1D Project Individual Logbook

Electronic Game Prototype

Team Number: 03-2

Name: James Raphael Tiovalen

Table of Contents

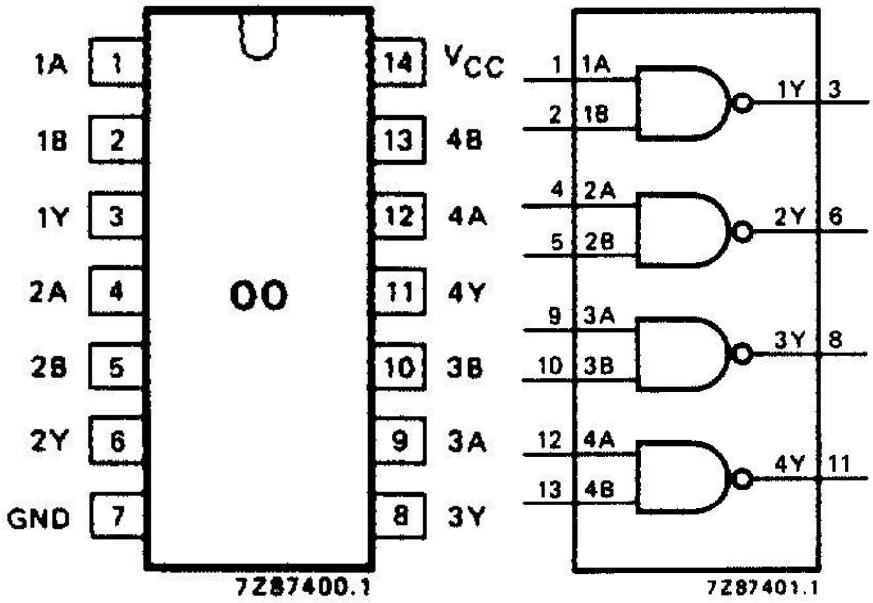
Week 1-2 Entry

Introduction

In the first two weeks, I learned more about how to use logic gates and how a basic adder works, which is useful for our MHP.

Personal Update: Introduction to Logic Gates (17 September 2020)

In Week 1 Lab, I learned how to configure terminals of the various logic gates for our MHP. This is useful for our MHP project because we would be required to assemble and solder a working prototype of a basic 4-bit adder using the gates. I learned about the specific arrangement of the terminals with respect to the notch located on the logic gate:



We used Multisim in the Digital Systems Lab to send test input signals to the logic gates and check the output signals. This way, we can quickly test the different possible logic gates provided to us.

Another new thing that I learned was the fact that we should connect pull-down resistors so as to ensure that a proper “0” signal is supplied when the switch is turned off. I was aware of the physics behind potential difference and all that, but it did not dawn upon me until Professor Natalie mentioned it during Lab. After she explained the reasoning, it makes perfect sense (assuming everything connected to the ground is connected to a common ground, of course).

Personal Update: Introduction to MOSFETs (24 September 2020)

In Week 2 Lab, we were tasked to design a 1-bit full adder in JSim. We built the full adder from low-level NFETs and PFETs. We learned that when dealing with hardware description languages, we should create our circuit in a modular fashion and test every single part of the circuit to ensure that whenever something goes wrong, it is not too difficult to debug. This is relevant to our MHP when we program our FPGA module.

One interesting thing that I learned was that MOSFETs also have limitations (it makes sense, but I could not think of any limitations until it was explained during lessons). MOSFETs are susceptible to overload voltages. If the voltage difference is very high, the depletion region could also “skew” over to create an uneven “triangular”-shaped path where the electrons can flow. However, since only one of the triangle’s points touches the connection between the drain and the source, it becomes a failure point/bottleneck since the resistance would greatly increase and eventually no current could flow, causing the MOSFET to become non-functional. We need to be aware of this when we design our circuit for MHP.

Reflection and Summary

I learned about the basic requirements of logic gate circuitry, as well as how to design a simple 1-bit adder, which we can just chain to 4-bits for MHP.

Week 3-4 Entry

Introduction

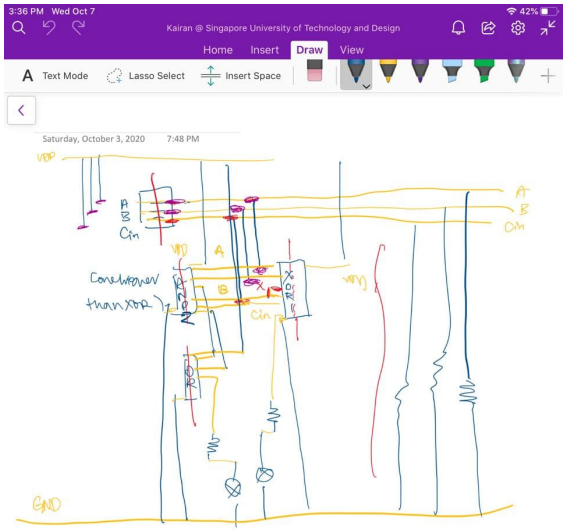
I learned about some basic soldering skills, some basic FPGA programming and how to create a tester for our adder module using Lucid. We also managed to submit our MHP just in time.

Personal Update: Introduction to Ripple Carry Adder (1 October 2020)

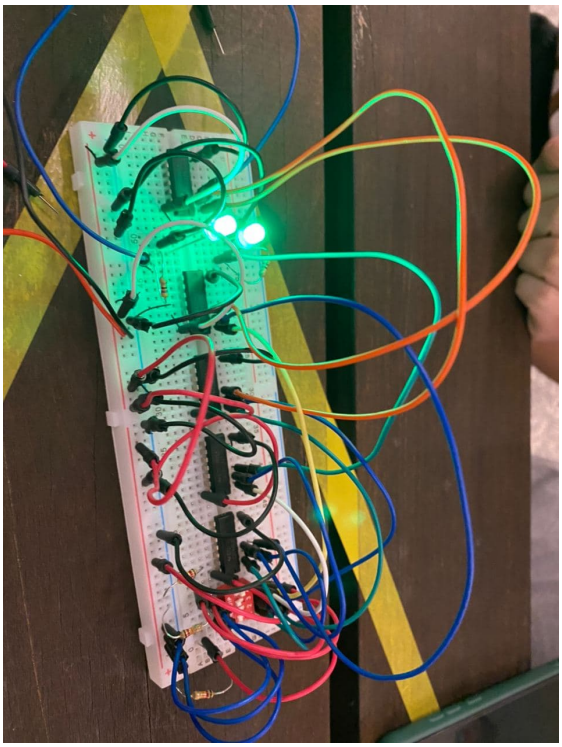
In the lab session, I learned how to build a 4-bit Ripple Carry Adder in JSim. This is useful for our MHP since we need to translate this design process into the hardware adder and the FPGA tester module. Again, emphasis was placed on ensuring that our adder design is modular so that our debugging process would be smoother. This is especially important since we need to design the XOR and AND gates used by the full adder in JSim as well. We also learned how to measure and interpret analogue signal levels using JSim. This way, we can test our design using JSim first before implementing it in hardware and as a FPGA module.

Team Update: First MHP Meeting (3 October 2020)

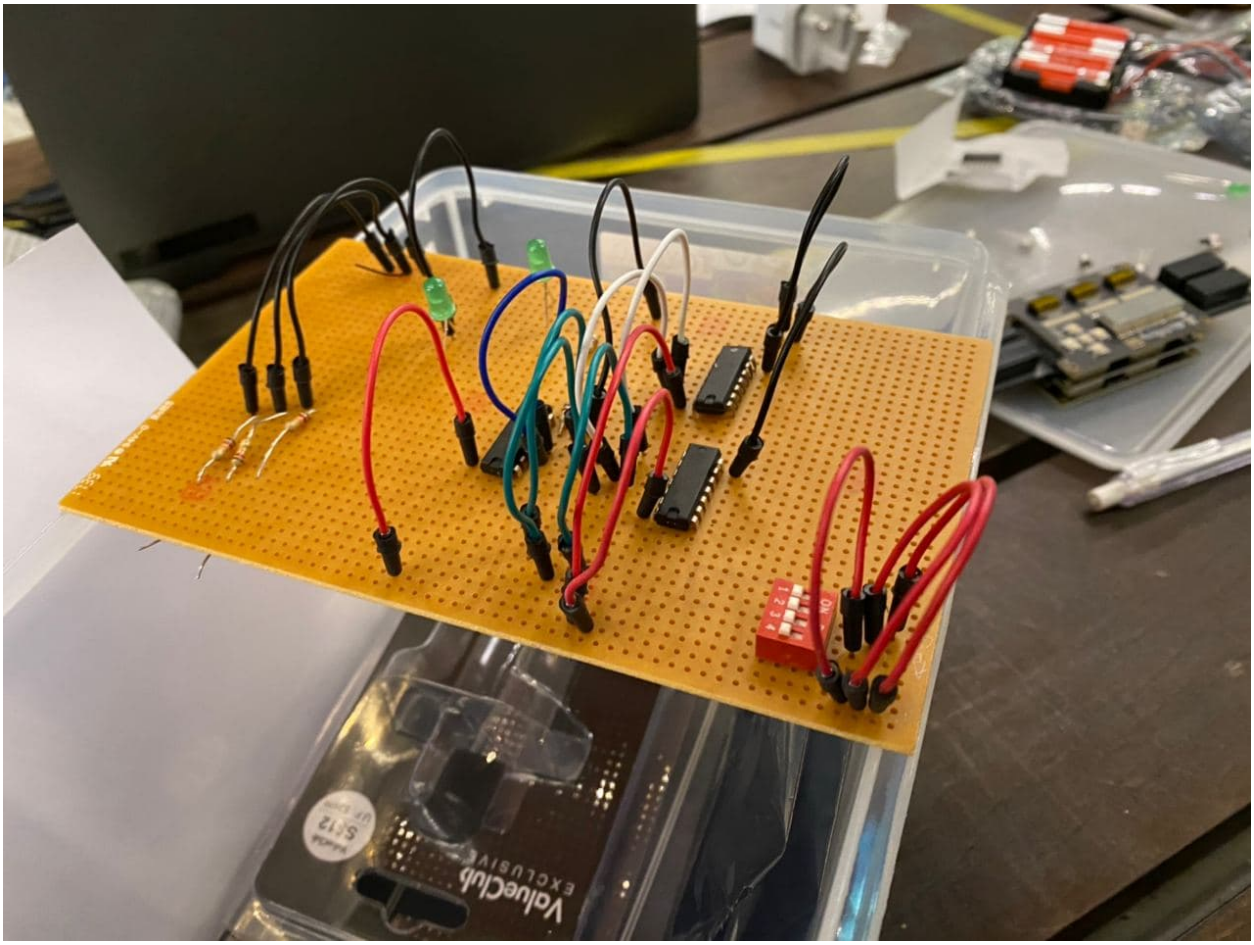
My team met up to get to know each other and start our work on the MHP. I already know 2 of my team members since they were from the same Freshmore class as myself. We met at the benches just outside LT1. I managed to source some of the resources needed for free (such as the solder material and some extra wires). We started with drafting the schematics of the 4-bit adder.



We also tried to figure out and catch the possible connection issues by drafting the connections on a breadboard.



After that, we tried to organize the components on the perfboard first before soldering them (it is very messy):



We managed to solder some of the gates and the wires onto the perfboard provided to us. One issue that we quite quickly identified was that the solder tends to spread out quite easily. This is due to the fact that the perfboard has these horizontal railings/"ridges"/tracks. As such, the solder quite easily spills over within the same track. We faced some challenges with the solder, and we decided to pause the work since it was getting late.

Team Update: Soldering Issues (5 October 2020)

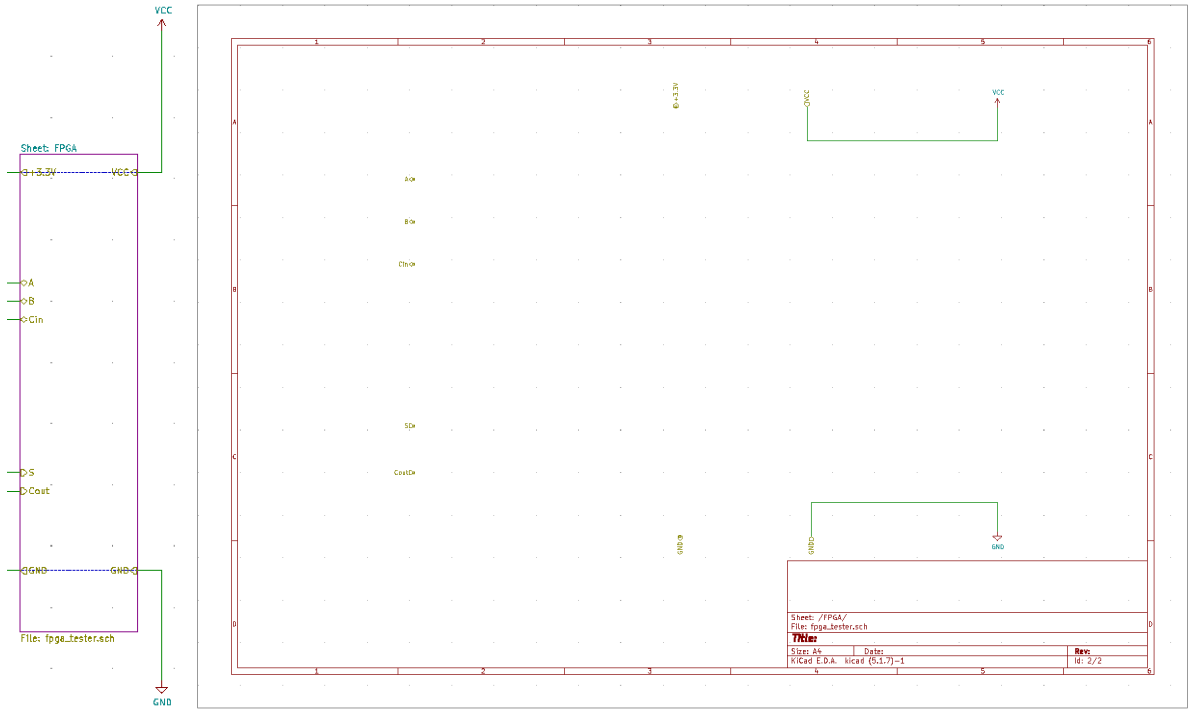
This is a short update. My team met at the benches in front of LT1 again. After trying our best to solder all of the parts, we tested for connectivity issues using a multimeter. The multimeter emitted some weird warnings about some high voltage issues, and we were a little worried (we did not know yet that we should not use a multimeter to debug logic-based circuits). Thus, we decided to scrap the old circuit (since the soldering work was quite sloppy as well) and remake the circuit on the second perfboard that we had. After some soldering work, we needed to desolder some of the components since we made some mistakes, and hence we decided that we would continue to work on it in the Fab Lab next time so that we could use the desoldering pump.

Personal Update: FPGA Build & KiCad Issue (6 October 2020)

Just a quick update. I started some work on the FPGA programming bonus part, but I was not able to build my code on my computer. It was stuck in the synth_1 part:

I asked Professor Natalie about this issue. I even reinstalled the whole Xilinx Vivado package and Alchitry IDE to no avail. In the end, Professor Natalie was kind enough to help me check whether our group's FPGA was fine or not (it was) and lend me her old laptop (which was demonstrably able to flash the code onto the FPGA) for me to work on for both MHP and EHP. She also suggested some items to buy for our 1D project.

I also tried to draw the schematic diagram using KiCad, but I realized that I cannot draw block diagrams in KiCad (without using hierarchical sheets). As such, I created a placeholder sheet for the FPGA's internal testing circuit (with placeholder entry and exit points), while the main adder's circuit was placed on the main sheet. I asked Professor Natalie and she informed me that she used vector-based software to manually draw the block diagram (which means that it could be done in Adobe Illustrator).



Team Update: Desoldering Session (7 October 2020)

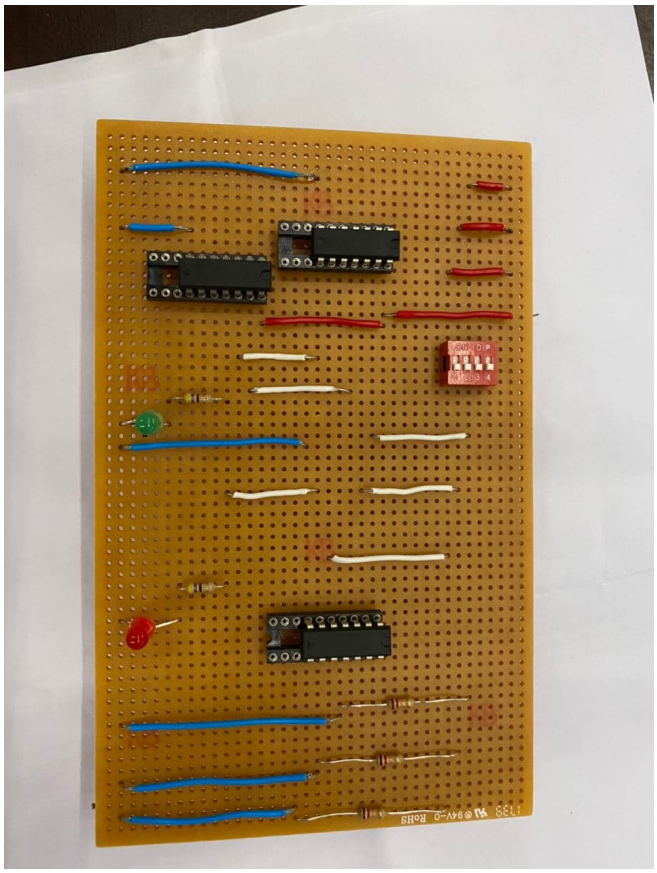
My team met in SUTD Fab Lab to continue the work on our MHP. We needed to use the desoldering pump available in Fab Lab. We managed to desolder the components that we needed. We also improved the modularity of our circuit by using IC sockets for the gates (courtesy of Professor Natalie's suggestion), so that we do not damage the terminals of the gates by directly soldering them onto the perfboard. We also cut these protruding parts using the tools in Fab Lab:

We managed to get the whole circuit properly soldered, but the logic seems to be incorrect. We re-checked our circuit diagram draft, but it was confirmed to be correct. We checked for any connectivity issues, but we could find any. We decided to call the day off since the Fab Lab is closing. We decided that we would consult Professor Natalie the next day to iron out any issues.

Team Update: Successful Debugging (8 October 2020)

In the lab session, we slowly went up the abstraction layer. During this lab, we were required to build a complete ALU and we discussed the different parts that make up the ALU and how to build each part and connect them together.

After the lab session, Professor Natalie helped us with our MHP circuitry in the Digital Systems Lab. I learned about the disadvantages and limitations of using a multimeter. A multimeter is only able to check for connectivity issues, but it would not be able to check if the logic implemented using the logic gates is correct. In fact, since the MOSFETs have separated “gates”, using a multimeter to debug logic gates can lead to some confusion. As such, Professor Natalie demonstrated how to debug using LEDs instead. True enough, we had a connection issue with the XOR logic gate where the input and output were accidentally connected together. After we scraped off the connection, the logic worked fine and thus our hardware circuit was complete. This is our final product:



Personal Update: Alchitry Br Pin Layout Issue (10 October 2020)

I learned about some basic FPGA programming as I tried to implement the adder testing module using Lucid. I faced some difficulties since the logic that I implemented was slightly awry (it was an issue related to the custom Alchitry Br “constraint.ucf” pin layout), but after some painstaking hours of debugging and reading the provided documentation of the custom Br’s pin layout, I managed to get it to work properly. In the end, I decided to go with the default pin layout (instead of creating my own “custom.ucf” file).

Reflection and Summary

I learned more about some possible hardware-related issues and how to debug them properly. I also learned about some basic FPGA programming.

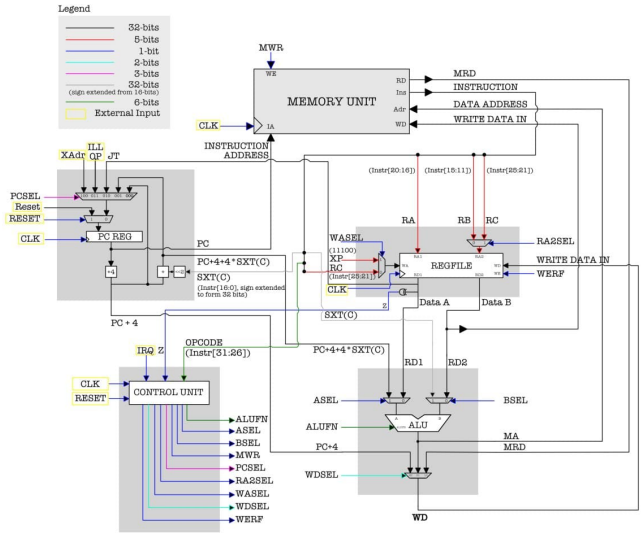
Week 5-6 Entry

Introduction

I learned about how to assemble and connect a basic CPU architecture through the datapath circuit diagram and how the components work with each other. Since the 2D project also touches on optimizing the adder architecture, I also learned a lot about methods to optimize the adder and the multiplier.

Personal Update: Datapath & Adder Optimization (15 October 2020)

Firstly, I learned how to design a datapath for a basic CPU architecture, which is useful for our game CPU's datapath. For example, this is a Beta's datapath:



For these two weeks, we were focusing on optimizing the circuit for the ALU. For the first part, I learned about the different ways to implement a full adder. I found this list of optimized 1-bit full adder designs in a textbook:

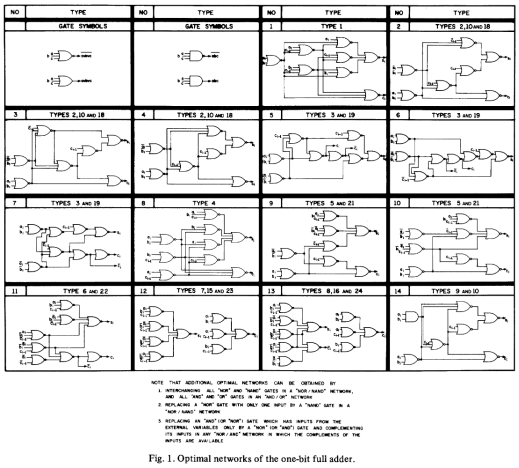


Fig. 1. Optimal networks of the one-bit full adder.

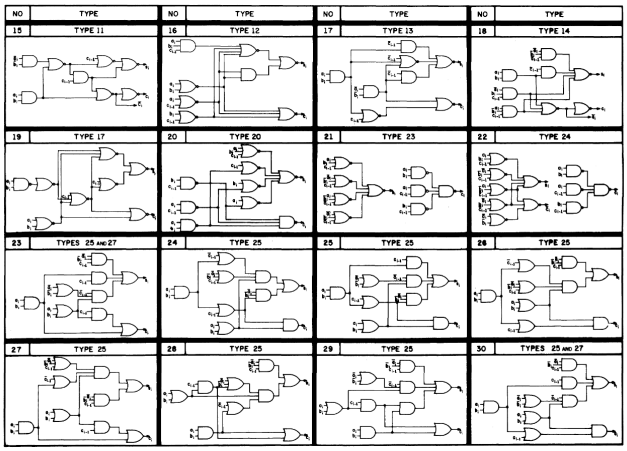
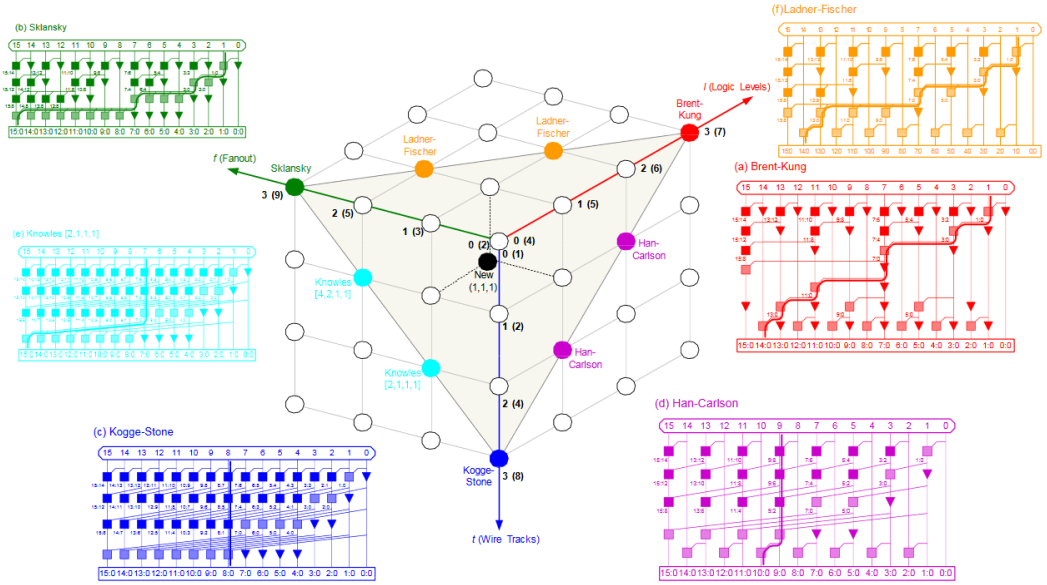


Fig. 1. Continued.

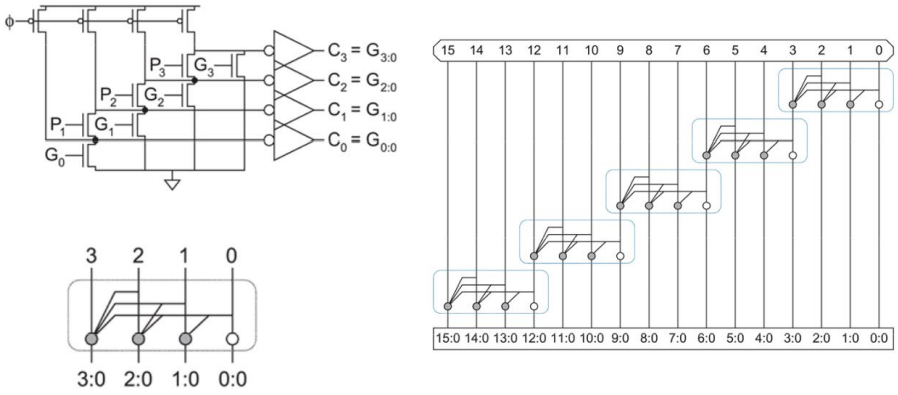
I learned about generating propagate and generate bits for the adder to make it faster since we do not need to wait for the carry bits to propagate. This allows us to design the Carry Lookahead Adder (CLA). This idea of saving time at the cost of more hardware and more area is applicable in many areas. This concept used in designing the CLA allowed us to understand flavours of the carry-lookahead concept, such as the Kogge-Stone adder, the Han-Carlson adder, the Ladner-Fischer adder, the Brent-Kung adder, the Sklansky adder, and the Knowles adder.

Taxonomy Revisited



Various other design variants (such as the Lynch-Swartzlander or the Kantabutra adder using the optimum spanning tree design) seem to be not implementable for the sake of our 2D project since they require the usage of Manchester Carry Chains.

Manchester Carry Chain



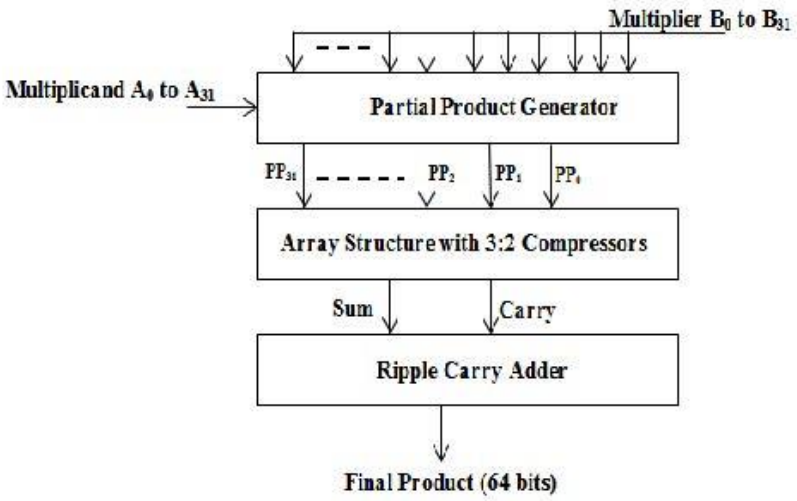
There were also other designs such as the Beaumont-Smith adder, the Ling adder, as well as other concepts such as carry-select and carry-skip. In the end, we decided to utilize the Han-Carlson design for our 2D. This optimization technique would also be useful when we design our adder for our 1D project.

Personal Update: Shifter & Multiplier Optimization (22 October 2020)

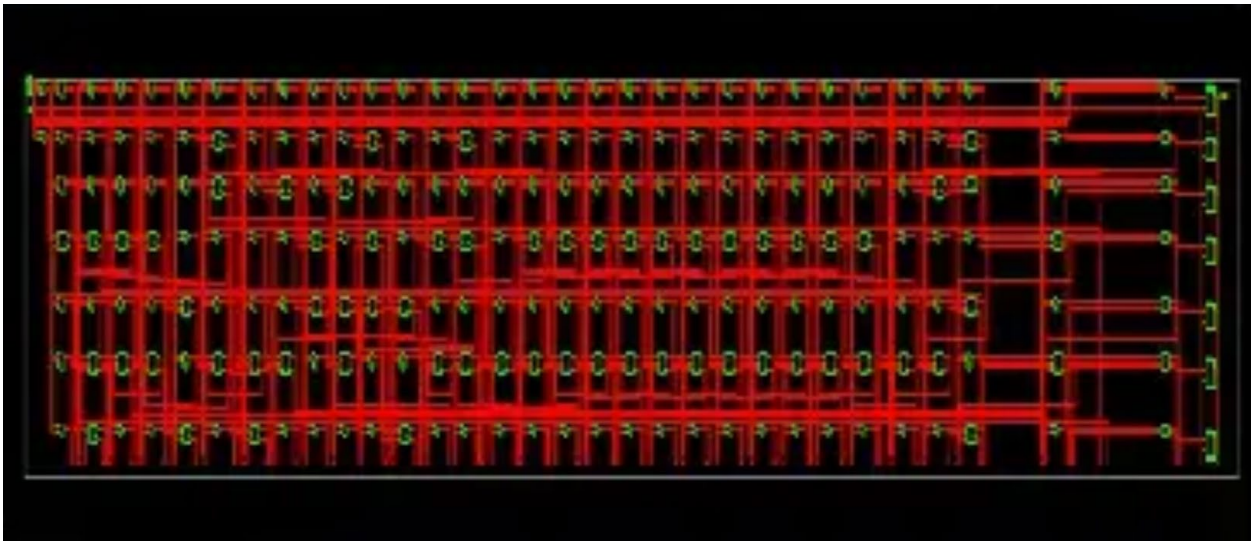
I made some further improvements to the adder design for 2D. I learned of a better way to get the V signal from the ALU for the comparator, which is by executing a XOR logic of the C30 (30th Carry Bit) and C31 (31st Carry Bit).

For the shifter, I managed to implement a more optimized design by combining the SHL, SHR and SRA functions together in one circuit (instead of 3 sets). This was done by using some additional multiplexers before and after the main cascade of multiplexers used to shift the bits. These additional multiplexers will perform some checks and minor modifications to the entering and exiting bits (based on the ALUFN values) such that we could accommodate for all of the 3 aforementioned instructions in one set of multiplexer cascade. That way, we save on the area, at the cost of higher latency. In overall, by running the simulations in JSim and comparing the timing analysis results, we have a lower area-delay product for my optimized version.

For the multiplier, instead of the traditional array multiplier, I learned about more optimized designs such as the Wallace tree multiplier, the binary tree multiplier, and the radix-4 Booth recoder multiplier. For instance, this is the block diagram of a 32-bit Wallace tree multiplier:

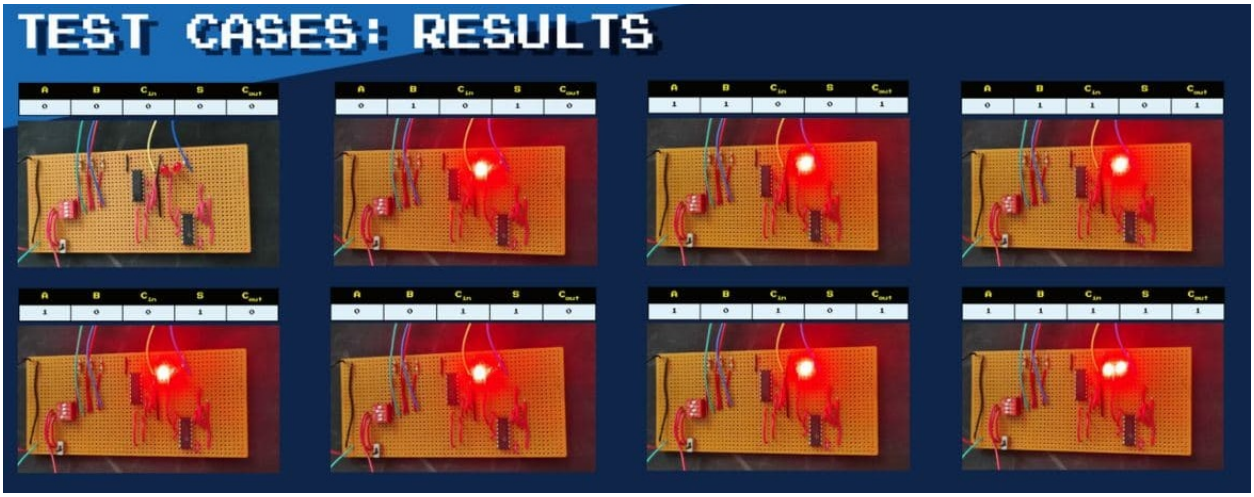


These designs would provide lower latency, at the cost of a messier/less ordered arrangement of the components used for the multiplier (such as the half-adders and full-adders). For example, this is the RTL schematic view of a 32-bit Wallace tree multiplier:



Team Update: Missing Test Cases (24 October 2020)

Professor Natalie informed me that we did not demonstrate images of the test cases in the poster and because of that, we did not obtain full marks for the MHP. She showed us what did the rubric mean by test cases:



While it is a bit sad to not get the marks for the test cases, the proportion/percentage of the test cases in terms of our overall grade is relatively small and as such, we decided to move on with the next parts of our project.

Reflection and Summary

I learned more about how a basic CPU architecture is assembled, along with its datapath connections. I also learned how each component contributes to the whole working process of the CPU, as well as how to optimize the components of the ALU.

Week 7-8 Entry

Introduction

My group did some game ideation and we chose to go with a modified version of Chopsticks (the hand game).

Team Update: ALU & Game Ideation (28 October 2020)

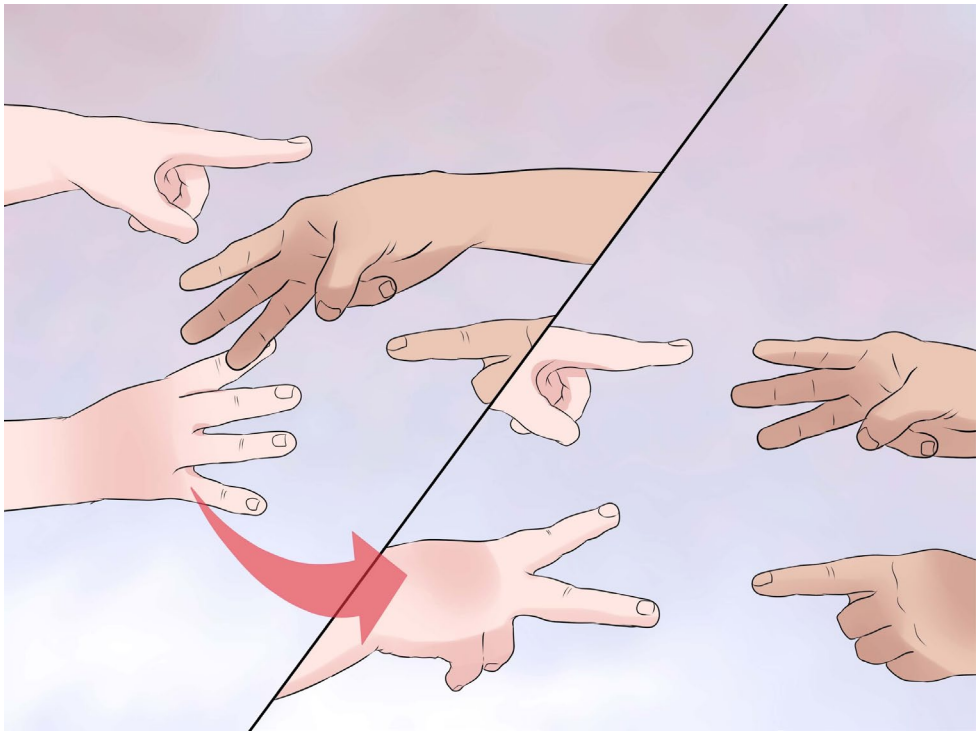
My team conducted a meeting at the benches outside of LT1 to discuss two things: the creation of the ALU and its tester, as well as our game idea.

For the ALU, we wanted to implement additional functionalities that support the game that we are going to develop. This is so that the design is streamlined and more in sync with the game (more optimized).

For the game, we came up with several ideas, such as:

- Button Smashing Bishi-Bashi Style
- Chopsticks
- Morra
- Kung Fu Panda Style Drumming Game
- Slapjack
- Ninety-Nine

We went to consult Professor Natalie about our ideas. This is important since we need to be aware of the limitations and the possibilities of implementing the game that we had in mind, given that we had limited knowledge in FPGA programming. We discussed the possibilities of creating each game within our timeline. In terms of scale and complexity, it seems that Chopsticks would be the best option. Chopsticks is a mathematical hand game.



We also decided to implement some modifications to the Chopsticks game so that there is an innovative element implemented in the game. The modification that we were considering would be to include boolean operations when playing the game.

Since we have decided on the game, we are ready for the checkoff next week. We would just need to design the datapath and purchase the hardware materials to start working on the game project.

More details for each game idea are discussed in the Appendix Section of our 1D Group Report.

Team Update: ALU Project (5 November 2020)

We divided the tasks of our ALU project such that half of the team members did the ALU module and the other half did the tester module.

For the ALU, the additional instructions that we implemented include:

- Multiplier:
    - MULTIPLY
    - FLOOR DIVISION
    - MODULO
- Adder:
    - INCREMENT A
    - INCREMENT B
    - DECREMENT A
    - DECREMENT B
- Boolean:
    - ZERO
    - NOR
    - NOT CONVERSE
    - NOT ‘A’
    - NOT IMPLY
    - NOT ‘B’
    - NAND
    - XNOR
    - IMPLY
    - ‘B’
    - CONVERSE
    - ONE

In the ALU’s GitHub repository, ‘A’ is denoted as ‘X’ and ‘B’ is denoted as ‘Y’.

For the tester, I faced some difficulties to implement it yet again. However, after some time debugging my code, I managed to get the tester to work (for both automatic and manual mode).

Reflection and Summary

I learned about properly managing the expectations and the scale of a game project so as to be able to complete it within the timeline reasonably. I also learned more FPGA programming for the ALU project.

Week 9-10 Entry

Introduction

For these 2 weeks, I researched on the possible implementations that could be done for the 1D game project that is feasible within the tight timeline provided. I worked with my team on the finite state machine and the custom datapath for our game, as well as started to source for materials and worked on the hardware of our project. We encountered some design issues and hardware-related difficulties which slowed down the progress of our project, but we marched on full speed ahead regardless.

Team Update: Designing the UI, Datapath and FSM (11 November 2020)

For our Checkoff 2, we were required to design the user interface, datapath and finite state machine for our game.

For the user interface, we brainstormed several ideas and took inspiration from those classic arcade games. To have that kind of arcade look, we decided to utilize the classic buttons that we actually used in our final product.

For the finite state machine (FSM), we started from the functionalities that we wanted to implement and designed “sub-collections” of states for each of those functionalities (such as POWERUP, SPLIT and the normal ATTACK mode). After that, we decided that instead of implementing a normal IDLE mode, since our game is turn-based anyway (between 2 players), we further simplified the FSM by cycling through the two players using two “areas” of states. Then, we bootstrapped our “sub-collections” of states into the turn-based FSM, making adjustments as and when we deemed necessary.

For the datapath, we take inspiration from the original Beta datapath and customized it to suit our needs. We removed some unnecessary parts such as the Program Counter and Memory Unit and added some paths of our own, such as certain constants as inputs to the ASEL, BSEL and WDSEL multiplexers. We adjusted the datapath depending on the needs of our FSM.

Professor Oka approved of our game during our Checkoff and gave the green light. It's go time!

Team Update: Time to Go Shopping! (14 November 2020)

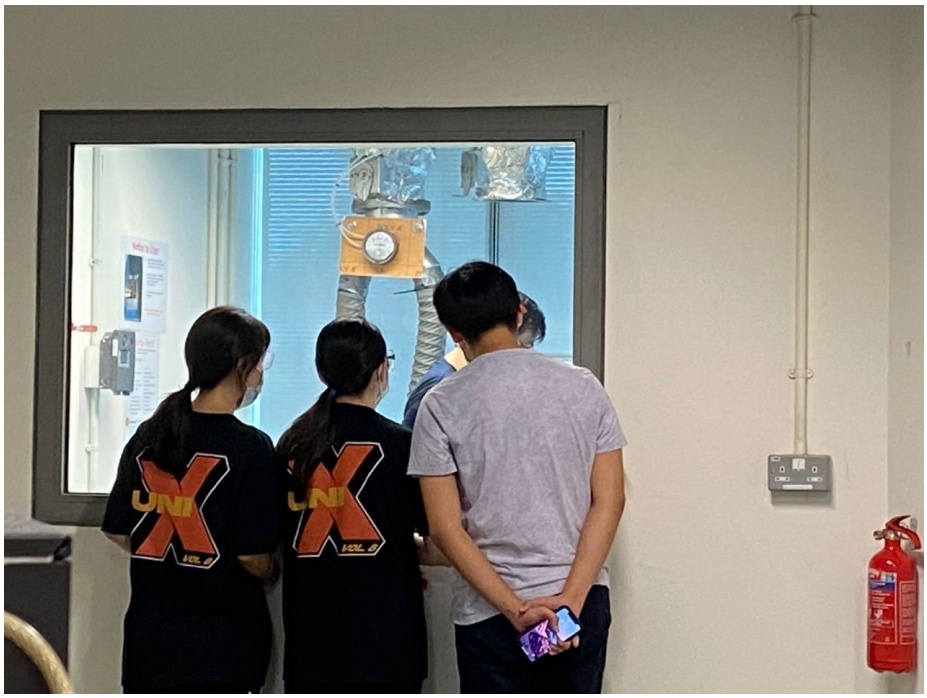
We went to Sim Lim Tower (not Sim Lim Square) to go shopping! Besides shopping for our necessary and important parts (as listed in the component list in our group report), we also found several other fascinating components, such as these flexible PCB "board":

We bought these large-sized donut boards so that we can have plenty of space to solder our components together:

For the acrylic, I managed to self-source them (with the red and blue color just as fit for a competition style of the game):

Team Update: Case Designing, Laser Cutting & Sanding (17 November 2020)

After acquiring our hardware components, we went to book a session in SUTD's FabLab to laser cut our acrylic case. These are my groupmates looking at me when I did the laser-cutting in the laser-cutting room (in my opinion, they look like cute children):



We made some minor mistakes with the CorelDraw CAD file when doing the laser cutting but they were easily resolved. One major problem with the laser-cut case was that the button holes were a little smaller than what we expected (even though we already accounted for these screw parts of the buttons):

As such, we needed to sand the holes individually until they were able to fit the buttons:

Reflection and Summary

For weeks 9-10, I learned to design my own computer architecture in the form of a hardware datapath for our game. I also learned to design a relatively simple and nice-looking user interface for our game prototype with the help of my groupmates. There are a lot of design considerations when crafting the game prototype and as such, these 2 weeks have been a very valuable learning experience. For the remaining 2 weeks, we just need to finish off assembling the hardware, finish the software component and hopefully manage to debug all of the potential issues.

Week 11-12 Entry

Introduction

For these last 2 weeks, I continued to work on the hardware for our project and started to craft and debug the datapath and finite state machine in Lucid. My team encountered some hardware issues as we assembled the prototype and as such, we had to re-solder some components and source for more components, parts and materials. We finished our hardware quite late into the

timeline, and unfortunately, we did not manage to finish debugging the software section in time for the submission and checkoff deadline.

Personal Update: A Slight Mishap (23 November 2020)

So, today I accidentally hurt my finger while carrying our prototype's acrylic base. The base somehow slipped off my hand as I carried it downstairs from my hostel room and in an attempt to re-balance the base's centre of gravity's position on my hand, I held the acrylic too firmly, resulting in the corner piece of the base slicing through my thumb's skin and a section of the acrylic's side broke off.

This is a non-explicit version of a picture of my thumb with a bandaid:

This is the result of the side piece breaking off from the acrylic:

Thankfully, we managed to glue back the broken piece using some hot glue. This was actually part of the reason why we decided to cover up the sides of the translucent acrylic base with a custom design of our own using sticker paper.

My groupmates somehow also injured themselves in one way or another when handling our prototype. I guess we really literally put our blood, sweat and tears into this project, eh?

We also managed to fully finish the construction of our whole case:

We also encountered several design issues which we have outlined in our group report. We managed to solve them eventually, but they definitely hindered us and slowed our progress.

Team Update: Alchitry Br Pin Layout Issue Strikes Again (25 November 2020)

Since our prototype requires the usage of more pins than normal (in fact, for the final product, we utilized about 95% of all of the available SingleEndedIO pins on the Alchitry Io!), it is time for me to delve deeper into implementing custom .acf constraints. Without using more pins than the ones pre-defined in the built-in included io.acf default constraint, it would be impossible for us to

implement the game. Thus, it became sort of a mandate for me to learn how to implement the custom constraints.

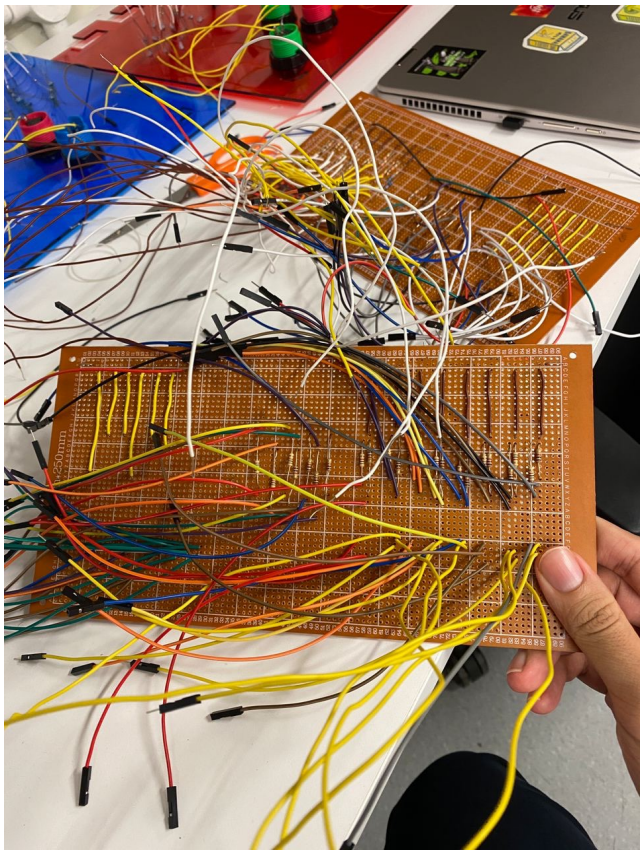
Personal Update: Last-Minute Sourcing for Components (30 November 2020)

Since we needed to purchase more male-female jumper wires to solve our connection issues, I needed to go to Sim Lim Tower again pronto. In the end, I definitely bought more than what was needed but some backup is always good. As the deadline for 50.001 Infosys' 1D Project is approaching close as well, I needed to quickly travel back and forth in time for me to regroup with my Infosys project to continue my work for my Android application. As such, I went to take Grab and a taxi (since my phone ran out of battery) to travel to-and-fro.

Team Update: Final Stretch (1-3 December 2020)

I apologize for the sparse updates since I have been working hard on our 1D game prototype. For these 3 days, we attempted to finish the assembly of our prototype and finish designing our hardware description using Lucid.

These are the 'beautiful' views of all of our connected electrical components and wirings:

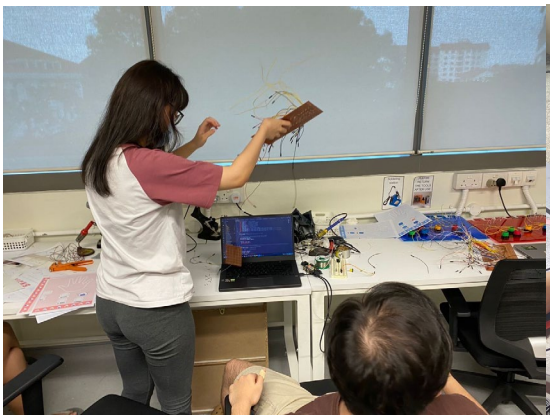


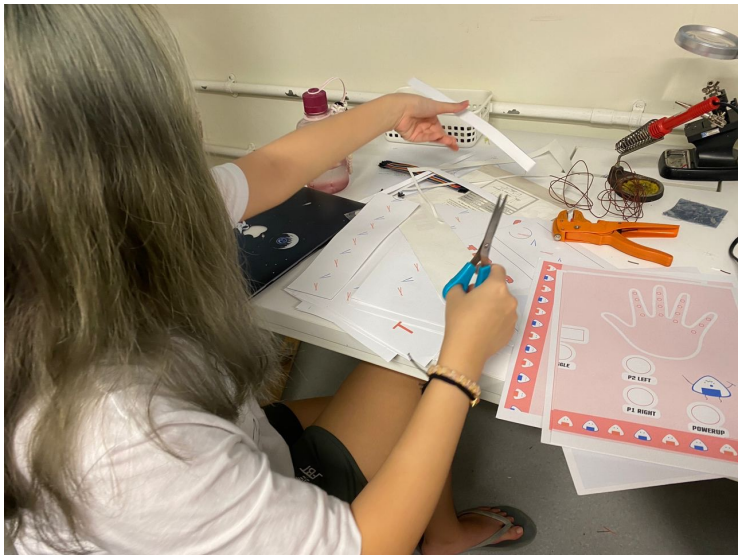
We stayed up for 3 whole days with only 2-4 hours of sleep each (swapping in-between) in an all-out effort to produce a meaningful deliverable:


PAPER
PLASTIC
RECYCLE
PAPER
HERE
RECYCLE
PLASTIC


LECTURE THEATRE 1

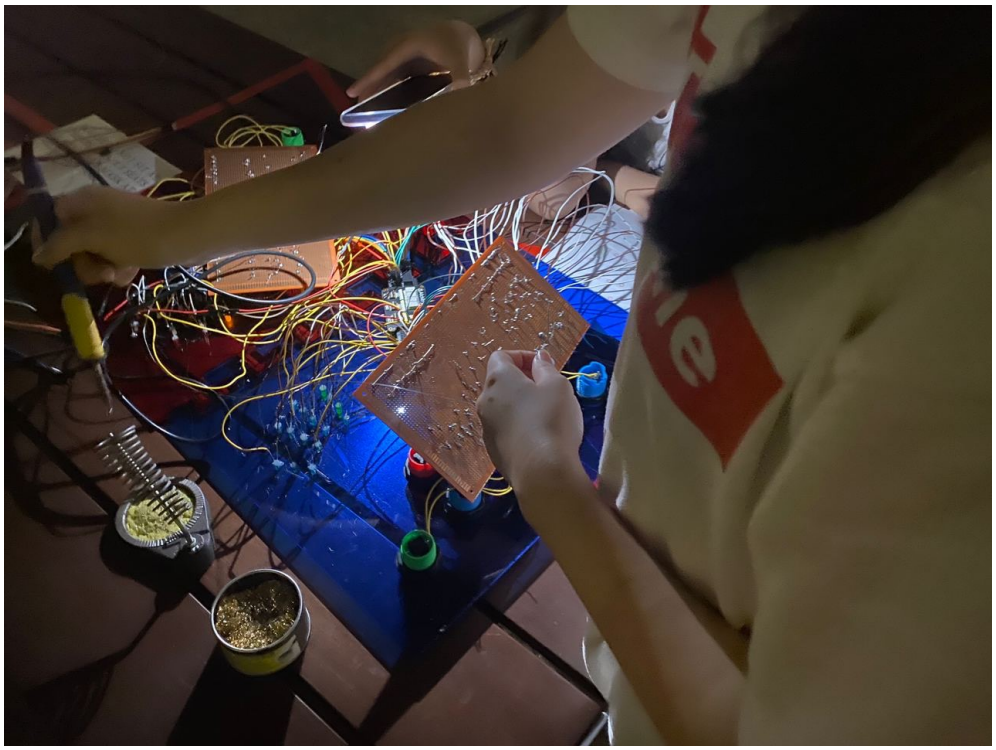
We worked at DSL during the day and in front of LT1 during the night due to the prevailing COVID-19 restrictions and guidelines imposed on the university:

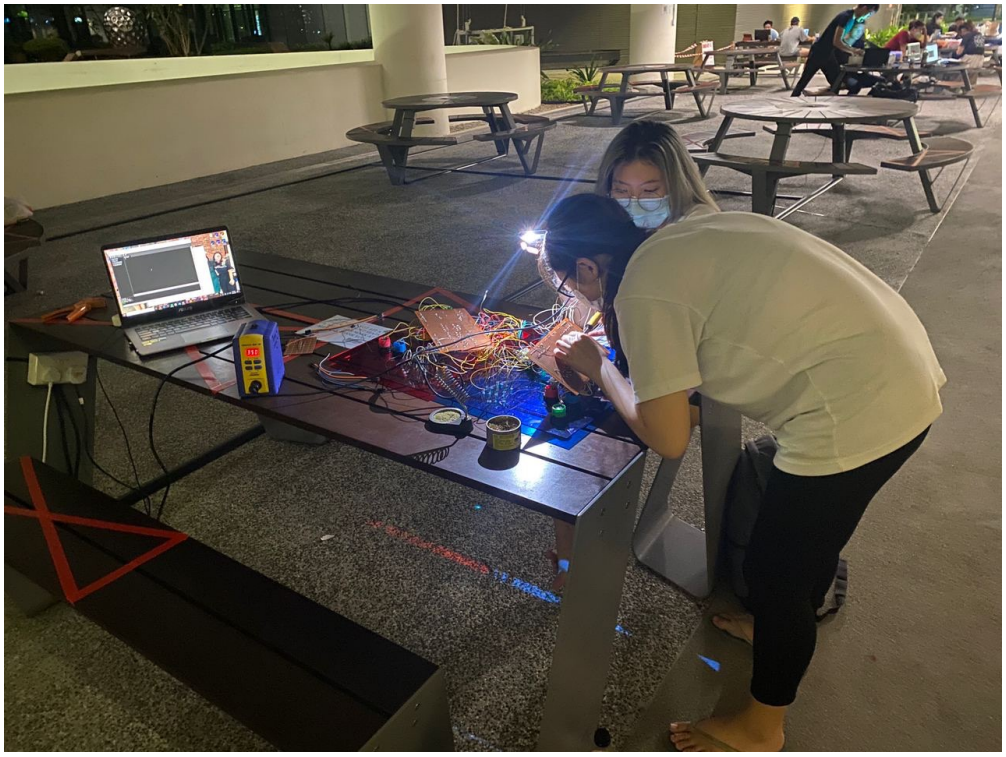


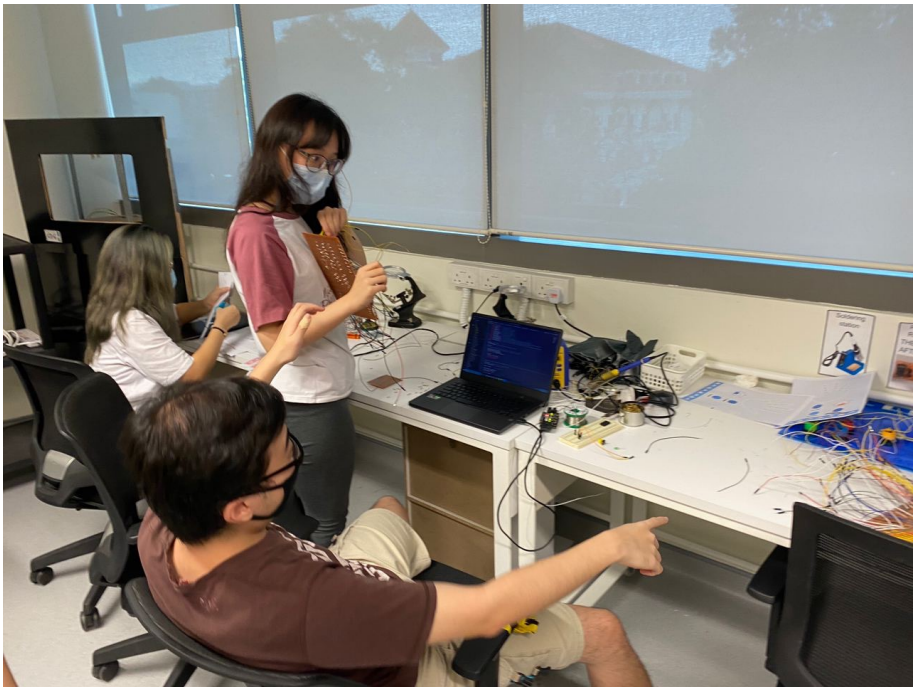


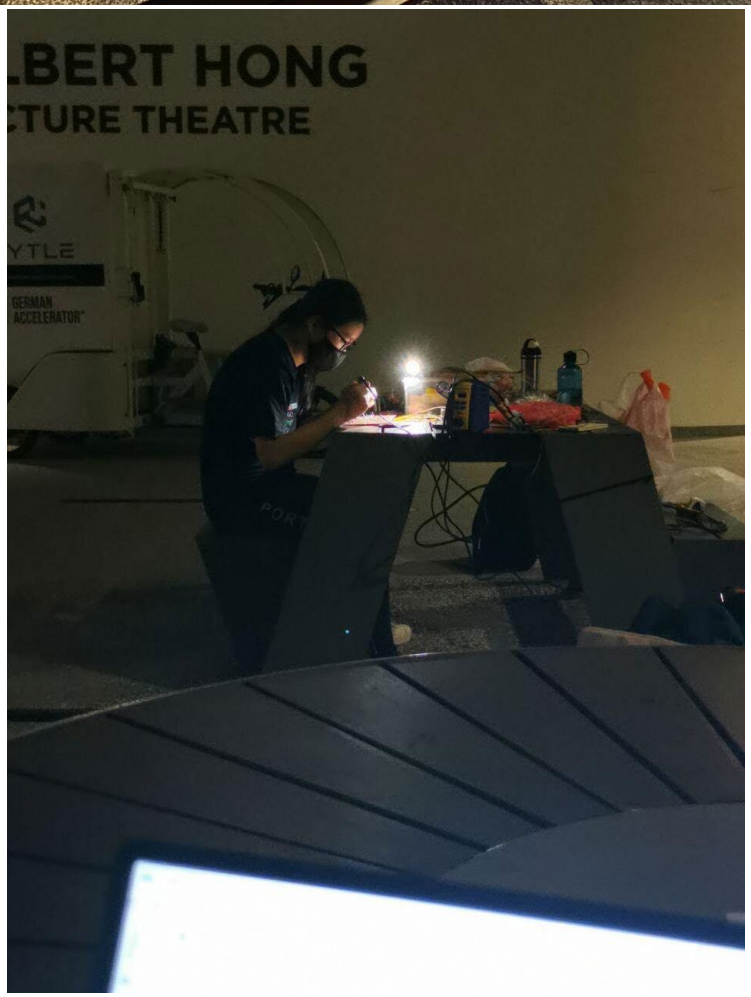
We had quick meals in-between and even ordered delivery food so that we did not have to waste time sourcing for food:

We also split into two concurrent teams, one for hardware and another for software. These are some of the photos taken during these trying times:








BERT HONG
TURE THEATRE

In the end, even though we managed to finish assembling the hardware, the software part was another story. We had several persistent bugs which hinder players from playing the game properly, causing us to be not prepared for our final submission checkoff deliverables. This is us trying to debug the software (notice the despicable red error lines on the screen):

As such, we accepted our fate and submitted our other deliverables (with a creative approach to our video deliverable) by the deadline, unfortunately without a properly working prototype. Perhaps I could finish debugging the game during these incoming holidays? (I could try using the Xilinx Vivado tool on an Ubuntu dual-booted system as suggested by Ragul, since it is not working on my Windows.)

Reflection and Summary

Even though we did not manage to finish our 1D project and get a properly working prototype as our deliverable in time for the deadline and checkoff, what matters is that we tried our best and put in our best effort. We also learned something new and valuable along the way. As we persevered through the tribulation period, we became stronger. And for that, I am grateful. Post-checkoff, I tried to debug the Lucid code and while I managed to get the turn-based state machine roughly working, I still faced some persistent and annoying bugs that I have described fully in our corresponding DigiSticks' README on our GitHub repository. Regardless of the status of our project, I am thankful that we had this short opportunity to learn more about computer architecture through this course and through this 1D game project. I certainly hope that one day, if my team could afford the time (perhaps this incoming holidays?), we could come together once again to properly fix the currently existing bugs with a fresh mind and have a properly working game prototype, not for the sake of academics, but for the sake of learning.

This is James, signing out.