Pressure Controller Module Documentation

Last modification: September 26, 2022

Version 1.0

# Sections

## Operation

## Arduino PINOUT

## Connector PINOUTS

## Code Documentation

# Overview

The Pressure Controller Module is designed to automatically regulate the pressure of sulfur hexafluoride gas within the spark gaps on the MARX generator used at the Nevada Terawatt Facility. The system will automatically adjust the pressure in selected gas circuits and operate the reclaimer as needed. The operator is able to use preset settings for the system, toggle individual gas circuits, adjust individual gas circuits to a desired pressure, purge individual gas circuits, and control the reclaimer. The operator is able to override the automatic control and operate the system manually if it is required.

# Operation

This section contains information about the interface functions and LCD menu navigation.

* **INTERFACE**
  + **POWER Switch**: The switch labeled “POWER” is used to switch the unit on or off.
  + **USB Port**: The USB port is used to connect the Arduino 2560 to an external computer.
  + **LCD and Button Membrane**:
    - The LCD displays the current state of the pressure controller and the settings menu to the operator.
    - The button membrane to the right of the LCD is used to navigate the settings menu.
      * UP ARROW/INC button is used to navigate up or increment values.
      * DOWN ARROW/DEC button is used to navigate down or decrement values.
      * LEFT ARROW/BEFORE button is used to return to the previous menu or shift the cursor to the left.
      * RIGHT ARROW/NEXT is used to navigate to the next menu or shift the cursor to the right
      * SEL button is used to select a menu option or to confirm a change in the settings.
  + **CIRCUITS**
    - **MARX ENABLE**:
      * The digital display indicates the current pressure in PSI for the MARX circuit.
      * The **IN** LED indicates the state of the input solenoid for the MARX circuit.
      * The **OUT** LED indicates the state of the output solenoid for the MARX circuit.
      * The “MARX ENABLE” button is used to enable or disable the MARX circuit.
    - **MTG ENABLE**:
      * The digital display indicates the current pressure in PSI for the MTG circuit.
      * The **IN** LED indicates the state of the input solenoid for the MTG circuit.
      * The **OUT** LED indicates the state of the output solenoid for the MTG circuit.
      * The “MTG ENABLE” button is used to enable or disable the MTG circuit.
    - **R. SWITCH ENABLE**:
      * The digital display indicates the current pressure in PSI for the R. SWITCH circuit.
      * The **IN** LED indicates the state of the input solenoid for the R. SWITCH circuit.
      * The **OUT** LED indicates the state of the output solenoid for the R. SWITCH circuit.
      * The “R. SWITCH ENABLE” button is used to enable or disable the R. SWITCH circuit.
    - **MARX TG70 ENABLE**:
      * The digital display indicates the current pressure in PSI for the MARX TG70 circuit.
      * The **IN** LED indicates the state of the input solenoid for the MARX TG70 circuit.
      * The **OUT** LED indicates the state of the output solenoid for the MARX TG70 circuit.
      * The “MARX TG70 ENABLE” button is used to enable or disable the MARX TG70 circuit.
    - **R. SWITCH TG70 ENABLE**:
      * The digital display indicates the current pressure in PSI for the R. SWITCH TG70 circuit.
      * The **IN** LED indicates the state of the input solenoid for the R. SWITCH TG70 circuit.
      * The **OUT** LED indicates the state of the output solenoid for the R. SWITCH TG70 circuit.
      * The “R. SWITCH TG70 ENABLE” button is used to enable or disable the R. SWITCH TG70 circuit.
* **RETURN PSI**: The digital display indicates the current pressure in PSI for the reclaimer.
* **SUPPLY PSI**: The digital display indicates the current pressure of the supply bottle**.**
* **START Button**: The green button labeled “START” is used to turn on the reclaimer in Manual Mode. This button has a light which will blink on and off when the reclaimer is in Automatic Mode. The button cannot be used in Automatic Mode.
* **STOP Button:** The red button labeled “STOP” is used to turn off the reclaimer in Manual Mode. This button has a light which will blink on and off when the reclaimer is in Automatic Mode. The button cannot be used in Automatic Mode.
* **RECLAIM. AUTO**: The button labeled “RECLAIM. AUTO” is used to toggle the Automatic Mode for the reclaimer on or off. When the Automatic Mode is disabled, the operator assumes manual control over the reclaimer.
* **PURGE Button**: The button labeled “PURGE” is used to enter Purge Mode state. When pressed, the pressure controller will purge all enabled circuits for a preset amount of time. When the purge is complete and Shot Mode is disabled, the system will automatically enter the Standby state, however if Shot Mode is enabled, the system will enter the Shot Mode state at half pressure.
* **SHOT MODE Button**: The button labeled “SHOT MODE” is used to enter the Shot Mode state. The Shot Mode state will automatically begin to adjust the pressure in the enabled circuits to the desired value. The Shot Mode state will attempt to maintain the pressure until the operator disables or aborts Shot Mode.
* **ABORT Button**: The button labeled “ABORT” is used to abort the Shot Mode state, but will lower the pressure in the enabled circuits to half of their desired value.
* **ALARM ON Button**: The button labeled “ALARM ON” is used to acknowledge an Alarm State when one is triggered. If the button is lit up but not blinking, an Alarm State is active and must be acknowledged by the operator. Pressing the button when it is lit up and not blinking will turn off the alarm (If the sound is enabled) and acknowledge the Alarm State. The button light will then blink on and off to indicate that the Alarm State is acknowledged and can be cleared. Once the operator corrects the conditions that caused the Alarm State, the operator can press the blinking button to clear the Alarm State and resume normal operations.
* **MENU NAVIGATION**: When the pressure controller is in the Standby state, the operator can press any of the buttons in the membrane on the right of the LCD to enter the Main Menu.
  + **CONTROL MODE**: The ”CONTROL MODE” option is used to set the pressure controller to work in Automatic mode or Manual mode. In Manual mode, the unit is on but will not perform functions.
  + **PRESETS**: The “PRESETS" option is used to enter the file preset submenu.
    - **SAVE PRESET**: This option is used to save the current settings to a selected preset file.
    - **LOAD PRESET**: This option is used to load a selected preset file.
    - **DELETE PRESET**: This option is used to delete a selected preset file.
    - **EXIT**: This option will exit the menu back to the Standby state.
  + **SET PRESSURES**: The “SET PRESSURES” option is used to enter the pressure settings submenu.
    - **SET MARX**: This option is used to set the desired MARX pressure to the current pressure reading in “MARX”.
    - **SET MARX TG70**: This option is used to set the desired MARX TG70 pressure to the current pressure reading in “MARX TG70”.
    - **SET MTG**: This option is used to set the desired MTG pressure to the current pressure reading in “MTG”.
    - **SET SWITCH**: This option is used to set the desired SWITCH pressure to the current pressure reading in “R. SWITCH”.
    - **SET SWITCH TG70**: This option is used to set the desired SWITCH TG70 pressure to the current pressure reading in “R. SWITCH TG70”.
    - **SET RECL. ON**: This option is used to set the pressure when the automatic reclaimer will turn on to the current pressure reading in “RETURN PSI”.
    - **SET RECL. OFF**: This option is used to set the pressure when the automatic reclaimer will turn off to the current pressure reading in “RETURN PSI”.
    - **SET MIN SUPPLY**: This option is used to set the minimum accepted pressure of the supply bottle to the current pressure reading of “SUPPLY PSI”.
    - **EXIT**: This option will exit the menu back to the Standby state.
  + **SET TIMES**: The “SET TIMES” option is used to enter the time settings submenu.
    - **SET PURGE TIMES**: This option will enter a submenu to allow the operator to set the purge time of individual circuits.
      * **MARX PURGE**: This option is used to set the desired time for the MARX purge.
      * **MARX TG70 PURGE**: This option is used to set the desired time for the MARX TG70 purge.
      * **MTG PURGE**: This option is used to set the desired time for the MTG purge.
      * **SWITCH PURGE**: This option is used to set the desired time for the SWITCH purge.
      * **SWITCH TG70 PURGE**: This option is used to set the desired time for the SWITCH TG70 purge.
      * **EXIT**: This option will exit the menu back to the Standby state.
    - **SET CIRC. DELAY**: This option will enter a submenu to allow the operator to set the circuit delay time of individual circuits.
      * **MARX DELAY**: This option is used to set the desired time for the MARX circuit delay.
      * **MARX TG70 DELAY**: This option is used to set the desired time for the MARX TG70 circuit delay.
      * **MTG DELAY**: This option is used to set the desired time for the MTG circuit delay.
      * **SWITCH DELAY**: This option is used to set the desired time for the SWITCH circuit delay.
      * **SWITCH TG70 DELAY**: This option is used to set the desired time for the SWITCH TG70 circuit delay.
      * **EXIT**: This option will exit the menu back to the Standby state.
    - **EXIT**: This option will exit the menu back to the Standby state.
  + **ALARM CONFIG**: The “ALARM CONFIG” option is used to enter the alarm configuration submenu.
    - **SOUND ON.OFF**: This option allows the user to enable or disable the alarm buzzer sound.
    - **MARX ALARM**: This option is used to set the desired time for the MARX alarm to trigger. A setting of 0 seconds will disable the MARX alarm.
    - **MARX TG70 ALARM**: This option is used to set the desired time for the MARX TG70 alarm to trigger. A setting of 0 seconds will disable the MARX TG70 alarm.
    - **MTG ALARM**: This option is used to set the desired time for the MTG alarm to trigger. A setting of 0 seconds will disable the MTG alarm.
    - **SWITCH ALARM**: This option is used to set the desired time for the SWITCH alarm to trigger. A setting of 0 seconds will disable the SWITCH alarm.
    - **SWITCH TG70 ALARM**: This option is used to set the desired time for the SWITCH TG70 alarm to trigger. A setting of 0 seconds will disable the SWITCH TG70 alarm.
    - **EXIT**: This option will exit the menu back to the Standby state.
  + **PID CONFIG**: The “PID CONFIG” option is used to enter the PID configuration submenu.
    - **MARX PID**: This option is used to set the desired MARX circuit PID tuning variables.
      * **KP**: This is used to set the KP tuning variable for the MARX PID.
      * **KI**: This is used to set the KI tuning variable for the MARX PID.
      * **KD**: This is used to set the KD tuning variable for the MARX PID.
      * **EXIT**: This option will exit the menu back to the Standby state.
    - **MARX TG70 PID**: This option is used to set the desired MARX TG70 circuit PID tuning variables.
      * **KP**: This is used to set the KP tuning variable for the MARX TG70 PID.
      * **KI**: This is used to set the KI tuning variable for the MARX TG70 PID.
      * **KD**: This is used to set the KD tuning variable for the MARX TG70 PID.
      * **EXIT**: This option will exit the menu back to the Standby state.
    - **MTG PID**: This option is used to set the desired MTG circuit PID tuning variables.
      * **KP**: This is used to set the KP tuning variable for the MTG PID.
      * **KI**: This is used to set the KI tuning variable for the MTG PID.
      * **KD**: This is used to set the KD tuning variable for the MTG PID.
      * **EXIT**: This option will exit the menu back to the Standby state.
    - **SWITCH PID**: This option is used to set the desired SWITCH circuit PID tuning variables.
      * **KP**: This is used to set the KP tuning variable for the SWITCH PID.
      * **KI**: This is used to set the KI tuning variable for the SWITCH PID.
      * **KD**: This is used to set the KD tuning variable for the SWITCH PID.
      * **EXIT**: This option will exit the menu back to the Standby state.
    - **SWITCH TG70 PID**: This option is used to set the desired SWITCH TG70 circuit PID tuning variables.
      * **KP**: This is used to set the KP tuning variable for the SWITCH TG70 PID.
      * **KI**: This is used to set the KI tuning variable for the SWITCH TG70 PID.
      * **KD**: This is used to set the KD tuning variable for the SWITCH TG70 PID.
      * **EXIT**: This option will exit the menu back to the Standby state.
    - **EXIT**: This option will exit the menu back to the Standby state.
  + **EXIT**: This option will exit the menu back to the Standby state.

# Arduino PINOUT

This section contains information about the PINOUT of the Arduino 2560 microcontroller used by the system.

* **ANALOG PINS 0 – 6**: Pins for reading analog data from pressure sensors.
  + **ANALOG PIN 0**: MARX pressure sensor.
  + **ANALOG PIN 1**: SWITCH pressure sensor.
  + **ANALOG PIN 2**: MTG pressure sensor.
  + **ANALOG PIN 3**: TG70 MARX pressure sensor.
  + **ANALOG PIN 4**: TG70 SWITCH pressure sensor.
  + **ANALOG PIN 5**: RECLAIMER pressure sensor.
  + **ANALOG PIN 6**: BOTTLE pressure sensor.
* **ANALOG PINS 7 – 13**: Pins for sending analog data from pressure sensors to displays. NOTE: THESE PINS ARE NOT CURRENTLY IN USE. DISPLAYS ARE CONNECTED DIRECTLY TO SENSORS.
  + **ANALOG PIN 7**: MARX analog data.
  + **ANALOG PIN 8**: MTG analog data.
  + **ANALOG PIN 9**: SWITCH analog data.
  + **ANALOG PIN 10**: TG70 SWITCH analog data.
  + **ANALOG PIN 11**: TG70 MARX analog data.
  + **ANALOG PIN 12**: RECLAIMER analog data.
  + **ANALOG PIN 13**: BOTTLE analog data.
* **PINS 14 – 15**: Pins for automatic reclaimer control buttons.
  + **PIN 14**: RECLAIMER STOP button.
  + **PIN 15**: RECLAIMER START button.
* **PIN 16**: ALARM SOUND relay switch.
* **PINS 17 – 21**: Pins to enable or disable individual gas circuits.
  + **PIN 17**: MARX ENABLE button.
  + **PIN 18**: MTG ENABLE button.
  + **PIN 19**: SWITCH ENABLE button.
  + **PIN 20**: TG70 SWITCH ENABLE button.
  + **PIN 21**: TG70 MARX ENABLE button.
* **PINS 22 – 31**: Pins to control gas circuit solenoid valves.
  + **PIN 22**: MARX OUT solenoid valve.
  + **PIN 23**: MARX IN solenoid valve.
  + **PIN 24**: MTG OUT solenoid valve.
  + **PIN 25**: MTG IN solenoid valve.
  + **PIN 26**: SWITCH OUT solenoid valve.
  + **PIN 27**: SWITCH IN solenoid valve.
  + **PIN 28**: TG70 SWITCH OUT solenoid valve.
  + **PIN 29**: TG70 SWITCH IN solenoid valve.
  + **PIN 30**: TG70 MARX OUT solenoid valve.
  + **PIN 31**: TG70 MARX IN solenoid valve.
* **PINS 32 – 33**: Pins to control the reclaimer relay switches.
  + **PIN 32**: RECLAIMER STOP relay switch.
  + **PIN 33**: RECLAIMER START relay switch.
* **PINS 34 – 37, (PIN 1)**: Pins for system function buttons.
  + **PIN 34**: SHOTMODE button.
  + **PIN 35**: PURGE button.
  + **PIN 36**: ALARM button.
  + **PIN 37**: AUTOMATIC RECLAIMER button.
  + **PIN 1**: ABORT SHOT button.
* **PINS 38 – 49**: Pins for button LEDs.
  + **PIN** **38**: MARX ENABLE button LED.
  + **PIN 39**: MTG ENABLE button LED.
  + **PIN 40**: SWITCH ENABLE button LED.
  + **PIN 41**: TG70 SWITCH ENABLE button LED.
  + **PIN 42**: TG70 MARX ENABLE button LED.
  + **PIN 43**: ALARM button LED.
  + **PIN 44**: EXTRA button LED.
  + **PIN 45**: SHOTMODE button LED.
  + **PIN 46**: PURGE button LED.
  + **PIN 47**: RECLAIMER START button LED.
  + **PIN 48**: RECLAIMER STOP button LED.
  + **PIN 49**: AUTOMATIC RECLAIMER button LED.
* **PINS 50 – 53**: Pins for the SD card reader.
  + **PIN 50**: MISO pin.
  + **PIN 51**: MOSI pin.
  + **PIN 52**: SCK pin.
  + **PIN 53**: CS pin.

# Connector PINOUTS

This section contains information about the PINOUTS for the sensor connector, the relay connector, and the reclaimer connector.

## Sensor Connector

The pressure sensor connector is the DB25 connector. The positive and negative connections are common between all sensors. The PINOUT is as follows:

* **PINS 2 – 4**: TG70 MARX sensor.
  + **PIN 2**: TG70 MARX sensor data.
  + **PIN 3**: TG70 MARX sensor GND.
  + **PIN 4**: TG70 MARX sensor positive.
* **PINS 5 – 7**: MTG sensor.
  + **PIN 5**: MTG sensor data.
  + **PIN 6**: MTG sensor GND.
  + **PIN 7**: MTG sensor positive.
* **PINS 8 – 10**: SWITCH sensor.
  + **PIN 8**: SWITCH sensor data.
  + **PIN 9**: SWITCH sensor GND.
  + **PIN 10**: SWITCH sensor positive.
* **PINS 11 – 13**: MARX sensor.
  + **PIN 11**: MARX sensor data.
  + **PIN 12**: MARX sensor GND.
  + **PIN 13**: MARX sensor positive.
* **PINS 17 – 19**: RECLAIMER (IN) sensor.
  + **PIN 17**: RECLAIMER sensor data.
  + **PIN 18**: RECLAIMER sensor GND.
  + **PIN 19**: RECLAIMER sensor positive 12V.
* **PINS 20 – 22**: BOTTLE (OUT) sensor.
  + **PIN 20**: BOTTLE sensor data.
  + **PIN 21**: BOTTLE sensor GND.
  + **PIN 22**: BOTTLE sensor positive.
* **PINS 23 – 25**: TG70 SWITCH sensor.
  + **PIN 23**: TG70 SWITCH sensor data.
  + **PIN 24**: TG70 SWITCH sensor GND.
  + **PIN 25**: TG70 SWITCH sensor positive.
* **AMPHENOL 6**: SENSOR INTERMEDEATE CONNECTOR
  + **PIN B**: Signal.
  + **PIN F**: +5V.
  + **PIN D**: GND.

## Relay Connector

The gas circuit solenoid relay connector is the Amphenol 24-28S connector labeled **J1**. The negative connection is common between all solenoids. The PINOUT is as follows:

* **PINS A – E**: OUT solenoid connections.
  + **PIN A**: MARX OUT solenoid connection.
  + **PIN B**: SWITCH OUT solenoid connection.
  + **PIN C**: MTG OUT solenoid connection.
  + **PIN D**: TG70 MARX OUT solenoid connection.
  + **PIN E**: TG70 SWITCH OUT solenoid connection.
* **PINS R, W – Z**: IN solenoid connections.
  + **PIN R**: MARX IN solenoid connection.
  + **PIN W**: SWITCH IN solenoid connection.
  + **PIN X**: MTG IN solenoid connection.
  + **PIN Y**: TG70 MARX IN solenoid connection.
  + **PIN Z**: TG70 SWITCH IN solenoid connection.
* **PINS F, G, H, K, L, M, N, S, T, U**: Common negative connections.
* **PINS J, P, Q, V**: Unused pins.

## Reclaimer Connector

The reclaimer relay connector is the Amphenol 18-1S connector labeled **J2**. The PINOUT is as follows:

* **PINS A – C**: RECLAIMER START relay connections.
  + **PIN A**: RECLAIMER START COMMON relay connection.
  + **PIN B**: RECLAIMER START NORMALLY OPEN relay connection.
  + **PIN C**: RECLAIMER START NORMALLY CLOSED relay connection.
* **PINS F – H**: RECLAIMER STOP relay connections.
  + **PIN F**: RECLAIMER STOP COMMON relay connection.
  + **PIN G**: RECLAIMER STOP NORMALLY OPEN relay connection.
  + **PIN H**: RECLAIMER STOP NORMALLY CLOSED relay connection.
* **PINS D, E, I, J**: Unused pins.

# Code Documentation

This section contains the documentation of the C++ code used for the Arduino 2560 microcontroller.

## Libraries Used

The following libraries were used in this program to drive necessary components.

#include <SD.h> *Used to drive SD card reader read and write functionality.*

#include <SPI.h> *Used for Serial Peripheral Interface communication.*

#include <Key.h> *Used to receive keyboard inputs. Unused.*

#include <Keypad.h> *Used to drive keypad matrix button membrane.*

#include <PID\_v1.h> *Used to calculate time window for opening solenoids.*

#include <LiquidCrystal.h> *Used to drive the LCD interface.*

## Functions

The following functions are used in the program.

* void setup()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to initialize the LCD, SD card reader, button membrane, Arduino pin modes, and button pull-up resistors. This function also reads the SD card for preset file data and initializes the pressure controller to the last used settings, or sets them to default settings if no presets exist.
* void main()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This is the main loop function. It will check for user inputs on the button membrane to access the menu, display the current operating mode, initiate shot mode, initiate purge mode, and to abort the shot.
* void ShotPressure()
  + **PARAMETERS**:
    - *bool half:* A flag to toggle between half pressure setting and full pressure setting.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function uses the desired setpoints for each circuit and passes them to the circuit PID. It then loops through each enabled circuit and calls functions to raise or lower the pressure in the circuit. Once the pressure is reached or shotmode is exited, the function will ensure that all solenoids are closed and will exit back to standby mode or await further action.
* void abortShot()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function aborts the sot mode and sets current pressure settings of enabled circuits to half pressure.
* void RaisePressure()
  + **PARAMETERS**:
    - *bool& circuitState:* A flag to check if a given circuit is enabled or disabled.
    - *double divisor:* The divisor used to find the half pressure setting of a given circuit.
    - *bool half:* A flag to toggle between half pressure and full pressure setting.
    - *double &currentpressure:* The current pressure sensor reading for a given circuit.
    - *double Threshold:* The desired setpoint of the given circuit.
    - *const int relayPin:* The Arduino pin corresponding to the inlet solenoid relay switch of a given circuit.
    - *const int analogPin:* The Arduino pin corresponding the pressure sensor of a given circuit.
    - *const int WindowSize:* The time window for the PID controller of a given circuit.
    - *double& input:* PID input variable for the PID controller of a given circuit.
    - *double& output:* PID output variable for the PID controller of a given cituit.
    - *PID& System:*The PID controller of a given circuit.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to increase the pressure of a given circuit by using a PID controller to reach the target pressure without overshooting. If the pressure cannot be increased, an error state is triggered.
* void ReducePressure()
  + **PARAMETERS**:
    - *bool& circuitState:* A flag to check if a given circuit is enabled or disabled.
    - *double divisor:* The divisor used to find the half pressure setting of a given circuit.
    - *bool half:* A flag to toggle between half pressure and full pressure setting.
    - *double &currentpressure:* The current pressure sensor reading for a given circuit.
    - *double Threshold:* The desired setpoint of the given circuit.
    - *const int relayPin:* The Arduino pin corresponding to the outlet solenoid relay switch of a given circuit.
    - *const int analogPin:* The Arduino pin corresponding the pressure sensor of a given circuit.
    - *const int WindowSize:* The time window for the PID controller of a given circuit.
    - *double& input:* PID input variable for the PID controller of a given circuit.
    - *double& output:* PID output variable for the PID controller of a given cituit.
    - *PID& System:*The PID controller of a given circuit.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to reduce the pressure of a given circuit by using a PID controller to reach the target pressure without overshooting. If the pressure cannot be rduced, an error state is triggered.
* void Purge(bool& circuitState, int intakerelayPin, int exhaustrelayPin)
  + **PARAMETERS**:
    - *bool& circuitState:* A flag to check if a given circuit is enabled or disabled.
    - *int intakerelayPin:* The Arduino pin corresponding to the inlet solenoid relay switch of a given circuit.
    - *int exhaustrelayPin:* The Arduino pin corresponding to the outlet solenoid relay switch of a given circuit.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to purge a given circuit by opening both the inlet and outlet solenoids for a given amount of time.
* void ControlButtonStateManager()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to check the state of each button on the user interface. If a button is pressed, the function will toggle the mode that corresponds to that button and call to enable or disable the LED light within. It is used to select operating functions and to enable and disable individual circuits.
* void ControlButtonStateCheck()
  + **PARAMETERS**:
    - *int reading:* The current reading from the Arduino pin corresponding to a given button.
    - *bool& buttonState:* The current state of a given button.
    - *bool& lastbuttonState:* The previous state of a given button.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function checks to see if the operator has pressed a button on the user interface, and will toggle the state when pressed while debouncing the signal.
* void ControlButtonStateCheckReclaimer()
  + **PARAMETERS**:
    - *int reading:* The current reading from the Arduino pin corresponding to the automatic reclaimer button.
    - *bool& buttonState:* The current state of the automatic reclaimer button.
    - *bool& lastbuttonState:* The previous state of the automatic reclaimer button.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function checks to see if the operator has pressed the automatic reclaimer button, and will toggle the state when pressed while debouncing the signal.
* void ControlButtonStateCheckAlarm()
  + **PARAMETERS**:
    - *int reading:* The current reading from the Arduino pin corresponding to the alarm button.
    - *bool& buttonState:* The current state of the alarm button.
    - *bool& lastbuttonState:* The previous state of the alarm button.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function checks to see if the operator has pressed the alarm button, and will toggle the state when pressed while debouncing the signal.
* void ControlButtonLEDStateCheck()
  + **PARAMETERS**:
    - *bool buttonState:* The current state of a given button.
    - *const int ledPin:* The Arduino pin corresponding a given LED light.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function toggles a given LED light based on the state of the corresponding button.
* void menu()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function controls the navigation of the menu on the LCD. The function reads signals from the button membrane and then navigates to the appropriate menu item, then calls to execute a given function that corresponds to the menu item.
* bool AutoManual()
  + **PARAMETERS**: None.
  + **RETURNS**:
    - *bool selection:* The true or false selection made by the operator.
  + **DESCRIPTION**: This function is used to allow the operator to select if they wish to run the controller in automatic mode or manual mode.
* bool YesNo()
  + **PARAMETERS**: None.
  + **RETURNS**:
    - *bool selection:* The yes (true) or no (false) selection made by the operator.
  + **DESCRIPTION**: This function is used to allow the operator to select “Yes” or “No” to confirm an operation.
* void SaveCurrentSettings()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to save the current settings of the controller to the preset file to the SD card that is loaded when the system boots up.
* bool FileWriter()
  + **PARAMETERS**: None.
  + **RETURNS**:
    - *bool true:* The function has been executed.
    - *bool false:* The user aborted the function.
  + **DESCRIPTION**: This function is used to save the current settings of the controller to a specific preset file on the SD card.
* bool FileReader()
  + **PARAMETERS**: None.
  + **RETURNS**:
    - *bool true:* The function has been executed.
    - *bool false:* The user aborted the function.
  + **DESCRIPTION**: This function is used to load a selected preset file from the SD card into the controller.
* bool FileRemover()
  + **PARAMETERS**: None.
  + **RETURNS**:
    - *bool true:* The function has been executed.
    - *bool false:* The user aborted the function.
  + **DESCRIPTION**: This function is used to delete a selected preset from the SD card.
* void SetPressure ()
  + **PARAMETERS**:
    - *int selection:* The number corresponding to the circuit selected by the operator.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to create a new setpoint for the selected circuit. Once a setpoint is set by the operator, the new setpoint is saved to the current setting file.
* void manualReclaimerControl()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function allows the operator to assume manual control of the reclaimer.
* void automaticReclaimerControl()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to automatically control the reclaimer.
* void checkSupply()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to monitor the pressure of the supply bottle. Calls for an error state if the supply is too low.
* void purgeConfig ()
  + **PARAMETERS**:
    - *int selection:* The number corresponding to the circuit selected by the operator.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function allows the operator to configure the purge settings for a selected circuit. Once a setting is changed, it is saved to the current setting file.
* void circuitDelay ()
  + **PARAMETERS**:
    - *int selection:* The number corresponding to the circuit selected by the operator.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to create a new time delay for the selected circuit. Once a new time delay is set by the operator, the new time delay is saved to the current setting file.
* void setPID ()
  + **PARAMETERS**:
    - *int selection:* The number corresponding to the circuit selected by the operator.
    - *Int tuneVariable:* The PID tuning parameter selected by the operator.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function allows the operator to edit the PID tuning parameters of a selected circuit. The new PID tuning parameters are then saved to the current setting file.
* void setAlarmOnOff ()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function allows the operator to enable or disable the sound from the alarm.
* void alarmConfig()
  + **PARAMETERS**:
    - *int selection:* The number corresponding to the circuit selected by the operator.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function allows the operator to configure the alarm settings for each individual circuit. The new alarm settings are saved to the current setting file.
* void alarmController ()
  + **PARAMETERS**:
    - *String errorString:* The string describing the error to be shown to the operator.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to display to the operator on the LCD which error the system is currently experiencing. The function lets the user acknowledge the alarm and then clear the error to resume operations.
* void resetLCD ()
  + **PARAMETERS**: None.
  + **RETURNS**: Nothing.
  + **DESCRIPTION**: This function is used to reset the LCD.