Botball Starting Lights
Steven Blasberg and Alexander Nabavi-Noori
Palm Desert Middle School

Botball Starting Lights

1. Hardware Design

In robotics, often we want to control something that uses high voltages, but with a low voltage controller. This can be established with the use of relays. Now, what is a relay? Well, according to Wikipedia, “A relay is an electrical switch that opens and closes under the control of another electrical circuit.”[1] This means that we can control a 120 volt AC circuit with the use of a battery, or in our case the motor output of a robotics controller.

1.1 Getting an idea

We thought it would be a fun idea to to make a light turn on and off with a Handyboard or other robotics controller, such as an XBC. We got the idea from the Botball starting lights. We contacted Dr. David Miller about how it was done, and he replied, “The trick is not in the code, but in the starting lights. We put a 5v relay into the starting lights so that they can be turned on and off by powering the relay (or not). We then connect the relay to the motor ports on the Handyboard or XBC. When the motor port is turned on at full speed, the relay closes and the light turns on.”

1.2 Finding the right relay

We researched different types of relays, and found one from Radio Shack. It was a bit pricey for the quality, but it was the only option without having it shipped in. This relay needs 5 volts DC to trip the relay, and the other circuit has a maximum of 110 volt AC at 1 amp. We found this sufficient, because a light bulb rated at 100 watts and 120 volts would draw .83 amperes according to Ohms Law that states amperes = watts / volts.[2] Still, just to be safe, we used a 75 watt light bulb.

1.3 Parts used

The cost of this relay which on average can withstand about 100,000 cycles was $5.00 each. This was the main component of the build, but there were also other items used. These were a circular junction box, a cover for the box, a security light fixture, a cheap extension cable, and signal wire.

1.4 Assembly

We started the assembly off by soldering the two signal wires to the relay at the pins meant to trip the relay. Next, we soldered a spare bit a wire to the common pin, and the normally open pin. Then, we attached one of the two wires from the cheap extension cable with the end cut off and one wire from the switch to the wire attached to the relay’s common pin. Next, we attached the other end of the switch and one wire from the light socket to the wire attached to the normally open pin. Afterwards, we connected

the other wire from the light to the other wire from the extension cable. Finally, we sealed everything up, and tested if it worked. When we saw that it worked, we made another one. Figure 1 shows the electrical diagram of the circuit, and Figure 2 shows what the inside of the box looks like.

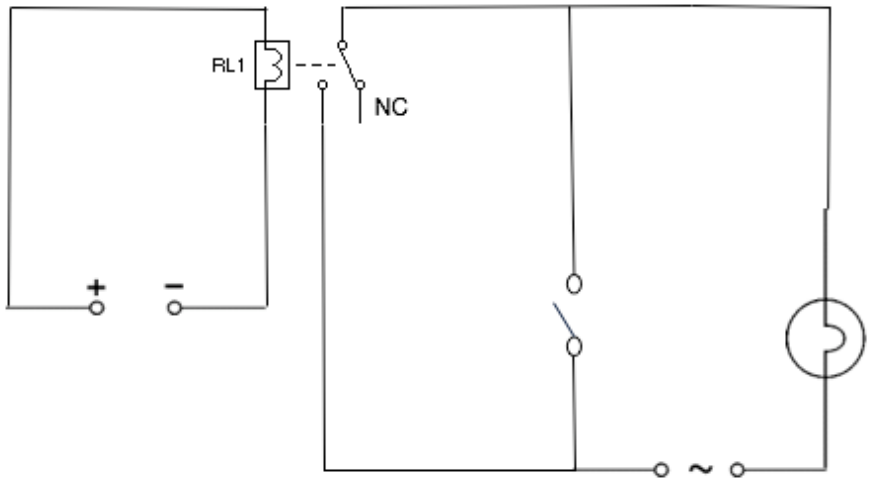


Figure 1: This image shows an electrical diagram of the circuit inside the junction box.

Figure 2: This picture shows the inside of the junction box.

2. Software Design

In this segment you will learn how to write the code for turning on and off the lights using a Handyboard. This can be used in applications such as conducting the botball game. Our program for this application, starts very simple, to very complex.

2.1 The Bare Minimums

As you already have read, when we contacted Dr. David Miller he said, " The trick is not in the code, but in the starting lights..." We didn't like how this sounded, because it implied that the code would be simple, short, and essentially boring. So we first started with the basics of what needed to be done: to trigger the relay, and to sleep to keep it on for five seconds after the game starts, and flash it once at the end. So the code would look like it does in Figure 3 [14 lines of code (keep this number in mind)]:

```
int wait_time=random(10)+5;
void main()
{
    while(wait_time>0)
      {sleep(1.);
        wait_time=wait_time-1;
    }
    fd(0);
    fd(1);
    sleep(5.);
    ao();
    sleep(114.);
    fd(1);
    fd(0);
    sleep(1.);
    ao();
}
```

Figure 3: These are the 14 lines of code for the minimum operation of the lights.

This basically states that it would sleep a random of 5-14 seconds. Then, once game has started, it would turn on both lights for five seconds, and then turn them off. After this, it would wait until one second before the game ends and flashes the light for the final second. To our perfectionist programmer, Alexander, this was just way too simple, and it just didn't have enough features yet. So what we decided to do was to start to add things.

2.2 Calibration Sequence

The first feature we could think of that would be a great convenience to have, was a calibration sequence! What we thought we would do was have the option when starting the code to go through a timed (90 seconds as in the real game) calibration sequence where the Handyboard would handover control of the light to the switch built on it. When 90 seconds had passed, the Handyboard would print

that the calibration sequence was over then it would continue on with the game sequence. The code would now consist of 29 lines of code.

2.3 Modified Game Sequence

Now seeing as this worked and was quite useful, we decided to tackle our very simple game sequence. This still consisted of just turning the lights on, waiting five seconds, turning it off, and blinking only once at the end. Very boring. So we decided that the only things we could change in it while still keeping it functional was to make the lights blink at the end of the game. What we thought we'd do was first, make a function that would blink the lights for whatever amount of time specified, and also would beep after it blinked. We also decided to have it blink the full fifteen times in the five seconds before the end of the game. With all the new functions and lines of code written, the code now consists of 42 lines.

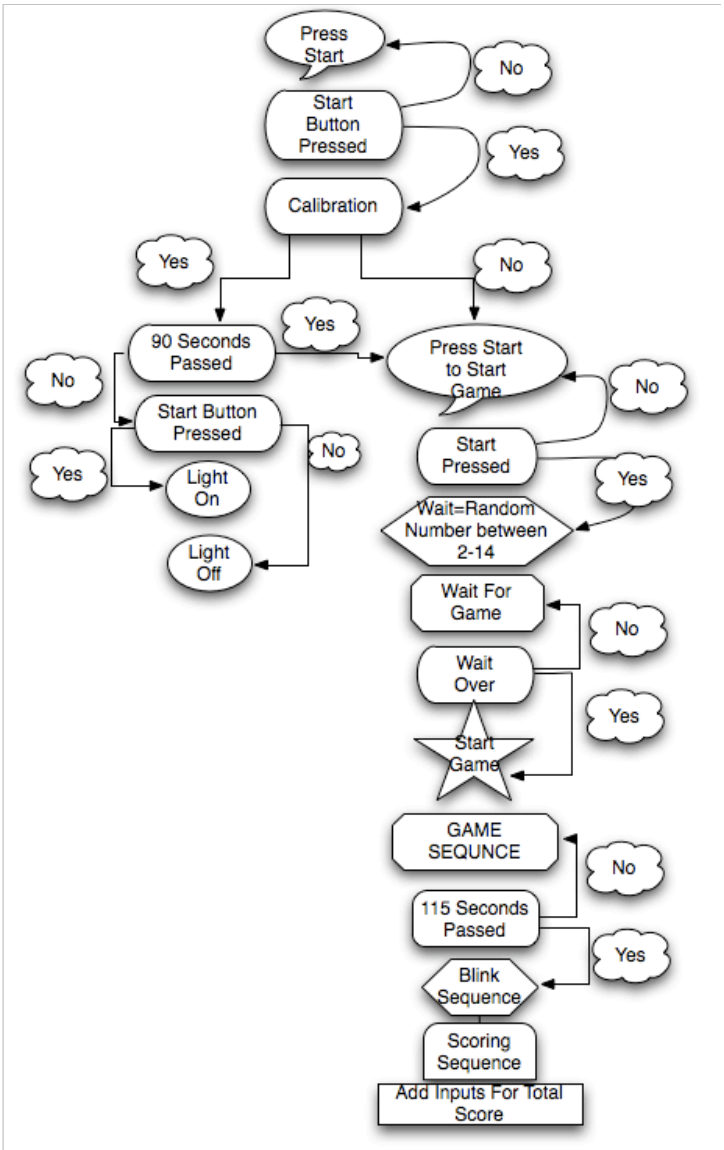
2.4 Advanced Scoring System

Running out of ideas on new features to add to the game sequence, we started thinking about other things we could add in different places to make the code better (and longer). So we thought it would be nice, that after the Handyboard has conducted the game, for it to score the game too! So what we did was utilize the knob on the Handyboard to score the game. This code would consist of the Handyboard asking you how many of different pieces such as cups or poms are on certain places on the board. To make this even more convenient, because we know the Handyboard knob can be so sensitive, the Handyboard will tell you if you have entered an invalid number of pieces (exceeding possible amount) and it also is constantly checking things like whether it needs to ask how many cups are in a shelter, if you have already stated they are all on the board. Now, unbelievably enough, adding in all the changes, the necessary functions, and variables needed for this, the code is now 1030 lines. Keep in mind that the minimum number of lines to make the code work to normal standards is 14 lines.

2.5 Extra Bit of Fun

As an extra bit of fun since there was really nothing else we could possibly put in (mostly because it wouldn't fit) we added a fun little password protection to the Handyboard. We utilized, again, the knob from the Handyboard, as a combo-lock of sorts. We set it so that anytime you want to password protect a feature, like the scoring system, you just put in the function, and it will ask you for a password that is pre-set in the function. If you get the password correct, you get to move onto the item it is protecting, if not then the Handyboard will stop the code, beep and print that the code is wrong. Now, with this fun little feature added in the final code for our Handyboard comes out to a whopping 1124 lines of code. Once again, it only really needs 14 lines to run the game. In essence, we multiplied the amount needed by 80 and made an overcomplicated way of turning on and off lights. After all of this, our code looks like this Figure 4:

Figure 4:



3. Conclusion

This technology does not have to be used for turning on and off lights. Other possible uses include driving high powered electrical motors for use in cars, and yes, controlling stoplights by turning the lights on and off.

References

[1]"Relay." Wikipedia: The Free Encyclopedia. 1 June 2008. 3 June 2006 <http://en.wikipedia.org/wiki/Relay>.

[2] “Ohm’s Law.” the12volt.com. 27 May 2008. <http://www.the12volt.com/ohm/ohmslaw.asp>