AUTOBREW:

Automated All Grain Homebrewing System

Critical Design Review

Nick Anderson

10/5/15

The Autobrew is meant to be an automated system that will perform the majority of the tasks involved with brewing with all grains (method of brewing where grains are used instead of purchasing an extract that contains all the components created after the malting stage).

The Autobrew will run the entire brewing process, only requiring human interaction to add grains to the mash tun, adding hops when required, moving beer from fermenter to bottles/kegs, removing spent grains from mash tun, and removing the leftover trub (solids remaining in the brewpot). The entire system will consist of multiple vessels (described later) required for homebrewing, a microcontroller, to handle the sequence of events with the physical vessels, a remote touchscreen for inputting times and temperatures for brewing, and a microprocessor which will function as the server for the information shown on the touchscreen and communicating that information to the micro controller.



Figure 1: Top Level Block Diagram

The functional sequence of events from the user’s standpoint will be to start the setup program on the remote touchscreen. This will open a window that will allow the user to select a variety of temperature setpoints and time durations:

* Mash Temperature
* Sparge Temperature
* Cool Wort Temperature
* Fermentation Temperature
* Mash Time
* Brew Time
* Fermentation Time

Once the user has input this information he will select to begin the process. At this point, the system will instruct the user to put the grains into the mash tun. When the user has done this, he will select it and the system will proceed through the homebrew process.

* Heat up water to the appropriate temperature to Mash the grains
* Keep the Mash at a specific temperature for a specific amount of time
* Move the liquid created during the Mash into the Brew Pot
* Start to boil the liquid when it is all in the Brew Pot for a certain amount of time
* When the boil time (brew time) is done, it will cool down the resultant wort to a specific temperature
* When temperature has been reached, it will move the wort into a fermentation tank where it will control the temperature for a set amount of time.
* When this time is complete, the user will have to remove the beer from fermentation and move into bottles or kegs.



Figure 2: Automation Module

**Automation Module**

The Automation Module consists of the microcontroller that will interface with the physical vessels required to do the home brewing. The first step in selecting the appropriate parts was to determine all of the I/O that would be required. Figure 3 shows the complete setup of brewing equipment along with sensors.

Knowing the equipment needed next was component selection. See table 1 for part number of I/O components.

|  |  |  |
| --- | --- | --- |
| pin # | Description | Input/output |
| 1 | Solenoid Valve | Output |
| 2 | Solenoid Valve | Output |
| 3 | Solenoid Valve | Output |
| 4 | Solenoid Valve | Output |
| 5 | Solenoid Valve | Output |
| 6 | Solenoid Valve | Output |
| 7 | Solenoid Valve | Output |
| 8 | Solenoid Valve | Output |
| 9 | Solenoid Valve | Output |
| 10 | Solenoid Valve | Output |
| 11 | Solenoid Valve | Output |
| 12 | Solenoid Valve | Output |
| 13 | Solenoid Valve | Output |
| 14 | Solenoid Valve | Output |
| 15 | Solenoid Valve | Output |
| 16 | Solenoid Valve | Output |
| 17 | HLT Heater | Output |
| 18 | BrewPot Heater | Output |
| 19 | Fermenter Relay | Output |
| 20 | HLT Level SW | Input |
| 21 | BrewPot Level SW | Input |
| 22 | HLT Output Flow Sensor | Input |
| 23 | BrewPot Output Flow Sensor | Input |
| 24 | HLT Temp Sensor | Input |
| 25 | Mash Tun Temp Sensor | Input |
| 26 | Brew Pot Temp Sensor | Input |
| 27 | Fermenter Temp Sensor | Input |
| 28 | Pump Power | Output |



Figure 3: Brewing System - Hardware

**Solenoid Valve –** These control flow into and out of all the vessels in Figure 3. ½” 2-way valve. Activates with 1.2A @ 12VDC. Requires interface control relay. Can be gravity fed or pressurized.

**Heater Outputs -** Heating Elements embedded into vessels. They are 1650W @ 120VAC. This will keep the current under 15A, which is important for source breakers and for management.

Will use the PowerSwitch Tail II power switching device to manage the heating elements. This can accept 5VDC @ <20mA for activation (3V requires 3mA and 12V requires 30mA). Can be connected directly to many different uC/uP (Beagle Bone, Arduino, Raspberry Pi, etc.).

**Fermenter Relay –** Fermenter will be an old mini fridge. In order to control the fridge, a Power Switch Tail II power switching device will be used.

**Level Switch** – The level switches used are food grade level switches from McMaster Carr. They can be setup to be normally open or normally closed. They act as a contact closure, so no interface is required. The 5VDC supply will be used as the common, with the NO side going into an input pin.

**Flow Sensor** – The flow sensor selected has a digital pulsing output depending on the amount of flow. This is normally used to measure amount of flow, but in my case I will be using it to measure if there is flow at all or not. I will take a snapshot over a time interval and look at the amount of pulses received in that time to determine if flow has stopped. This functions on a 5VDC supply and Ground connections. The pulse pin can be tied directly to an Input pin.

**Temp Sensor** – The temperature sensor used is the DS18B20. It functions on a 5VDC power supply and ground connections. Its output is a 1-wire interface that has libraries to work with many uC. It is designed such that a single bus can have multiple drops. The physical sensor is waterproofed by the manufacturer, so it can be immersed in liquids.

With the sensors selected, the uC used in this module was selected. The Arduino Mega was selected as it would provide the amount of I/O pins without needing any additional I/O cards. It also allowed for room for expansion, and it will accept direct connections to many of the components selected and provide the power to the components.

[I-O List for CDR.xlsx](file:///C:\Personal\Personal\JHU\Embedded%20Project\I-O%20List%20for%20CDR.xlsx)

Looking at the requirements in the linked spreadsheet, there are a couple issues.

**Power** – The intention is to plug the module into a standard US wall socket. This means we are restricted to a 15A circuit. This means that the following restrictions must be maintained:

* If heating element is active, no other relays can be activated
* Maximum of 10 solenoid relays can be active at one time.
* General purpose relays will be used to interface between uC and solenoid relays.

The software for the Automation Module will rely on the use of state machines. Within each state machine, there will be function to control the flow of the program. Wait loops will be utilized often to wait for certain events to happen and trigger the next state. Each state will initiate a reset of all outputs before starting the functions of that specific state.



Figure 4: User Interface Module

**User Interface Module**

The user interface module will be a touchscreen interface, which will provide the user with a screen of items to monitor, and input blocks to configure the brewing process. All input values must be entered before the brew process starts. Once the brewing process starts, the screen will display temperature and stage statuses of the brewing process.

**Microprocessor** – The uP used is a Raspberry Pi. It was selected because of its ability to interface with numerous different screen options. Requires 5VDC power. Many options for providing power. Will use adapter. Optional upgrades include providing battery power to allow User Interface Module to be transportable.

**Touch Screen** – The touch screen will have a USB and USB controller for the touch screen portion of the screen. The video will be controlled by an HDMI+VGA interface controller board.

The user interface is a straightforward design, electrically. The touchscreen was designed to interface directly to the Raspberry Pi. The software will require the most amount of work.



Figure 5: Wireless Network Comms

**Wireless Network Comms**

The intent of the design is to allow the user to control and monitor the brew process remotely. In order to do this, both the Automation Module and the User Interface Module must have the capability of transmitting and receiving information.

**Automation Module** – There will be an Arduino WIFI shield attached to the Arduino. This will require WIFI libraries to be included in the program. They were added in ahead of time to be aware of the impact in the software build.

**User Interface Module –** There will be a Raspberry Pi WIFI dongle added to one of the USB ports on the Raspberry Pi B+ board.

The actual Ethernet communication between the two modules via a wireless network will require leveraging resources locally and via the internet. The hardware is direct interfaced with the Arduino/Raspberry Pi.

**Schedule**

The schedule is shown below. The big time frames are time spent for software production and testing. Once the software is complete, the hardware will be put together and interfaced (wiring) to the software. The only area currently behind schedule is programming for the Raspberry Pi. This is partly due to a delay in acquiring parts, but that should be addressed now and going forward.

**Conclusion**

All of the parts have been selected and the final design has been determined. The schedule shows the tasks that need to be done, and the estimated time frame to aim for. All things point to an on time production. There is also some slack built in, to accommodate any unforeseen issues.



Figure 6: Schedule