GNG 1103

Design Project User Manual

SD Gate 1000

Submitted by:

Project Group A18

Cole Palmer - 300007272

Moiz Adamjee - 300149509

Fadi Oussta - 7958597

Kim Alain Kazenga - 300038697

December 7th, 2019

University of Ottawa

Abstract

We decided to tackle CEED's problem regarding efficiency of the maker space. We have created the SD gate 1000 to serve as a means to that end. The SD gate 1000 is a 3D printer managerial system that is able to show users whether the 3D printers are currently in or out of use with the use of a potentiometer. The product is powered by a NodeMCU and displayed on a Dashboard interface. This product should give a real boost to the productivity and efficiency of the space

Table of Contents

List of Figures

List of Tables

List of Acronyms

Acronym	Definition
CEED	Center for Entrepreneurship and Engineering Design
STEM	Science, Technology, Engineering, Mathematics

1 Introduction

At the start of the 2019 Fall semester, we met with the CEED staff to discuss what they'd like to see changed within the MakerSpace, and the Brunsfield Centre. From this meeting, we observed that the CEED spaces could really use a boost in productivity and efficiency. They were especially focused on the lack of organization surrounding the different machines such as the laser cutters and 3D printers. We wanted to focus more on one individual area of the MakerSpace, rather than attempting to solve issues in both spaces. This way we'd be sure to end up with a more refined and polished prototype. After polling several CEED staff, and STEM students, we decided that the area that was the most in need of an increase in efficiency was the 3D printers. Every day, especially closer to exam season, these printers are constantly in use, but there's currently no way to monitor which ones are in use, and when they'll be available, aside from checking on each individual printer in person. We wanted to make a product that would automatically display which printers are in use, and how long it will be until they're complete, in a clean and easy to read manor. To accomplish this task, we came up with the SD Gate 1000.

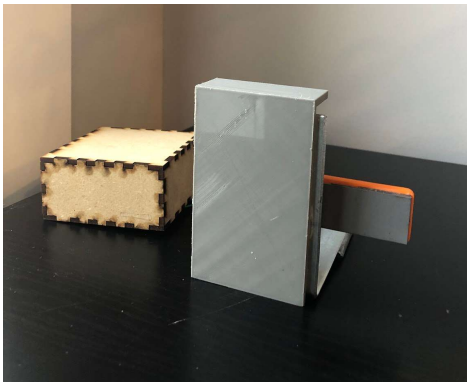

Figure 1 - SD Gate 1000

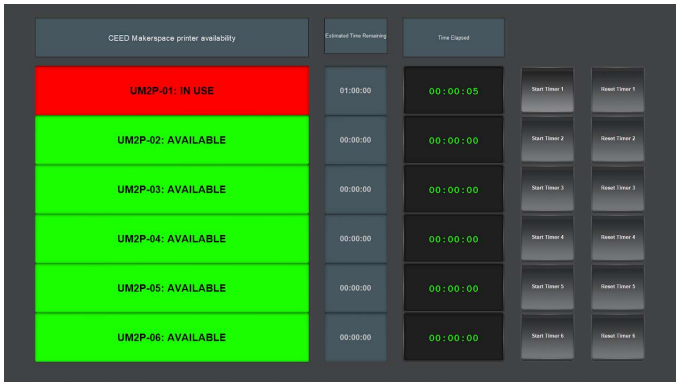

Figure 2 - DashBoard Interface

1.1 How it Works

The SD Gate 1000 will detect whether a 3D printer is in use or not through the use of a potentiometer and the SD card slot. When the 3D printer is not in use, the SD Gate will completely cover the SD card slot. Say a user wanted to print something. They'd need to lift the gate, insert the SD card, then the gate will be held in the "open position". The potentiometer is

then calibrated to send a signal that the printer is in use. Once the print is complete, the SD card is removed, then the gate will shut automatically into the closed position. These results will then be displayed on the DashBoard interface. Finally, the user will have the option to enter the approximate time remaining on the print, as start a timer to show the time elapsed since it started.

1.2 What Sets it Apart

Our product stands apart from the competition for 2 main reasons. Firstly is the DashBoard design. The goal was to create an interface that anyone could understand at a second's glance. It has both text, and colour coordination so even those that are colour blind are able to make use of this product. In the future, there could be an option to have the interface be bilingual as the University of Ottawa is a fully bilingual school. The main thing that separates this design from others however is the choice of sensor. Many similar products rely on either video, motion, or touch sensors on the printer. Through various tests, we found that the motion of the printers was too inconsistent, and sensors such as PIR, ultrasonic, or magnetic seemed to be somewhat unreliable. In the case of the SD Gate 1000, there's no way for the potentiometer to receive an incorrect reading so long as it's calibrated properly.

2 How the Prototype is Made

This product was constructed as a makeup of 3 subsystems. Firstly, is the mechanical subsystem which focuses on the gate design. Next is the electrical side which focuses on the wiring of the NodeMCU to the sensor (potentiometer). Finally, is the software subsystem which focuses on the coding as well as the DashBoard design. Each subsystem is analyzed in further detail below.

The mechanical system consists of a 3D cover, L-shaped hinge, the gate and metal scraps. The L-shaped hinge is glued to the 3D printed box so that the box can function standing vertically. The gate was made with a hole with the same shape as the potentiometer head. Then, the potentiometer was glued on the hinge in a way that the gate can be connected to it. The design revolves around one idea that after the door is opened it should be able to close by itself. To attain

that, many small metal scraps were assembled and put on the back of the gate so that when opened it could close itself due to their weights.

2.1 Mechanical

2.1.1 BOM (Bill of Materials)

Part	Price
Potentiometer	Free
L-Bracket	$3
3D printer gate	Free
3D printed cover	Free
Super glue	$2
Metal scraps	Free
Total	$9

Table 1 - Mechanical BOM

2.1.2 Equipment list

- Clamps
- Metal file
- 3D printer

2.1.3 Instructions

1. Print the gate and cover pieces in a 3D printer using the provided files.

2. Super glue the L-bracket onto the left side of the extrusion of the cover piece. Clamp to dry.

3. Measure the distance up from the surface to the SD card slot on the Ultimaker 2+ 3D printer. Super glue the potentiometer to the L-bracket at that same height (be sure not to get any glue on the pins of the potentiometer).

4. Glue the gate piece onto the end of the potentiometer through the hole in the center. If necessary, file down the plastic end of the potentiometer so it's completely flush with the gate piece.

5. Glue as many 2cm by 4cm scrap aluminum pieces as needed to the end of the gate so that it will shut naturally when let go (depends on the resistance of the potentiometer but should take around 3 pieces).

6. Take another 2 cm by 4 cm scrap piece of aluminum and bend into an L-shape. Glue this piece at the top of the L-bracket so that the edge of the aluminum scrap will catch the interior edge of the door and hold it in a 90 degree position.

Figure 3 - Gate Mechanism

2.2 Electrical

BOM (Bill of Materials)

Part	Price
NodeMCU	$9
Wires	$2
Micro USB Cable	$4
100 Ohm Resistor	Free
LED	Free
Breadboard (for testing)	$10
Protoboard	Free
Solder	Free
Total	$25

Table 2 - Electrical BOM

Equipment list

- Soldering iron

Instructions

1. Assemble the circuit below first in a breadboard to ensure that there are no faulty components.

2. Once each component has been checked, assemble the circuit into the protoboard and solder all the connections in place.

3. Be sure that the positive wire is attached to the positive lead of the potentiometer (and the same for the negative). For this potentiometer, the positive lead is on the right, the negative lead is on the left, and the data is in the center.

4. Using a small piece of tape, secure the plastic case of the wires to the L-bracket to ensure the connections don't slip off.

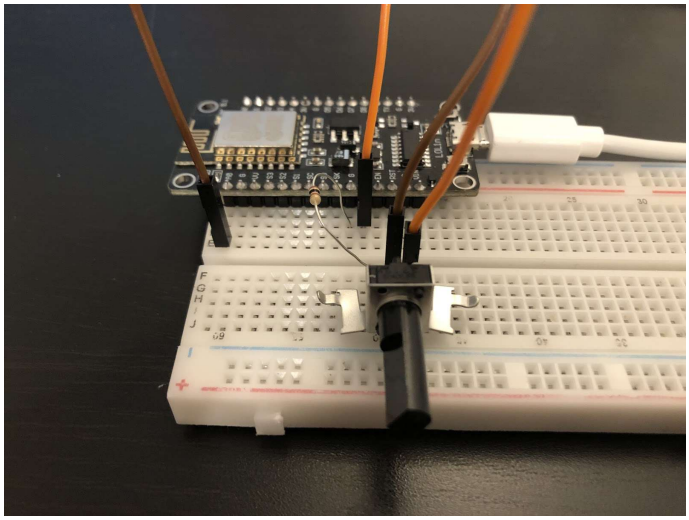

Figure 4 - Circuit Diagram

Note: The brown wire goes from the center pin of the potentiometer to the A0 pin of the board, the orange wire goes from the positive lead to the 3.3V pin, and the resistor goes from the negative lead the the G pin.

2.3 Software

BOM (Bill of Materials)

There was no software to purchase. The only “materials” for this subsystem are the free Arduino and DashBoard softwares.

Equipment list

- PC Computer

Instructions

1. Assemble the DashBoard shown below. Each column is made using a 6x1 simple grid. The printer labels are read-only labels with a parameter id, the ETR labels are text entry, and the timers are read-only labels set as clocks.

2. Call the oid of the first label “printer1” and the id “changecolour1”

3. Set backgrounds, fonts, and alignments to whatever is most visually pleasing. This DashBoard used grey background, “bigger” font size, “bold” font and center allignment.

4. Set up a listener. Make sure to check start automatically, set the port as ‘12345’ and set the delimiter type as ‘new line’.

5. Add a task in the listener and copy the code below into the software.

6. Moving on to the Arduino IDE. Make sure the NodeMCU ESP8266 libraries are installed. The instructions for how to do so are in appendix I.

7. Open up the “Wifi client basic” example in the Arduino IDE. There is a space in the top of the code to enter the ssid, password, and ip address of a wifi network. Enter the wifi

information of the computer hotspot off the computer that will be running the program. Also, enter the port number from DashBoard (12345). Finally, change all ‘delay(5000)’ to ‘delay(2000)’ so that the program will check the status every 2 seconds instead of 5.

8. After the } of the if statement “if(!client.connect(host,port))” enter the Arduino code noted in appendix I. Note: The sensor value of the “open” position for us was calibrated at 700, however this may differ depending on the model of potentiometer, and it’s exact placement. Be sure to calibrate the potentiometer for each individual iteration of this design.

9. Compile and upload the sketch to the NodeMCU, and the program will start automatically. Open the serial monitor to check on the status of the connection (make sure the baud rate is set in the serial monitor to whatever it is in the code (115200 or 9600)).

3 How to Use the Prototype

1. Place the L-bracket flush against the printer such that the gate is covering the SD card slot
2. Secure the product in the desired spot
3. Lift gate up to expose the SD card slot
4. Insert SD card and let go of the gate
5. After printing is complete, remove the SD card and the gate should drop to its original position

Figure 5 - Closed Position

Figure 6 - Open Position

4 How to Maintain the Prototype

The most important aspect this hinge has to live up to, is durability. For testing this, we moved the hinge up and then released it so that it could go back into its original position 100 times. If the hinge breaks during this test, the design would have to be redone again reinforcing the spring. There was a risk of the gate breaking since most of the parts are glued together using super glue. In addition, the gate was placed in the open position and left like that for 2 hours (average printing time). These tests had 0 effect on the gate's durability. So, we can confidently say that the gate can handle extensive and more heavy usage.

If for some reason there was no change in the message on dashboard after moving the gate, the likely culprit would be loose wiring in the arduino. So, maintenance of the arduino wiring could be required.

None of the parts are particularly prone to breaking, unless an individual specifically wanted to break it. In which case, the part that would like break would be the potentiometer or the gate itself. Fortunately, both these components are easily replaceable.

5 Conclusions et Recommendations for Future Work

In summary, we were able to create a comprehensive prototype with a functional hinge mechanism and Arduino-Dashboard integration. The product is able to successfully indicate if a printer is in/out of use. For future work, we would like to be able to integrate dashboard on a webpage that would allow users to check the status of prints from off campus. Also, we would like to incorporate more printers into the system as well as making our dashboard interface more user-friendly.

6 References

Knox, D. (2019) *GNG 1103 Lecture Slides and Project Instructions.* Retrieved from: https://uottawa.brightspace.com/d2l/le/content/116873/Home

APPENDICES

APPENDIX I: Additional Files and Images

All design files are also located on MakerRepo. The link for which is below.

https://makerepo.com/Madam056/group-a18-3d-printer-online-management

```
if(event.getEventType() == 1){
   var rawData=event.getBytesAsString();
   ogscript.debug(rawData);
   params.setValue('printer1',0,rawData);
   if(rawData == "UM2P-01: AVAILABLE"){
      ogscript.setStyle('changecolours', "bg#00FF00");
   }else(){
      ogscript.setStyle('changecolours', "bg#FF0000");
   }
}
```

DashBoard Listener Code

```
int sensorValue = analogRead(A0);
Serial.println(sensorValue);
if(sensorValue <= 700){
  client.print("UM2P-01: IN USE");
}else{
  client.print("UM2P-01: AVAILABLE");
}
client.print("\n");
```

Arduino Code

You will also need to install the proper board libraries on Arduino. To do this:

a) Go to File > Preferences and in to the area beside "Additional Boards Manager URLs" enter: http://arduino.esp8266.com/stable/package_esp8266com_index.json

b) Go to Tools > Boards > Board Manager. In the manager search for "esp8266" and install the "esp8266 by ESP8266 Community" library.

ESP8266 Library Instructions