Vending Machine Design Report

Ismael Diaz, Aashika Jagadeesh

ECE 350: Digital Systems

Table O' Contents

Overall Project
- MVP
- Mechanical Structure

I/O

Processor Modifications

Challenges
- Wiring
- I2C
- *insert other challenges here*

Testing

Assembly Code Overview

Improvements

Appendix A: Final Result Pictures

Appendix B: CAD Model Pictures

Overall Project

The goal of this project as outlined in the original project proposal was to “design and develop a **small snack vending machine** with our processor”. The proposal also outlined the following computational components that needed to be implemented as guidelines towards our MVP.

MVP

Our MVP was to get a vending machine that could release a snack to the user. To reach our MVP the following components needed to be added:

- Stepper motor spinning coils
- Input to select items to be dispensed
- LED Strip for lighting up the snacks
- Speaker to beep when a snack was released
- Payment system
- Display showing price and selected snacks

We also outlined POST-MVP items:

- IR Sensor for snack falling detection.
- Servos to move at more precise angles
- LCD display / VGA / Seven-segment display
- Currency system with NFC Reader (our own “food points” system)

In our vending machine, we were able to implement stepper motors, buttons for user input, an LED strip, a speaker to indicate a snack dispensing, and a payment system. We were fortunately able to implement an IR sensor for snack dispensing detection as a Post-MVP item instead of hard-coding the number of stepper motor rotations before dispensing a snack. As another post-MVP item, we wanted to add a VGA display or LCD to display to users the price of an item selected. However, we soon discovered that an LCD requires the I2C communication protocol or with a seven-segment display, we would need to use 14 cables, which was more then the available PMOD pins on the FPGA. The 3-point system was a little ambitious given that we would have needed to implement the currency system to know which users actually needed to be stopped from purchasing more items. We researched achieving LCD communication with an FPGA using a finite state machine, but we were not able to debug the issues with slave device addressing on our FPGA in time.

Despite not being able to complete those Post-MVP items we did have a successfully working slot for our vending machine. The machine was able to work from the push of the button to the retrieval of a snack (and it could be used as many times as needed!). See Appendix A for the final result.

While this project showcases the processor and circuit board design, we also heavily relied on various mechanical components to dispense snacks to successfully users.

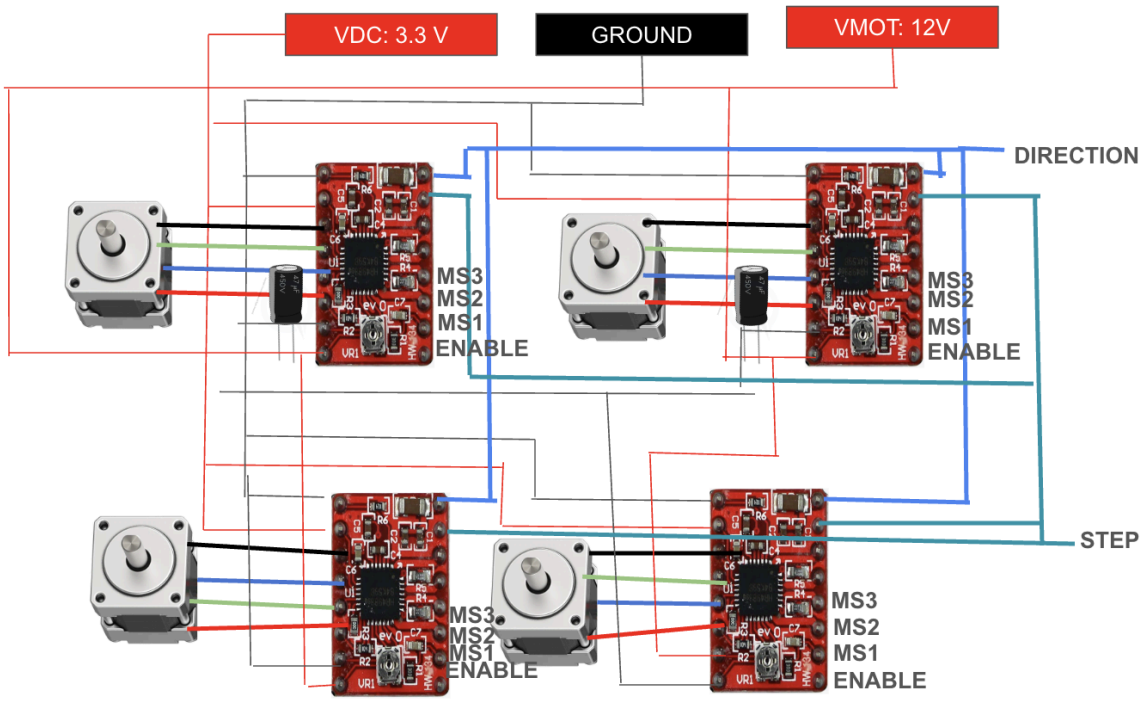
Mechanical Structure

We opted for a 2ft by 2ft box that would hold 4 slots and house all electronic components without it becoming a cramped space to work with. The following CAD models needed to be designed:

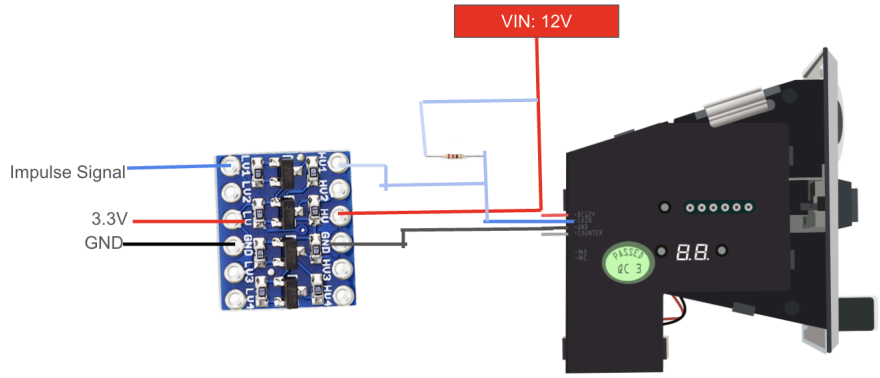
- Coil: the main delivery system of the snack. (See Appendix B: Figure 1)
- Snack Container: The place where the motors, coils, and snack are housed, along with a sliding out mechanism for easier restocking. (See Appendix B: Figure 2)
- Stepper Motor-Coil Insert: Needed a way for the stepper motor and coil to attach to each other, so this was definitely a must needed 3d print. (See Appendix B: Figure 3)

Circuit Diagram

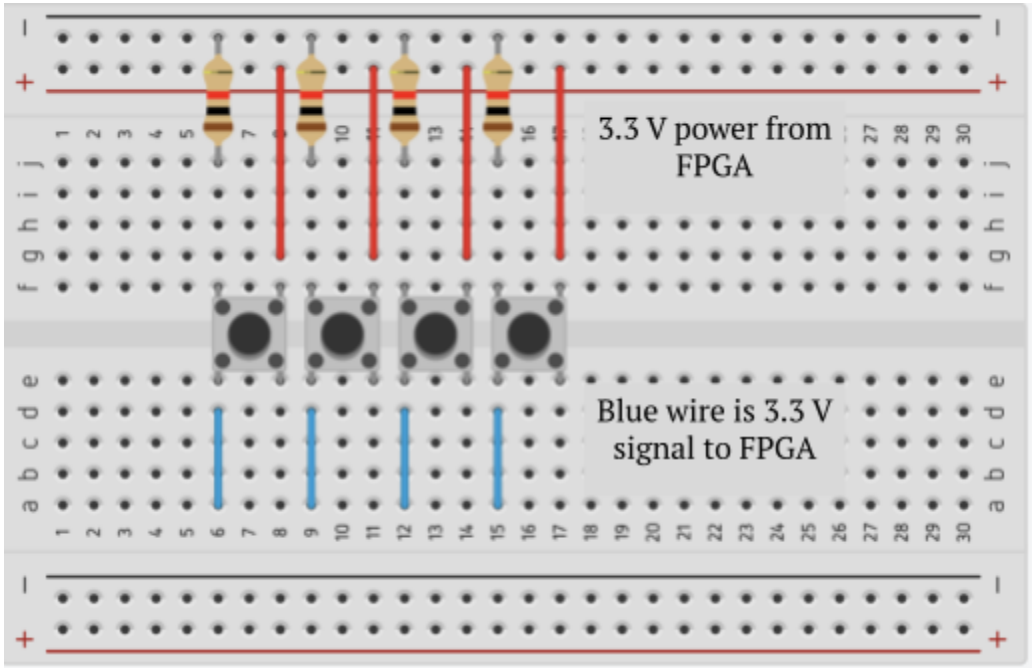
Motor Controller Circuit



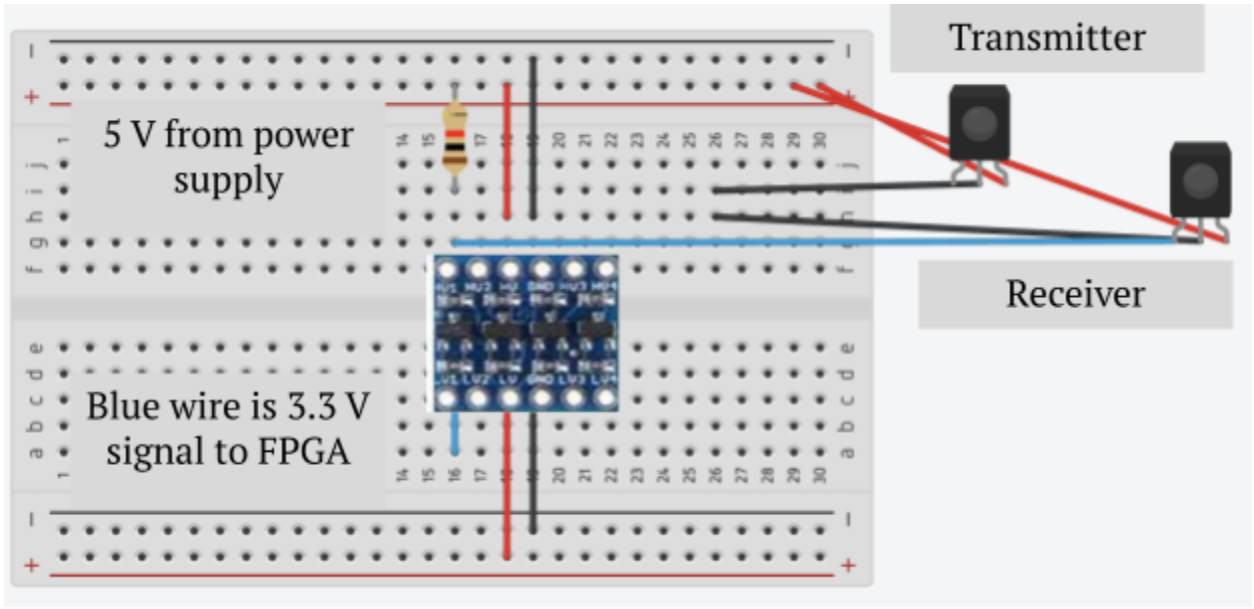
Coin Slot Circuit



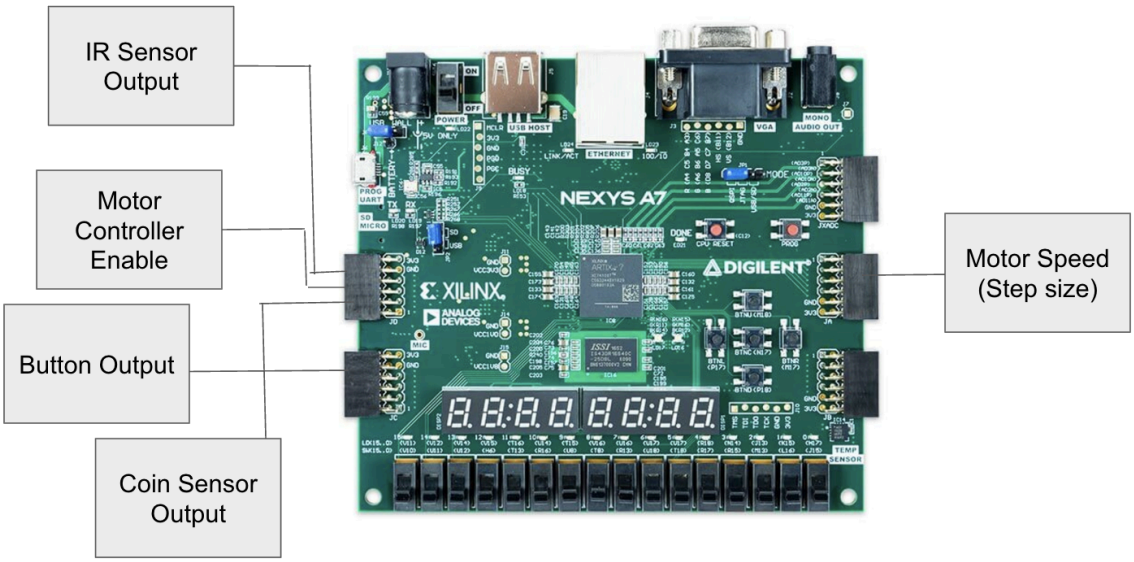
Push Button Circuit



IR Sensor Circuit



FPGA Connections



I/O

- 4 push buttons for simple snack selection
- Speaker output to detect that enough money was inputted into the coin slot

Processor Modifications

There were no processor modifications necessary for the successful implementation of this project. It would have been nice to have a move instruction, however, as it was definitely something I had to do often.

Challenges

A myriad of challenges presented themselves including (but not limited to):

Wire Management / High Current Draw

We did not properly implement wire management techniques such as grouping wires together and strain relief. Initially, we implemented a perf board for the motor controllers, which is why solder traces were made to connect the motor controller board to the other connections on the board. However, this made it easier to short connections from the lack of insulation. We also used low gauge wires at times for easier soldering, but with a motor controller board using four motors, low gauge wire would not be able to support high current draw.

Solid core wires in screw hole terminals were not the best idea for stable connections, as it may have been better to use frayed wires in JST connectors. Due to many of these wire management issues, we had to debug various loose connections. A protoboard may have been better as well because it includes in-buit connections by row, especially for the A4988 motor controller connections. We faced some issues where an individual motor would be on, use 6V and max current (0.2 A). It was hard to understand where this issue came from because there were no clear shorts on the motor controller board between VMOTOR or VDC and ground. We believe a possible source of this issue was the voltage stability, since not all four motor controllers were using their own bulk capacitor.

I2C

We ran into issues with debugging the LCD display, as it was not initializing properly. We tried almost everything, from changing the address we would write to, to running test benches, to adding pull up resistors.

Timing

When trying to read signals from the coin slot, we realized that the coin slot signals had a 50% duty cycle with a period of 100 ms. To resolve conflicts between a 44 MHz processor and 100 ms period, we counted the number of CPU clock cycles in behavioral verilog till all of the impulses were sent. In order to detect an initial impulse, we detected a negative edge of the coin slot signal.

Testing

The only significant testing was ensuring that the single slot was working effectively. We ran it over 15 times testing ensuring the LEDs on the FPGA were lighting up to indicate that:

a) A button was pressed
b) The amount of money in cents was displayed in binary
c) There is a sufficient amount of funds

In a way, the onboard LEDs on the FPGA acted as outputs that helped us visually indicate where in the assembly code we were. That really helped debug a multitude of issues such as whether or not the buttons were all working, whether our coin slot was adding up the right amounts of money, and whether or not we set the right price threshold.

Assembly Code Overview

The assembly code (linked here) can be narrowed down to 3 major chunks. The first is button reading, which utilizes memory mapped I/O, to read the current state of the button inputs and saves the price associated with that button. The next section is the impulse reader. The behavior verilog in the Wrapper takes care of catching all the impulses and the assembly takes care of identifying which impulses map to which coin in USD. If a user changes their mind, they can always press another button as the loop checks for new user input and the total amount of money put in by the user is constantly being checked to see when it needs to branch to the vending state. The final vending state, plays a sound on the speaker and vends the item until the IR sensor picks up the signal that an item was dispensed. Once this all occurs, the vending machine resets back to the restart label and begins this process again!

Improvements

- Creating a PCB for the motor controller board for no loose connectivity issues
- Using JST connectors with frayed wire instead of screw hole terminals
- Preventing a release of a snack if not enough money is inputted into the coin slot
- Creating a separate slot so the user can receive the snack
- Covering exposed wires

Appendix A: Final Result Pictures

Appendix B: CAD Model Pictures

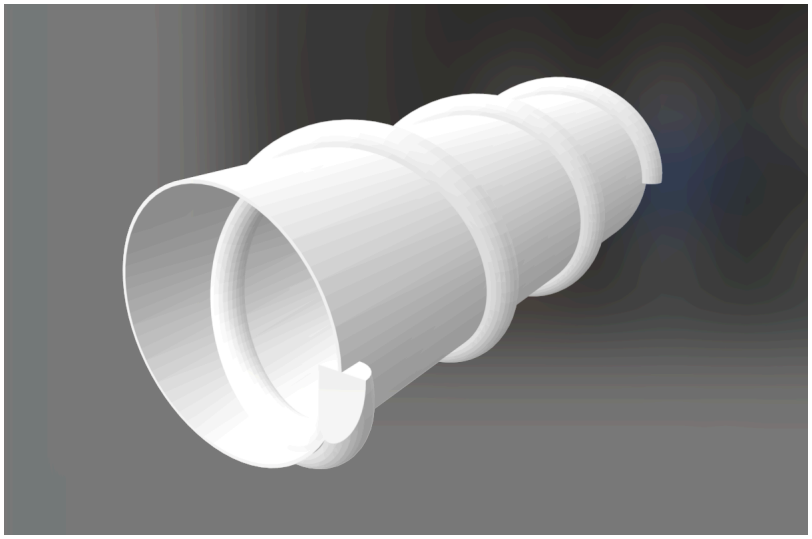


Figure 1: The coil was printed in such a way so that it had supports going all the way up. It was a pain to post-process but overall simple enough and quick to print.

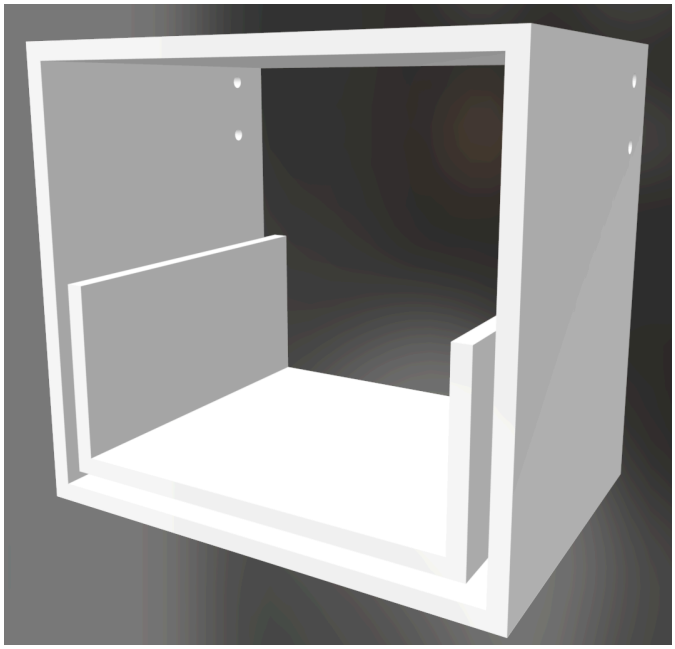


Figure 2: The container which would be printed as two parts and later glued together as one. The holes on the side were meant as places for drilling but was ultimately repurposed for wiring.

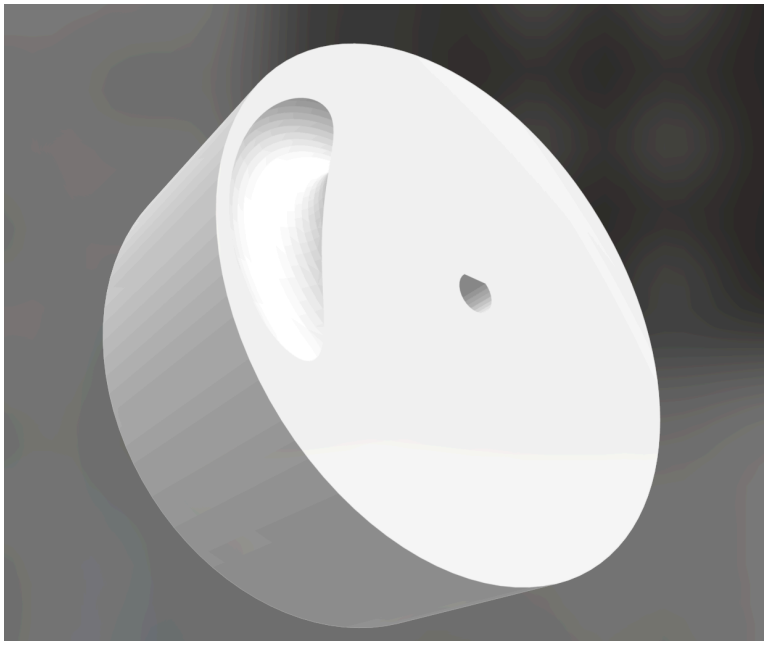


Figure 3: The coil would wrap around a half revolution into the insert and the hole in the middle is for the stepper motor to slide in and rotate the coil.