**System Testing and Demonstration**

Once prototypes for the fire alarms and hub are completed, unit testing and hardware testing will need to be run by the term to ensure the system is running as intended. The testing will be done in two phases. The first phase will be the hardware testing. The hardware testing will first need to be performed on the breadboard circuit designs.

Once the breadboard circuit designs pass all applicable tests, the team can be certain that their designs meet our requirements for our fire alarm system. This will allow to team to send out the schematics to the PCB manufactures and print the final PCB designs. The hardware testing well then be done again once the parts are soldered to the printed circuit boards. This will ensure that all parts are working as intended. The hardware tests will be performed on all essential hardware parts in the circuit.

**Hardware Tests**

The first hardware test that will need to be completed will be a test to confirm that the buzzer is working properly. The team will write a script for the fire alarm that turns on the buzzer for 5 seconds and then turns off the buzzer. The scrip will be using the same Arduino tone() function that was specified earlier in the hardware section of this document. The script will then turn on the buzzer at a higher pitch for 5 seconds and then turn off the buzzer again. This will ensure that the buzzer is working as intended. The script will be loaded onto the fire alarm using the Arduino development board as the USB serial connection

The second hardware test that will need to be completed will be a test to confirm that the left arrow LED is working properly. The team will write a script for the fire alarm that turns the pin connected to the left LED high for 5 seconds using the pinout function built in to the Arduino programming language. After 5 seconds the script will run the pinout function again but to turn the LED low. The test will repeat this one more time to ensure that the LED is working as intended. The script will be loaded onto the fire alarm using the Arduino development board as the USB serial connection similar to the first test.

The third hardware test that will need to be completed will be a test to confirm that the right arrow LED is working properly. The team will write a script for the fire alarm that turns the pin connected to the right LED high for 5 seconds using the pinout function built in to the Arduino programming language. This will be identical to the second hardware test but will be turning on a different pin on the ATmega 328. After 5 seconds the script will run the pinout function again but to turn the LED low. The test will repeat this one more time to ensure that the LED is working as intended.

The fourth hardware test that will need to be completed will be a test to confirm that the middle LED is working properly. The middle LED is the LED to alert users to turn around as they are going the wrong direction. The team will write a script for the fire alarm that turns the pin connected to the middle LED high for 5 seconds using the pinout function built into the Arduino programming language. This will again be identical to the second and third hardware test but will be turning on the pin connected to the middle LED on the ATmega 328. After 5 seconds the script will run the pinout function again but to turn the LED low. The test will repeat this one more time to ensure that the LED is working as intended.

The fifth hardware test that will need to be completed will be to test to confirm that the battery monitor for the backup battery is working as intended. The team will write a script that passes a voltage through the proper pin and measures the resistance across that area. This resistance reading read by this pin will tell the system how much power is left in the battery. The test will read this value and spit it out onto the screen. This will ensure that the backup battery is working as intended.

The sixth and last hardware test that will need to be completed will be to test and confirm that the Xbee wireless receiver can send and receive a signal from the hub system. The test will also confirm that the hub can send and receive signal from a fire alarm. The team will configure the Xbee modules to connect to each other as described in the wireless configuration section of this paper. Once the Xbee modules have been configured the system be will able to run our Xbee module test. The hub will be running a script that will wait for it to receive a message and then reply that it has received the message. The fire alarm will run a script that uses the Xbee send function to send the message “test” to the hub. The fire alarm will then wait to receive a message. The hub will wait to receive a message called “test”. Once it receives this message it will send a message back to the fire alarm. The message that is sent will be the message “test received”. The person who is testing the software will be looking at the hub to see that the message “test: has been received by the system. The tester will then look at the computer connected to the fire alarm and verify that the message “test received” has been received by the fire alarm and sent out to the screen. Once these two conditions are met we can verify that the system is able to send and receive wireless signals the way that the system needs to and the modules are working as expected.

Once all the above hardware tests pass on the printed circuit board module, we will be able to verify that all components work as intended. We can then work to test out software algorithms and system as a whole. These hardware tests will be performed on every fire alarm periodically to verify that each alarm is working as intended.

**Software and System tests.**

The next set of tests that will be run on the system will be used to verify that the entire system works as a cohesive unit and that our exit searching algorithm is working as intended.

The first test that will need to be performed on the system will be the multi fire alarm buzzer test. The hub will send out a signal to each and every alarm in order to verify that the wireless modules, buzzers, and LEDs are working as intended within the scope of the Smart fire alarm system. The hub will send out an Xbee wireless signal that turns on the fire alarm buzzer and all of the LEDs for the first alarm. This will go on for five seconds on the first alarm. The hub will then send out a reset signal to the first alarm to turn off the buzzers and LEDs. The hub will then send out a Xbee wireless signal to the second alarm that will turn on the that alarm’s buzzer and LEDs for five seconds. After the five seconds, the hub will again send out a reset signal to the second alarm. The hub will then repeat this process for the third, fourth, and fifth alarms. Once this test is completed the team will be able to verify that all alarms are connected and working as intended.

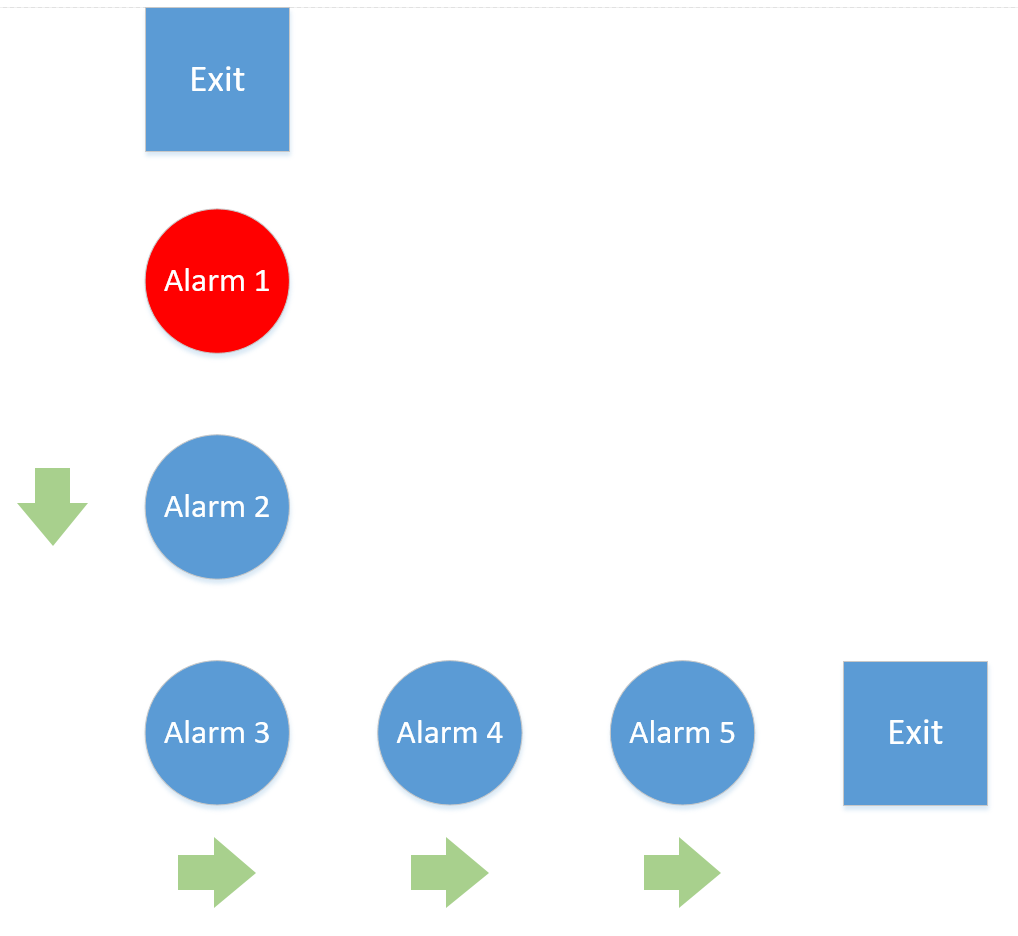
The next set of tests will be testing the algorithm for detecting the best path to the exits. The first test that we will verify will be checking the algorithm on a simple line. The fire alarms will need to be put into a line starting with alarm 1 on the left and proceeding in order until alarm 5 is all the way to the right. This will position alarm 3 in to the middle of the line of alarms. The test will also load an exit to the left of alarm 1 and to the right of alarm 5. This will position exits at both ends of the line. The test will set off alarm 3 which will tell that hub that there is a fire in the middle of the hallway of alarms. The hub should then set alarms 1 and 2 left LEDs to high telling users to head to the left exit. The hub should also then set alarms 4 and 5 right LEDs to high telling users to head to the right exit. If this is indeed the case then the system and algorithm is working is intended. After 15 seconds the hub will send out a reset signal to reset the LEDs and buzzers for all of the alarms.



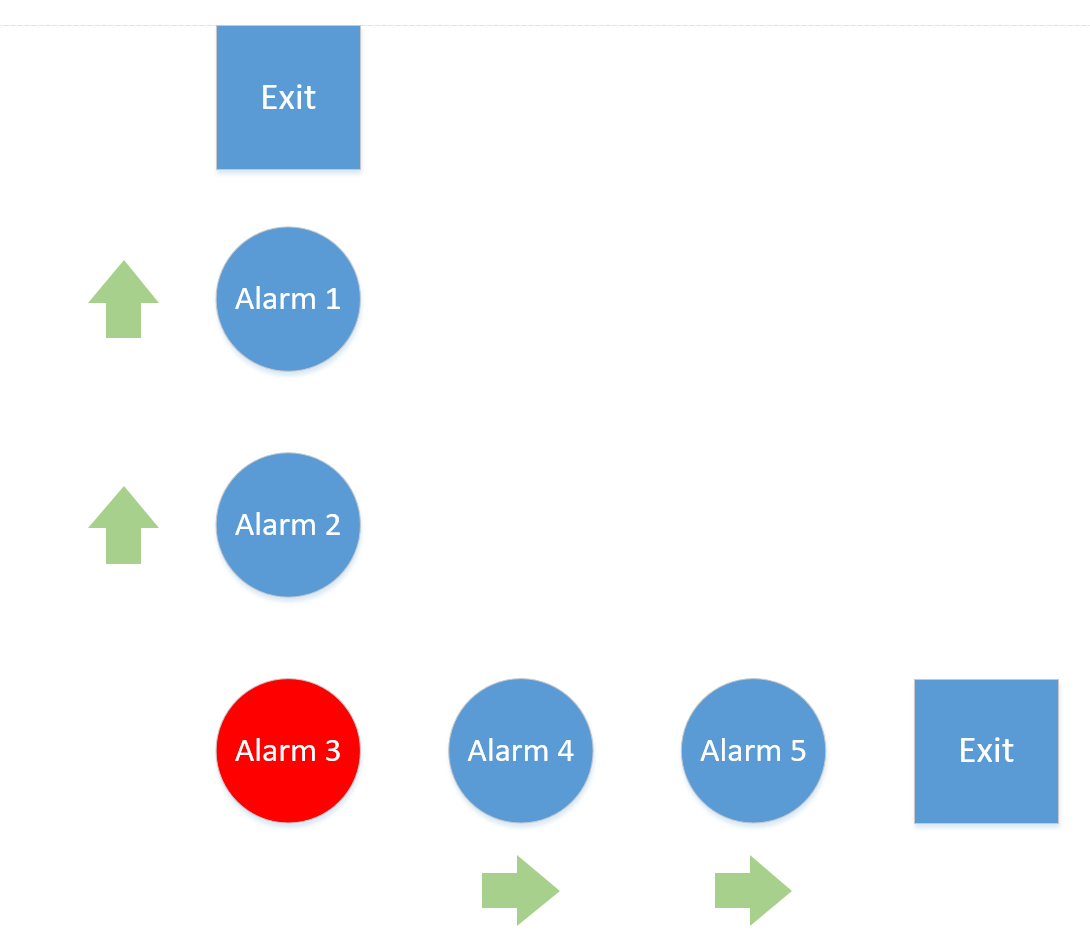
The next test that will be run will also be testing the fire alarms in a line position. The fire alarms will again need to be put into a line starting with alarm 1 on the left and proceeding in order until alarm 5 is all the way to the right. This will again position alarm 3 in the middle of the line of alarms. The test will also again load an exit to the left of alarm 1 and to the right of alarm 5. The different between this test and the first test will be that this test will set off alarm 1 instead of alarm 3. This will act as an edge case test for our system. All alarms should ignore the exit to the left of alarm one and point the users to the right. This will lead all users of the system to the exit to the right of alarm 5. If this is indeed the case that our system and algorithm is running as expected.



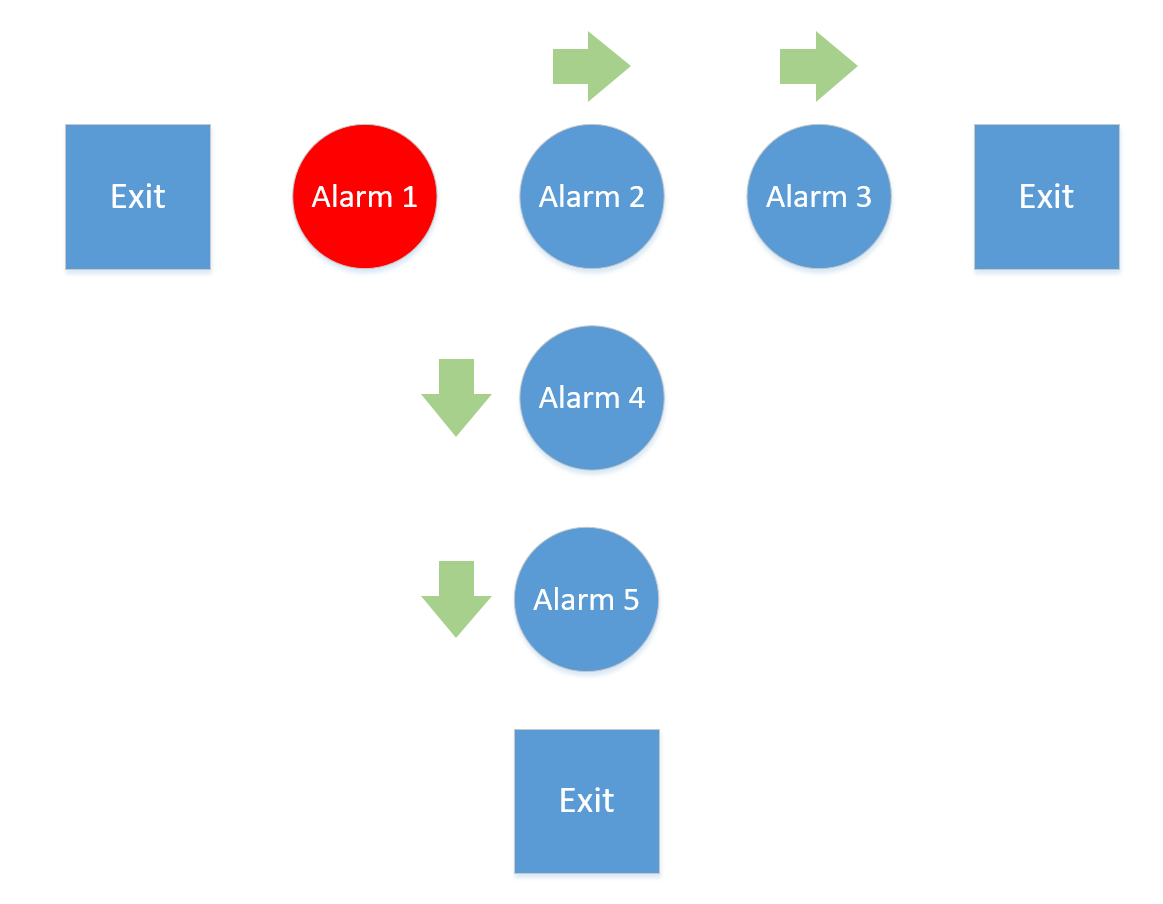
The next test that will be run will be in the shape of an L. This can also be called an elbow. The point of this text will confirm that the system can correctly handle a turn or corner in a hallway. Alarm 1 will be at the top. Alarm 2 will be positions to the bottom of alarm 2. Alarm 3 will be positioned below alarm 2. At this point we will have created a line similar to the above tests. Now alarm 4 will be placed to the right of alarm 3 and alarm 5 will be placed to the right of alarm 4. This will create an L shape with alarm 3 at the corner of the L. We will then place an exit above alarm 1 and another exit to the right of alarm 5. The hub will then send out a signal to turn on alarm 3. If the algorithm is working as expected, alarms 1 and 2 will send users to the top exit while alarms 4 and 5 will send users to the right exit. This will confirm that the system is able to handle emergencies at the corner of a hallway.



The next test will branch off of the above test. This same L shape will be created with alarm 1 at the top, alarm 3 at the center of the corner, and alarm 5 at the right. There will be an exit at the top of alarm 1 and another exit to the right of alarm 5. The hub will send a signal to alarm 1 telling it to go off. If the algorithm is working correctly, the system should send users from alarm 2 around the corner to the exit past alarm 5. If this is indeed the case, then the system is handling corners as expected.



The last type of configuration that will be used to test the system will be that of the T shape. Users will set up alarms 1, 2, and 3 in a line. Alarm 4 will be to the bottom of alarm 2 .Alarm 5 will be to the bottom of alarm 4. There will be 3 exits connects to this system. There will be one exit to the left of alarm 1. There will be another exit to the right of alarm 3. Lastly, there will be one exit to the bottom of alarm 5. The trick here is to set off an alarm at one of the endpoints allowing the system to make a true calculation on what decisions should be made. The hub will send a signal to set off alarm 1. This will have the system make calculations for the other alarms. If the system is working correctly, alarms 2 and 3 should point users to the exit to the right of alarm 3. Alarms 4 and 5 should be pointing to the exit below alarm 5. This ensures that alarms 4 and 2 had to make real choices for which exit it should head to. If the system performs the above operations, we can confirm that the system and algorithm is working as intended.



Both the hardware tests and software unit tests will be run multiple times throughout the design process of our system. The hardware tests will be run during the initial phase as we are building and designing the hardware circuits for the hub and fire alarm. The software unit tests will be run during development of the algorithm both within a full software emulation environment as well the hardware fire alarms themselves. This will aid in confirming that we are on the right track during development as well as confirming that the system works as it is intended once it is finished.