch's software is very intuitive.

6.Contributions:

I was the only one working on this project

7.References:

http://web.craftbeerpi.com/
https://github.com/manuel83/craftbeerpi
https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md
http://www.public.coe.edu/departments/Nursing/psychomotorskills/calculatingflowrateformula.htm
https://www.allaboutcircuits.com/projects/build-a-raspberry-pi-pushbutton-switch/
various stackover