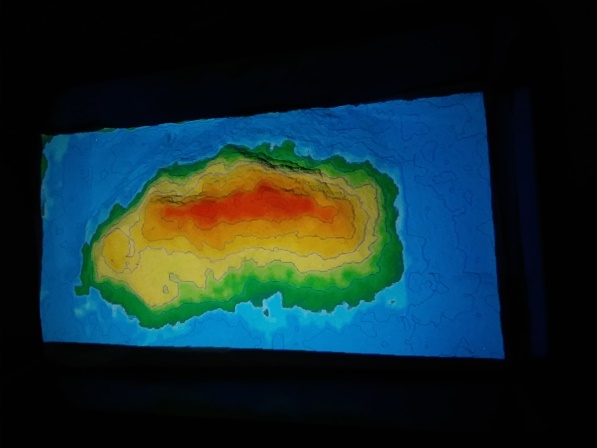
Last year Elena Bachman, Sharon Dempsey and I made the Magic Sandbox, an Augmented Reality Game that applied open source software, Windows Laptop, XBOX Kinect camera and cheap projector to create a colored topographical image onto a sand filled tray that can be manipulated to simulate mountains and oceans based on the height of the sand, as a Maker Faire exhibit. We will show this again at the Maker Faire 2019 at GMU.

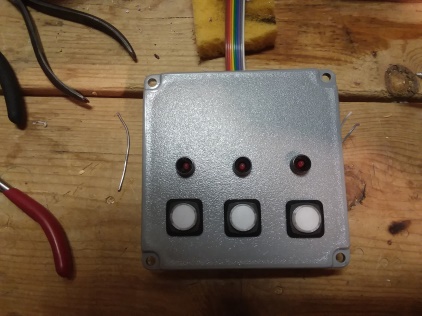
Because some of the games require keyboard manipulation to start the games or change the parameters, I was concerned about sandy fingers getting onto my laptop and into the keyboard. So I decided I needed a way to provide game choices with a separate control that players can access safely.

Magic Sandbox

The Game Software has single keystroke control of the game selections and reset, so I set out to build a box whose buttons sent these characters to the laptop as if it was from a keyboard. I suppose I could have used a separate keyboard but I thought buttons and lights were more fun.

First I needed to learn how to simulate buttons on a USB port. It turns out that some (but not all) Arduino microcontroller boards can emulate a device USB interface, and I would be able to write a program interface to buttons and lights and send characters to the laptop. I also found that other people had implemented a similar function. (See this single button on a box. <http://rasterweb.net/raster/2011/05/09/the-button/> ) I also found there was a library function called not surprisingly, “keyboard.h” This needs to be “included” at the start of the Arduino program. ( i.e. #include <Keyboard.h>)

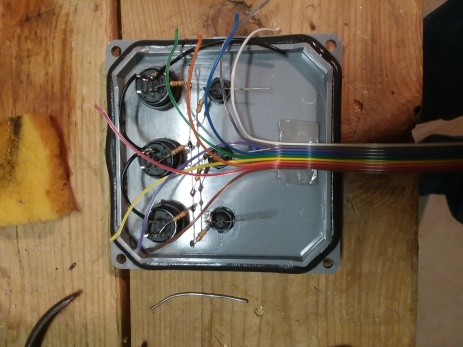
The Arduino Leonardo has the USB functionality and looked like a good fit as I did not need a really small board because I expected the buttons to be fairly big. Then I needed a box. I decided to buy rather than 3D print and I found a nice plastic electrical box at Lowe’s that was about a 4 inch cube.

The characters I wanted to emulate were the M, the F and the R. These would pick 2 games involving animated “fish” and “bunnies” (projected onto the sand) that you could guide by manipulating water and land heights to help them reach their goals and R was the game reset.

Buttons and LEDs mounted

I found 3 short momentary (push to contact) pushbuttons with separately controlled LEDs and 3 small panel mounting LEDs at Mammoth Electronics (a source referenced by the link) which fit nicely across the width of the box in rows of 3.

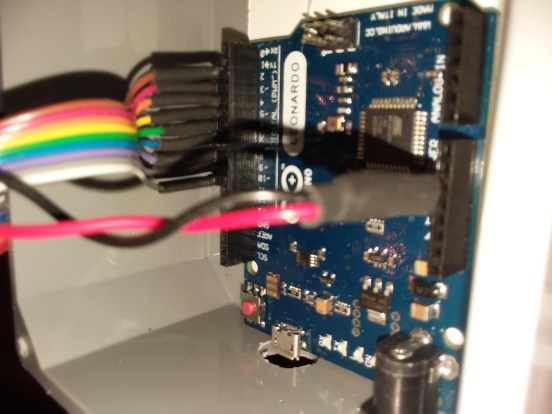
Next, I marked and drilled the box lid and started the assembly. I had some ribbon cable so I decided to use this to keep the wiring neat. I laid in a bus bar for the power for the LEDs and wired in the voltage dropping resistors – to the button and lamp LED terminals, and wired in the switch ground wires. I wired the switches so that when pushed they connected the Arduino input to ground, and planned to use the Arduino internal pin pullup resistor to ensure that when the button was not pushed the inputs were held high.

I could then position the ribbon cable and separate the individual wires and trim them to the length to reach their individual destinations.

Wiring cut and routed

Then I prepared the other end of the ribbon cable and power wiring for plugging into the Arduino board, by soldering the wires onto header pin strips that fit the Arduino and using heat shrink sleeves over the joints.

I was able to locate the Arduino on the bottom of the box, raised up on a foam core board backing, and I drilled a side hole in the box to accommodate the USB connector. Then I attached the backing and Arduino board with double sided sticky tape, and plugged in the header wires.

While this is not a complex project, I wanted to make sure the hardware worked before I wrote the full software. The first thing I did in testing was to verify the LED lights and the Switch LEDs operated and the brightness was appropriate and that the Arduino could read the switches and control the LEDs.

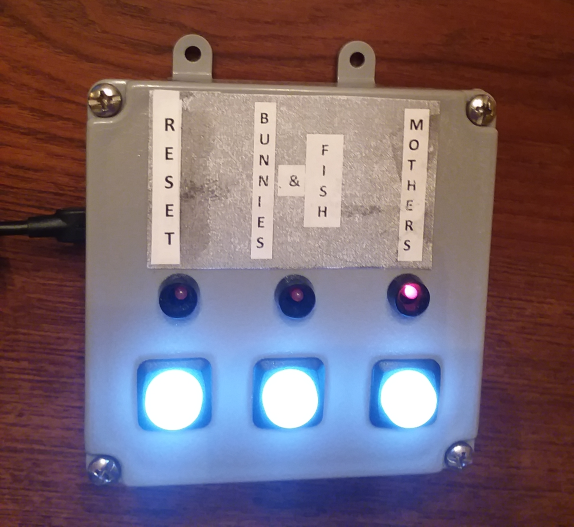
Arduino connections

At that point, I found the white LEDs on the switches were way too bright especially since the Sandbox is used in dim lighting, so I changed the resistors for those LEDs to a higher value. I could have programmed the Arduino to dim the LEDs by using pulse width modulation but I did not think of that at the time.

I wanted to have the operation designed in a way that would discourage “button mashing” so I did several things to discourage that.

First, I used the LED on the Switch to indicate that the game was changing states by flashing the white switch LED on and off until the game had changed state (also trying to discourage more button pushing). Second, I used the red LEDs to indicate what game state the program was in, and only light that LED once the game was in that state. and third, once a button push was detected, I waited so see if it was a sufficiently long push to be intentional, then ignored further pushes until the game had changed state. The red game state LED was positioned above the switch that caused that state. Since the box was emulating a keyboard, I did not get information on game state from the laptop so I waited for a suitable delay instead while the game state was changing. I basically used brute force programming for speed, testing for each button sequentially within the execution loop and did not try to minimize the code.

In testing, I found that because the code sent keyboard characters back, if a program error caused characters to be sent without button pushes, or if buttons were pushed when the program window was active, the characters would embed themselves into the code or open window – so I needed to be careful to open the Monitor window or have a different window active to accept the characters once the code was uploaded and running.



Completed box in action

Then I labelled the button/status combinations for the games.

This button box was used successfully for several public occasions until last time, one button stuck in, surely due to a sand grain getting into it. Subsequently I pried the button back out with knife but in future I will cover the switches with plastic wrap when in use to prevent further sand ingress.



https://drive.google.com/open?id=1dQKuk6bJjF9JIbjcHju9P31oE-xjRBxMD4njPrmVdKo

Sandbox Slides link