Pool Control System Proposal



by Unknown Author is licensed under

Team BST – Members:

Taft Barrott

Brenton Dube

Steven Rioux

**Team Profile:**

All members of this team bring unique values to the table that will set this project up for success. Steve has many year background in hardware design and industrial automation and controls. His most recent project was the PLC system design and programming for the Fast and Furious Ride at Universal Studios Florida. Steve also has many years of project management that should prove useful for this project. One of his passions is home automation.

Brenton has experience with Java and Python projects, and has been learning about the broad topics that are involved in making web applications. He is open to learning and using previously unknown technologies to make a project work successfully, so is flexible on the range of responsibilities needed to contribute well to accomplishing this team project. His preference of focus is in back-end web development and general application software.

Taft has worked with Java and C++. He is also willing to learn about the different parts of web and app interaction with hardware along with all different aspects of the project. He has also done some networking which may help with web communication.

**Proposed Project Description:**

Most home pool systems are made up of several controls, one for each main function of the filtering system. This Pool Control System (PCS) will combine all needed control functions into 1 control system. This system will allow for local and remote access based on a touch screen and app or web page browser. Through the interface the user will be able to manual control each sub system of the PCS to manual change states for testing and maintenance purpose. The system will also be able to function in a full automatic mode to allow for automated control.

**Customer Application:**

The PCS is designed to be easily installed by any home owner, to have a central control of a pool system. This will allow any home user to easily tell the status of there pool system and manually control all elements of the system and full automatic mode, with variables set by the customer.

**Design Considerations:**

**Typical Pool Configuration:**

The configuration for each pool can differ, for this system the following will be considered standard equipment:

* Pump – requires 240 volt contactor to apply power to motor when filter is required
* Pressure sensor -Analog signal to detect the pressure of the system, mainly used for diagnostics
* Pool Water Temp Sensor – This is an analog signal to detect the water temperature
* Roof Top Temp Sensor – This is an analog signal to detect the solar panel temp to determine if the solar heating system should turn on.
* Valve for heater – This is to redirect the water to the solar panels.
* Digital output – This is to control the standalone heat pump or heater.

Since Each pool can be configured differently, each sensor is optional, and functionality can be enabled/ disabled from the GUI.

A Raspberry Pi will be the main controller for the system. A secondary control circuit may be used to integrate from control to logic voltages if needed.

At the base level the PCS will turn the pump on and off at a user defined time. This device also could have the capability to be set based of the local weather if the PI is online, it should be turned on 1 hour after dawn and run for 1 hour for each 10 degrees of high weather temp. So if the High for a day is 90 the pump should run for 9 hours. If heat is selected the PCS will direct the water flow to the solar panels if the water is below a certain temp (set by the user) and the roof is above a certain temp ( set by setpoint, 30 degrees hotter then the current pool temp). It will also call for heat from the heat pump if a heat pump is part of the system. The pressure sensor shall be used to send alert of possible clogged filter or other diagnostics.

**Functionality:**

The PCS shall have the following functionality:

* Easy to install
* Low Cost
* Open Source Programming environment, so end user can modify if needed to add additional sensors and devices.
* Each component shall be able to be disabled individually and allow the rest of the system to continue to operate. For example, if a pool does not have a solar heater that functionality should be able to be disabled.
* An user interface will be created to interface to the controller, this will be either web based or an app.
* User defined system
* User defined Max / Min pressure ( will shut the pump down if pressure is too high or low)

**Preliminary Parts List:**

* Raspberry Pi 3 B
* I2C 4 channel ADC
* 8 channel Relay output card
* GPIO break out module.

**Software Approach:**

The Raspberry Pi will be programmed using C++ and will be the main frame of the system. Each Read / Write operation shall be its own function so that the system can be programmed modularly, by calling a function to get the data need or to write to the appropriate output.

The User interface will be a web site that displays all of the component information and functionality. The customizable aspects of each component will be displayed and able to be changed here. This web site is intended to be written as a single-page application, possibly using the React.js framework. It will be hosted either on the Raspberry Pi or on a separate web host. If possible, it might be ported to a mobile phone application as well.

The backbone will connect the web page server to the system functionality on the Raspberry Pi. The backbone will connect each action on the user interface web site to an underlying function in the Pi system code. The way this will be done is to be determined. The C++ Pi system code might contain an API for the user interface web server to communicate with, or it might directly run a server within the application.

**Plan of Work:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Week Number | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| Design base project |  | |  |  |  |  |  |  |  |  |  |  |  |
| layout tasks |  | |  |  |  |  |  |  |  |  |  |  |  |
| assign tasks |  | |  |  |  |  |  |  |  |  |  |  |  |
| Hard ware design |  |  | |  |  |  |  |  |  |  |  |  |  |
| Functional design |  |  | |  |  |  |  |  |  |  |  |  |  |
| finalize tasks |  |  | |  |  |  |  |  |  |  |  |  |  |
| Finalize system requirements |  |  | |  |  |  |  |  |  |  |  |  |  |
| Complete design Specs |  | | | | | | | |  |  |  |  |  |
| Develop User Interface |  |  |  | | | | | | | | |  |  |
| Develop Code |  |  |  | | | | | | | | |  |  |
| Scope of Work |  | | | | | |  |  |  |  |  |  |  |
| Final Design documentation |  |  |  |  |  |  |  |  |  |  |  |  | |