Potter Bar update 4/1

Hey Adam! Sending you an update on the research and work I've been doing in between trips to Chicago for work :-P. After meeting with you, Savannah, and Guy, I set out to prove the master controller, scene controller design we decided upon. I've not seen any microcontroller design similar, so I think we are breaking exciting new ground with this design. Based on the design, I'm tackling it in an evolutionary way proving out the scene controller design first, then moving onto the master scene controller. Over the last couple weeks, I've focused on testing the scene controller design.

Looking at the 5 scenes, they fall into 2 architectures:

- "Basic": Actuator, lighting, & audio triggered manually. This includes Lachlan and the shrunken heads.
- "Complex": One actuator, lighting, & audio triggered by proximity sensor and another actuator, lighting & audio triggered by IR wand. This includes both windows and the door.

I've been working on the "complex" architecture, as it is a superset of the functionality required by the "basic" as well. My first goal is to determine whether each window scene can be controlled by a single Arduino controller or whether it needs to be divided up between controllers for each sensor/actuator.

Single Arduino Controller Design

In this design, both sensors, both actuators, lighting, and audio will be controlled by a single Arduino. For compactness and simplicity, this is the approach I'm hoping will work and have been focused on proving.

Multi-Arduino Controller

If the above design doesn't perform as I hope, I'll separate the sensors, actuators, & lighting, from the scene controller.

Interrupt Handling

The challenge regardless of approach is handling the interrupts that can occur during scene execution. There are 2 interrupts; triggering of the wand "IR action" and triggering of the overall "master scene".

IR Sensor Interrupt

This interrupt occurs when someone triggers the IR sensor using the wand. The desire is for the "spell action" to immediately start following the IR sensor interrupt. As long as the proximity sensor recognizes someone in front of it, it will run the "proximity action" (e.g. frog jumping). The occurrence of the IR interrupt (wand) can happen at any point during this proximity action. I can't just freeze the proximity action; it needs to return to a "start state" (e.g. frog lands or knocker stops). At this point it will be in a state ready to start when the IR action completes. This parallel processing has been one of my focal points.

Master Scene Interrupt

This interrupt occurs when you trigger the Voldemort master scene. The master controller will send a “pause” signal to each of the scene controllers to stop while the master scene executes. This “pause” can’t just stop what is currently executing, it needs to return to the “start state” so that when it receives a “resume” trigger from the master controller (when the master scene completes), it can start clean. If this didn’t happen, the candy stick might be mid-flight or the frog might be mid-hop. We also have to stop any currently playing scene specific audio and change the lighting.

The master scene controller will send a “resume” request to all of the scene controllers to re-enable their sensors and handling proximity and IR actions.

When you want to “shutdown” at the end of the night, the master controller will send a “stop” request to the scene controllers stopping all sensors and turning off the lighting.

Recent Focus

I’ve been focusing on creating a reusable execution framework to run in each scene controllers to handle the parallel execution of proximity and IR actions and to handle the start/stop/pause/resume commands from the master controller.

I also have the *individual* proximity actions (frog jump, door knocker, audio, lighting) and *individual* spell actions working (candy stick rise, fan relay, sound, lighting) independently. The next step is to combine the individual actions into collective scene proximity and spell actions and then configure the execution framework to invoke these actions.

Task List

To update you on an ongoing basis, I’ve put together the task list below of the things I need to build/test.

1. ✓Create frog hop script
2. ✓Create levitating candy script
3. ✓Create lighting control script
4. ✓Create audio control script
5. ✓Create door knocker script
6. Create bell ringing script
7. ✓Create relay (fan) control script
8. Assemble "proximity action scripts"
    a. Ice cream store window proximity action script = lighting control script + frog hop script
    b. Door proximity action script = door knocker script + shopkeeper audio script
    c. Tattings proximity action script = audio control script + lighting control script
9. Assemble "IR action scripts"
    a. Ice cream store IR action script = lighting control script + levitating candy script
    b. Door IR action script = bell ringing script
    c. Tattings IR action script = fan relay script + lighting script

10. Assemble “manual action scripts”
    a. Lachlan = button + servo + spot + audio
    b. Shrunken heads = button + 3 servos + 3 spots + audio
11. Create “Master scripts”
    a. Startup
    b. Shutdown
    c. Voldemort
12. Create “Scene Controller Execution Framework”
    a. ✓Execute interpreted scene script rather than compiled scene script in preparation for script refresh from master controller.
    b. ✓Recognize when someone is in front of the proximity sensor and execute proximity action script.
    c. ✓Recognize when someone triggers IR via the wand and execute the IR action script.
    d. When IR action is triggered, transition from proximity action to IR action and back cleanly.
    e. Handle “start”, ✓“pause”, ✓“resume”, and “shutdown” requests from master scene controller when a master scene is triggered.
    f. Poll the master controller upon startup to check for updated scene script.
13. Using framework above, combine proximity action script and IR action script to determine if scene controller can run on single Arduino or if functionality needs to be distributed among multiple Arduinos.
14. Create “Master Scene Controller Execution Framework”
    g. Allow Adam to “start” all scenes
        i. Trigger scene controllers to “start”
    h. Allow Adam to trigger master scenes
        ii. Trigger scene controllers to “pause” on start of master scene
        iii. Trigger scene controllers to “resume” after master scene
    i. Allow Adam to “shutdown” all scenes
        iv. Trigger scene controllers to “shutdown”
    j. Allow Adam to trigger Lachlan scene
        v. What should happen to other scenes if they are underway?
    k. Allow Adam to trigger shrunken head scene