Lab # 3
Continued Instruction Set Design; TTL SSI Chips vs simulation, HDL FPGA

Lauren Cugliotta
Senior Computer Engineer
Demonstrating: -

Scott Tevis
Junior Computer Engineer
Demonstrating: -

Andrew Boeren
Sophomore Computer Engineer
Demonstrating: -

1. Assignment

ASSIGNMENT: LAB #3 **Continued Instruction Set Design; TTL SSI Chips vs simulation, HDL FPGA**
COURSE: EGR/CS433 Advanced Computer Engineering Lecture & Lab
SYLLABUS: http://users.etown.edu/w/wunderjt/syllabi/CS433%20Wunderlich,Joseph.htm
INSTRUCTOR: JT Wunderlich PhD
LATE PENALTY: Minus 33.3% per class period for each late item
LAST REVISED: **3/11/20 ADAPTATION FOR CORONAVIRUS 3/12/20 CLASSES CANLCELED 3/13-3/17**

PRE-LAB:
Read our custom NanoLC Lab Manuals:

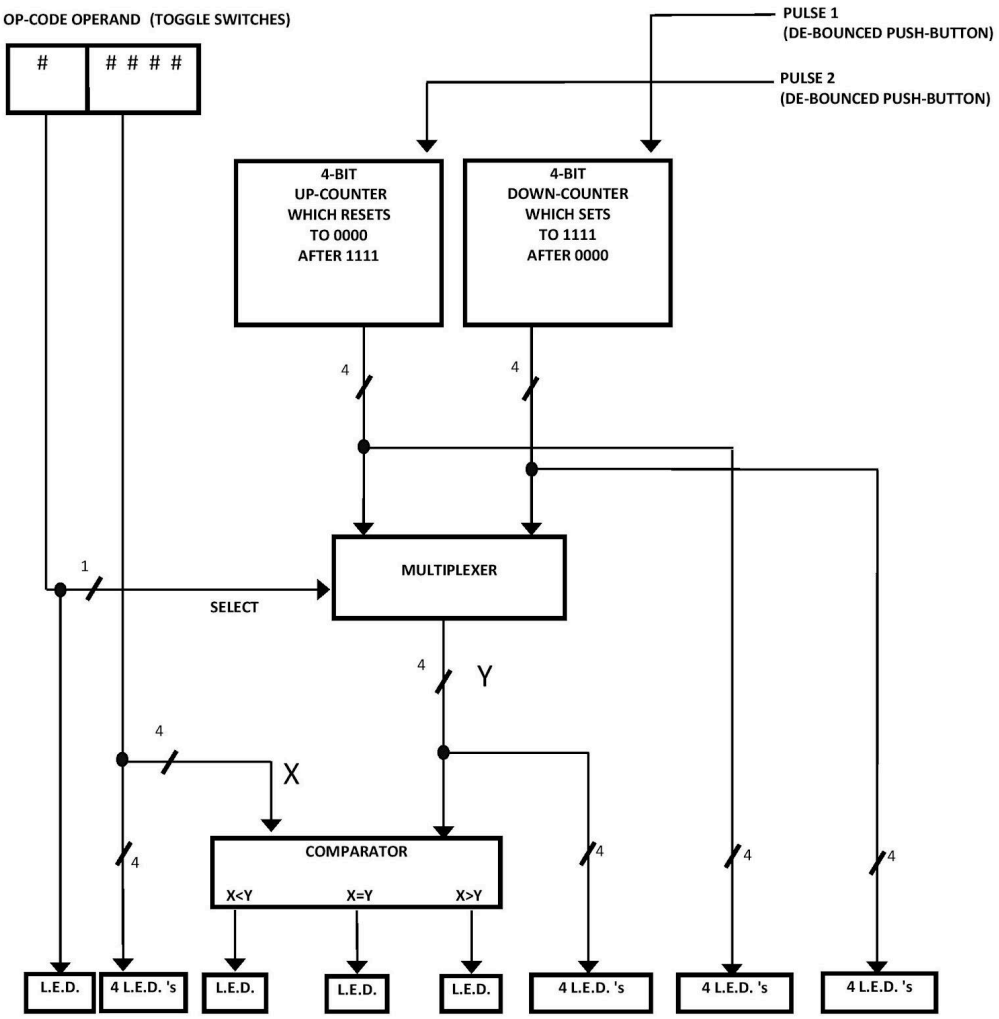
- **PACKET 16 BOOKSTORE 433 Chips_PSU_Trainer Lab Manual** *v1.2*
- **PACKET 17 BOOKSTORE 433 2019 MANUAL FPGA** using **HDL** (Hardware Descriptive Language)
- For reference only: (EGR 333 topics): **DE-BOUNCING SWITCHES**: http://users.etown.edu/w/wunderjt/333_BOUNCE_2.pdf
  RESISTORS: http://users.etown.edu/w/wunderjt/333_RESISTORS.pdf **and** http://users.etown.edu/w/wunderjt/333_RESISTORS_Supplement%202.pdf

DURING LAB: Design and implement the circuit below in five ways:

1. A **Logisim circuit simulation** using only FLIP-FLOP's, INVERTORS, AND's, OR's, and NAND's for all circuits
2. ~~**TTL SSI chips** on circuit trainers using only FLIP-FLOP's, INVERTORS, AND's, OR's, NAND's. Also implement **20 LED's** (with resistors too, if needed)~~
3. ~~**2019 FPGA** using **HDL** (Hardware Descriptive Language) **OUTPUT ALL BUT THE OP-CODE AND OPERAND (i.e., since there are only 16 LED's on the FPGA)**~~
4. **CREATE ONE YOUTUBE VIDEO DEMONSTRATING ALL OF THE ABOVE (MAXIMUM 10 MINUTES !)**

FPGA HDL BOARD/SIMULATION and TTL BREADBOARD WILL BE MADE INTO ANOTHER LAB WHEN WE RETURN TO PHYSICAL CLASSES

Instruction Set:
(OP-CODE=1): Compare operand to up-counter count
(OP-CODE=0): Compare operand to down-counter count



GRADE PERCENTAGES and DUE DATES:

~~(20%) MON 3/23/20 AT 3:30PM~~ ~~Demonstrate **IN YOUR YOUTUBE VIDEO PAD-234 Circuit Trainer circuits** [primarily for **5 volt** TTL chips]~~
none Demonstrate **Old Circuit Trainer circuits** [primarily for **5 volt** TTL chips]
none Demonstrate **RadioShack Circuit Trainer circuits** [primarily for **~3 volt** CMOS chips]
(33%) FRI 3/27/20 AT 3:30PM Demonstrate **IN YOUR YOUTUBE VIDEO Logisim Circuit simulations**
none Demonstrate **Phoenix Contact NanoLC PLC simulations AND real-time systems**
none Demonstrate **Phoenix Contact Advanced Axioline PLC real-time systems (PCworks software)**
none Demonstrate **Phoenix Contact Advanced PLCnext PLC real-time systems**
~~(20%) MON 3/23/20 AT 3:30PM~~ ~~Demonstrate **IN YOUR YOUTUBE VIDEO 2019 Field Programmable Gate Array (FPGA) simulations and Real-time systems (in HDL Hardware Descriptive Language)**~~
none Demonstrate **2018 Field Programmable Gate Array (FPGA) simulations and Real-time systems**
none Demonstrate **INTEL 8051/80251 or ARM microcontroller simulations**
none Demonstrate **INTEL 8051/80251 or ARM microcontroller real-time systems**
none Demonstrate **Raspberry Pi real-time systems**
none Demonstrate **Arduino real-time systems**
none Demonstrate **Basic-Stamp real-time systems**
none Demonstrate **Direct PC-port real-time systems**
none Demonstrate **Remote mobile-device real-time systems**
none Demonstrate **LabView real-time systems**
none Demonstrate **Isolated high-voltage bench-test** (with low Voltage electronics disconnected)
none Demonstrate **Other**:
(33%) FRI 3/27/20 AT 3:30PM Submit Written Report **VIA EMAIL TO ME**
(33%) FRI 3/27/20 AT 3:30PM Submit PowerPoint, Poster, or Video: **YOUTUBE VIDEO GRADED ON PRODUCTION AND COMMUNICATION/EFFICIENCY VALUE**

LAB RULES & PROCEDURES

IF LABS ARE BUILT (AND POSSIBLY REBUILT), BUT DON'T FULLY FUNCTION: For **demonstrations** and **reports**, deduct depends on how adequately you identify problems. For example, make test set-ups to verify functionality of isolated simulation sub-parts, chips, circuit trainer elements, software, relays, other electronics, motors or other higher-voltage circuits and devices. PROVE THAT NO EASY FIX OR SUBSTITUTION WAS POSSIBLE or EASILY IDENTIFIABLE AT THE TIME. Discuss (1) How you identified problems, and (2) How you tried to fix them. Include evidence that you fully understand and have properly connected all pins on a given chip (including considering floating-pins, powering the chip, needed pull-up resistors, proper voltage levels, etc.), and that you have exhausted much time attempted to solve all problems.

DEMONSTRATIONS Alternate team members demonstrate lab to me (partners must be present). We do this for previous week's assignment while you begin the next assignment.

TEAM LEADER has responsibility of coordinating all equipment problems and acquisition of needed parts. Try to stick with same person for this role.

REPORTS must include:

- **PHOTO'S OF ALL CIRCUITS AND TEST SET-UP'S BUILT**
- Title Page with lab number, name of lab, your names, Majors, Year (e.g., Junior), who is demonstrating, and who is the designated TEAM LEADER
- Sections numbered and tilted as follows (always list all of these, and simply put "NA" if not applicable):
  1. **"Assignment"** (An exact copy of everything in this document -- exactly how it looks here)
  2. **"Equipment Used"** (A list of hardware and software) **INCLUDE PHOTO'S OF ALL EQUIPMENT**
  3. **"Methodology"** (including all **DESIGN STEPS**, analysis, **DECISIONS MADE**, etc.) **INCLUDE PHOTO'S OF ALL CIRCUITS BUILT**
     ALL DESIGN STEPS MUST BE INCLUDED for Digital Logic designs (**NUMBERED** as in EGR/CS 332). If design step not done, list as "N.A."
     For Combinational Digital Logic Design:
     Step 1: Define problem
     Step 2: Encode variables
     Step 3: Create truth table
     Step 4: Find simplified function(s)
     Step 5: Draw logic circuit
     Step 6: Convert to NAND's
     Step 7: Check assumptions
     Step 8: Chip circuit diagram
     For Sequential Digital Logic Design::
     Step 1: Define problem
     Step 2: Create state diagram
     Step 3: Encode variables
     Step 4: Minimize machine
     Step 5: Create state table
     Step 6: Append flip-flop inputs
     Step 7: Find simplified function(s)
     Step 8: Draw logic circuit
     Step 9: Convert to NAND's
     Step 10: Analyze any unused states
     Step 11: Revise state diagram
     Step 12: Check Assumptions
     Step 13: Chip circuit diagram
     COLOR-CODED LOGIC DIAGRAMS are required for any digital circuit (Breadboard, FPGA, etc.)
     COLOR-CODED CIRCUIT SCHEMATICS are required for any circuit implemented (Breadboard, PLC, ladder logic, etc.), color is a must, hand-colored is ok
     FLOW CHART is required for any program
     COMMENT EVERY LINE OF CODE
  4. **"Options"** (if applicable, a comparison of each method used)
  5. **"Problems Encountered"** (including any debugging methodology) **INCLUDE PHOTO'S OF ANY TEST-CIRCUITS BUILT**
  6. "**Testing Methodology**" (including timing traces, test-vectors, and **RATIONALE FOR HOW YOUR METHODOLOGY ASSURES QUALITY**)
     Including estimated PROBABILITY of satisfactory coverage by chosen test vectors
  7. **"References"** (in standard IEEE format)
  8. **"Appendices"** (for spec sheets, etc.)

GRADING For both demonstrations and reports, a **92** is for everything done very well and professional; and to get more points **enhance things in creative ways**

2. Equipment Used

Software:

- **Logisim**

3. Methodology

Design Process Steps

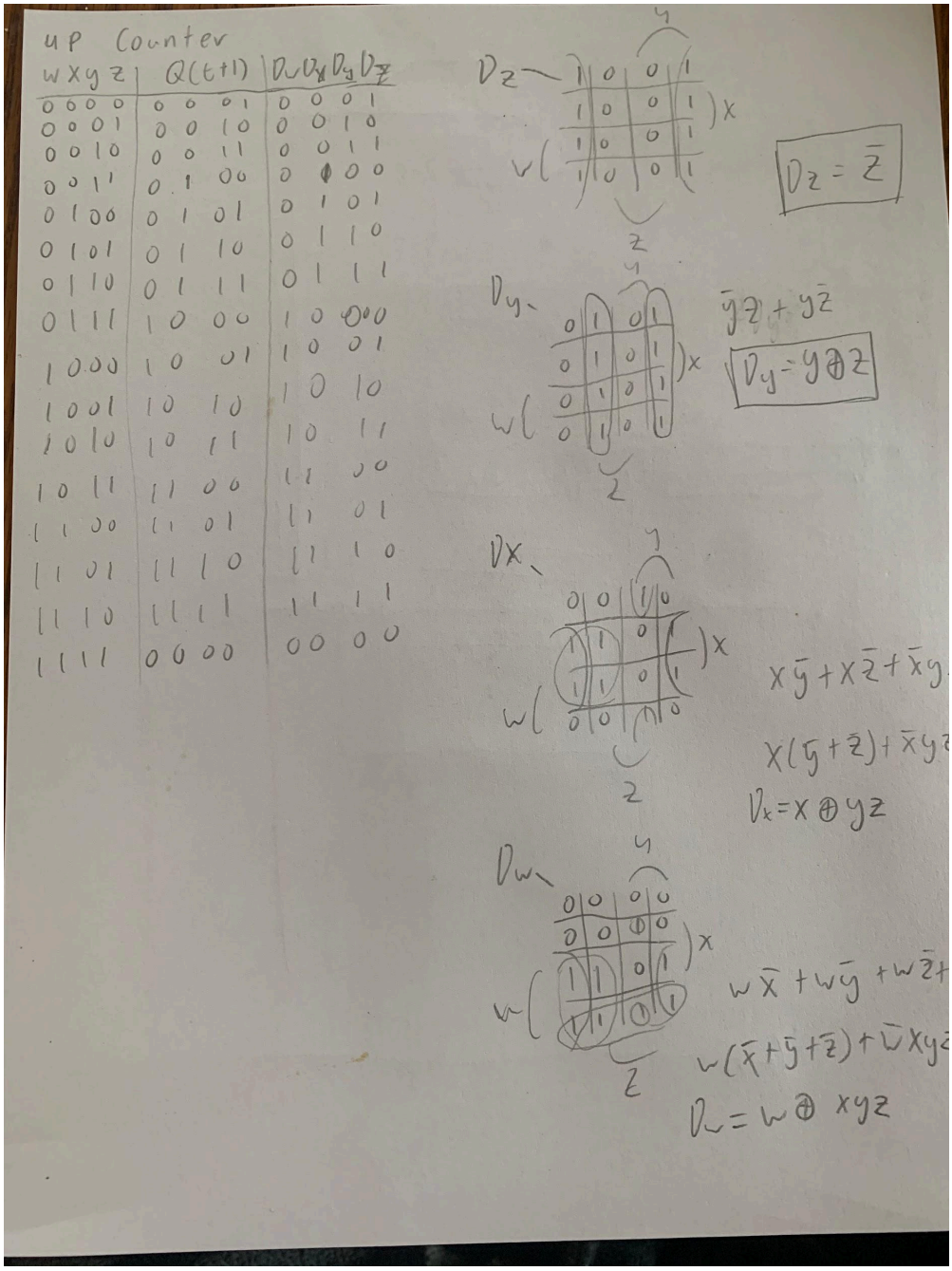
Step 1, Define Problem

The problem for this lab was to create two 4-bit counters, one up and one down, a multiplexer and a 4-bit comparator. This requires a total of 8 inputs (clocks for each counter, a reset, operand, and comparator inputs).

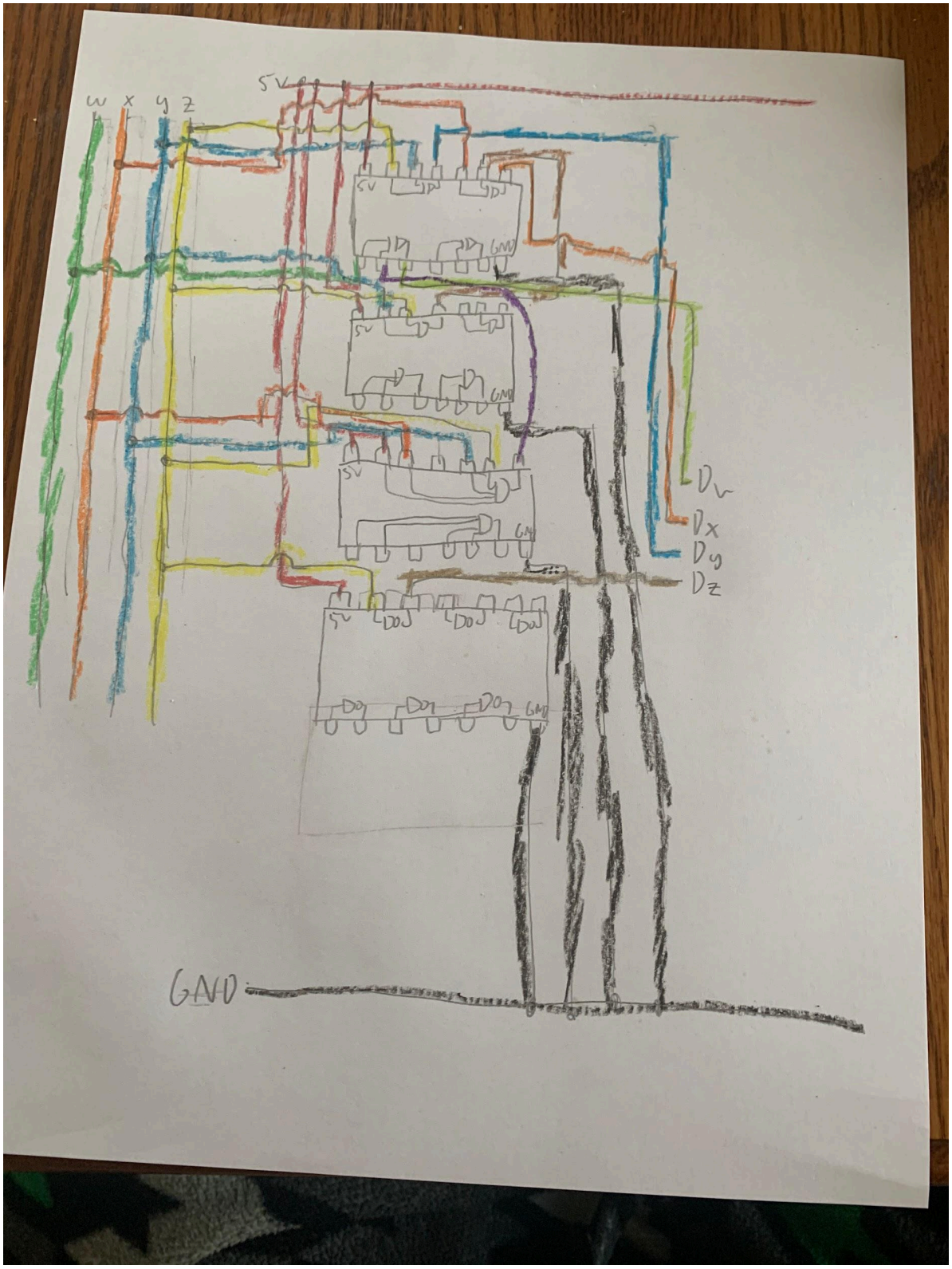
Step 2-6: Create state diagram, Encode Variables, Minimize machine,Create state table, Append flip-flops

For each counter we now have 16 states. For each counter what will change is the progression of states. Each counter needs 4 D Flip-Flops to work and expanded on the difficulty of the previous counters from Lab 2.

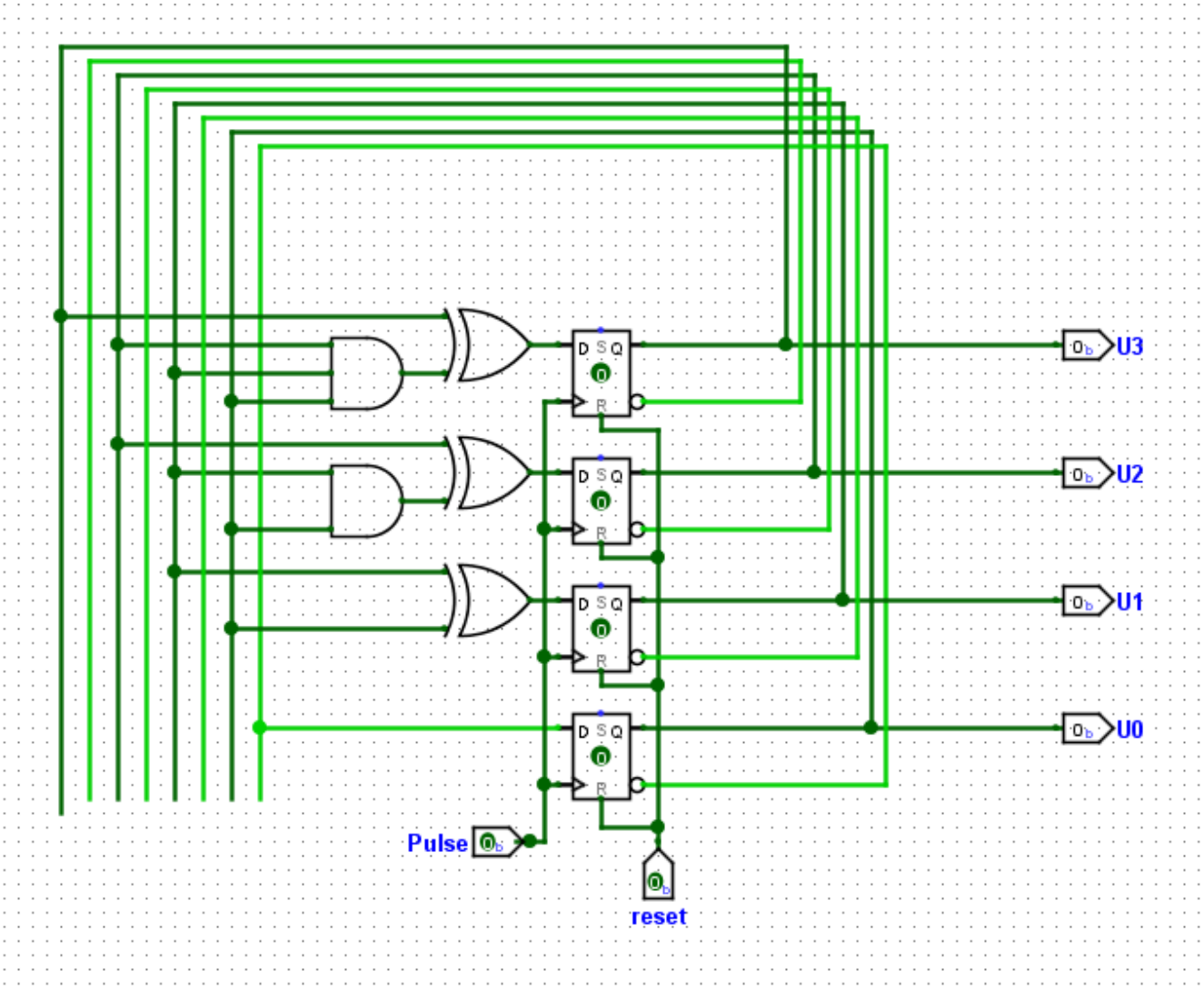
Truth Table Logic for Up Counter



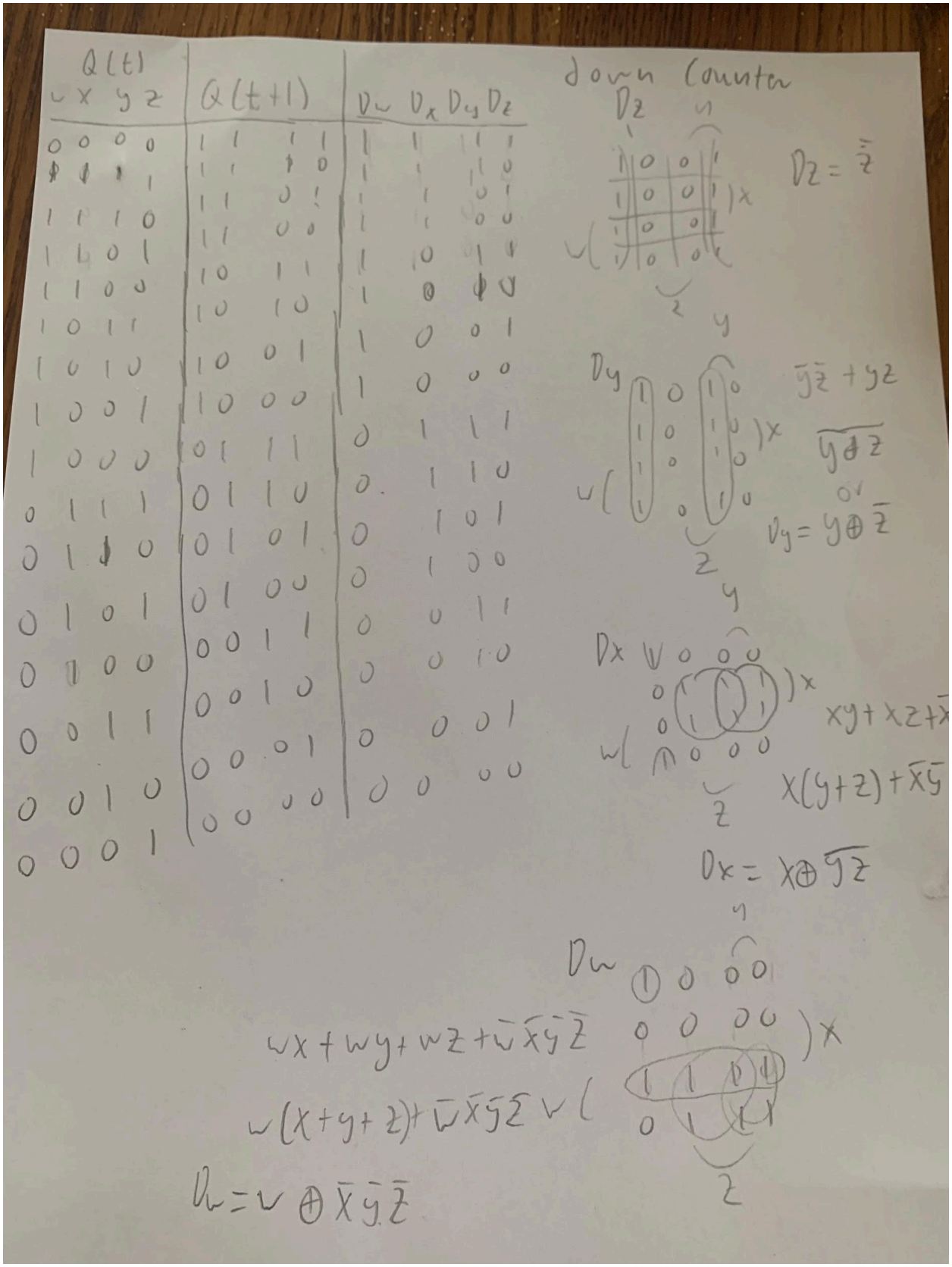
Chip Diagram for Up Counter



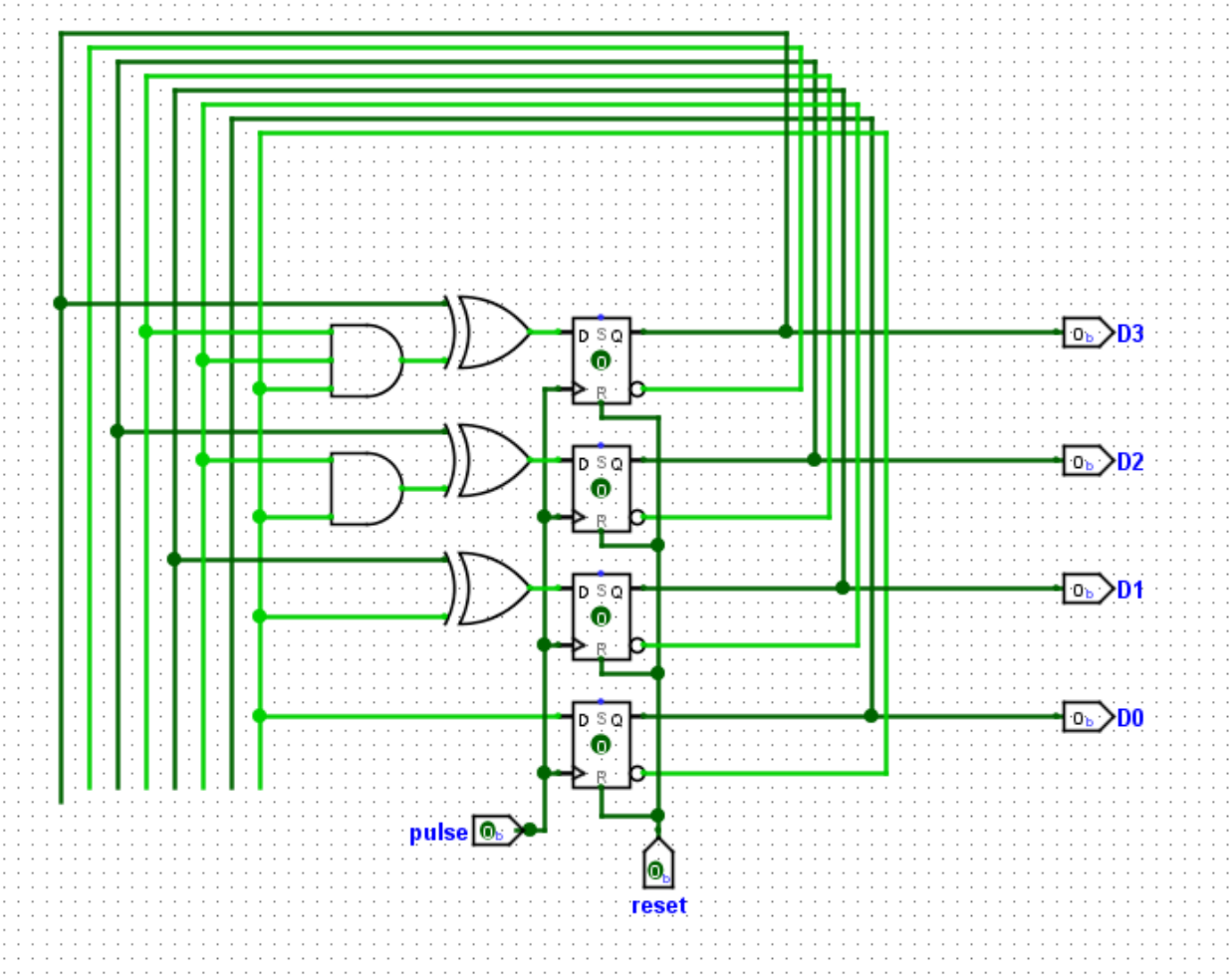
Up Counter in Logisim



Truth Table for Down Counter



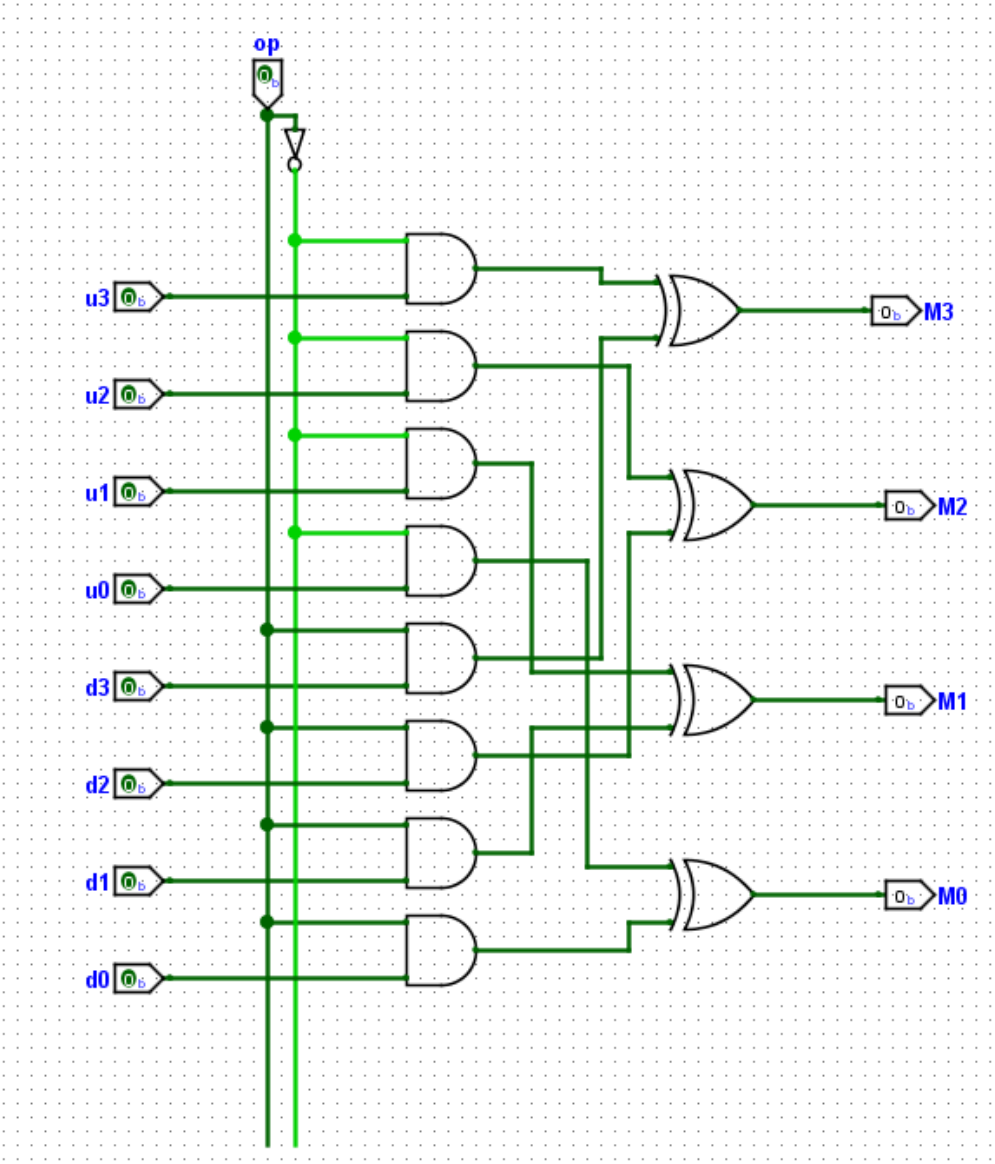
Down Counter in Logisim



Step 7-8: Find simplified function(s), Draw logic circuit

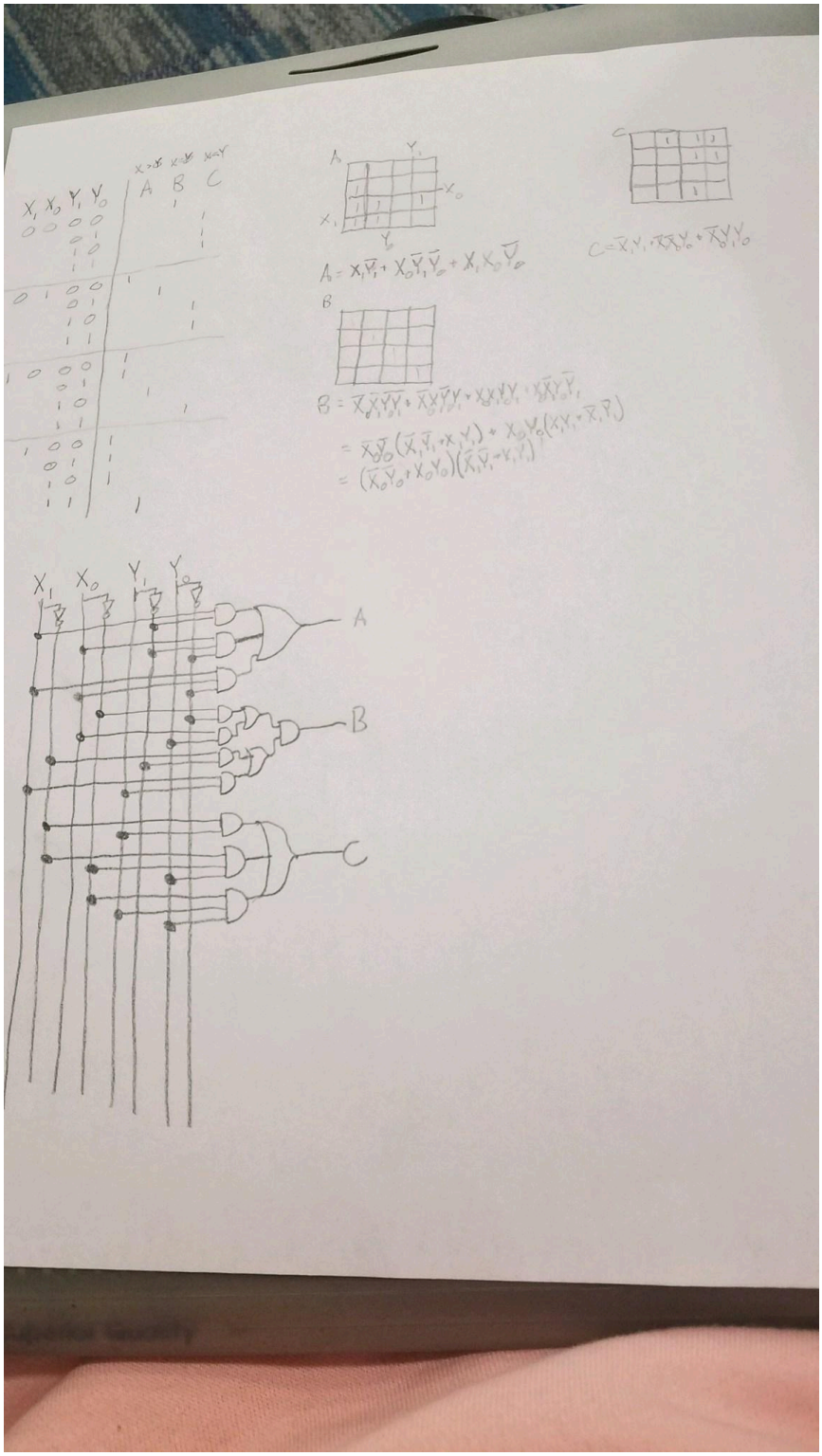
Next we made an Ad Hoc Multiplexer. Since an 9-input truth table wasn't a logical step to take, it was easier to use the idea of the previous multiplexer and expand on it from there. We knew that by reusing the assets for each bit from Lab 2 Multiplexer we could implement them for each new bit.

Multiplexer

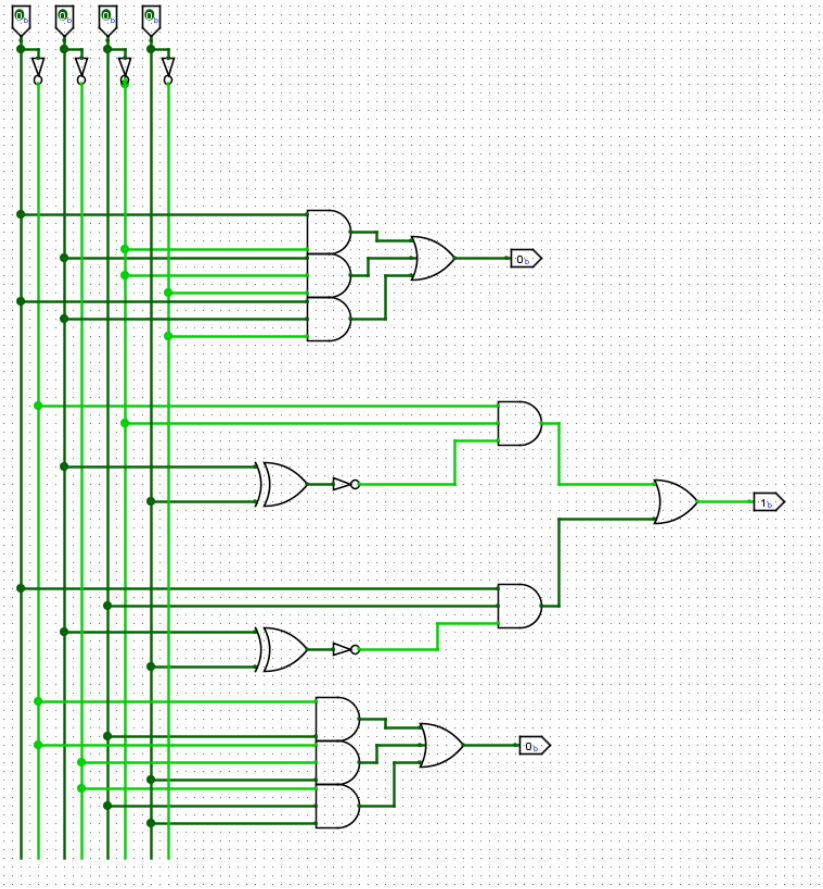


After that we began to work on the comparator. At first we were overwhelmed since we couldn't do an 8-input truth table. While we contemplated how to solve the issue, Clay suggested using two 2-bit comparators, instead of making one big 4-bit comparator. This completely changed the difficulty and we were able to easily solve this problem with a little ingenuity from Andrew.

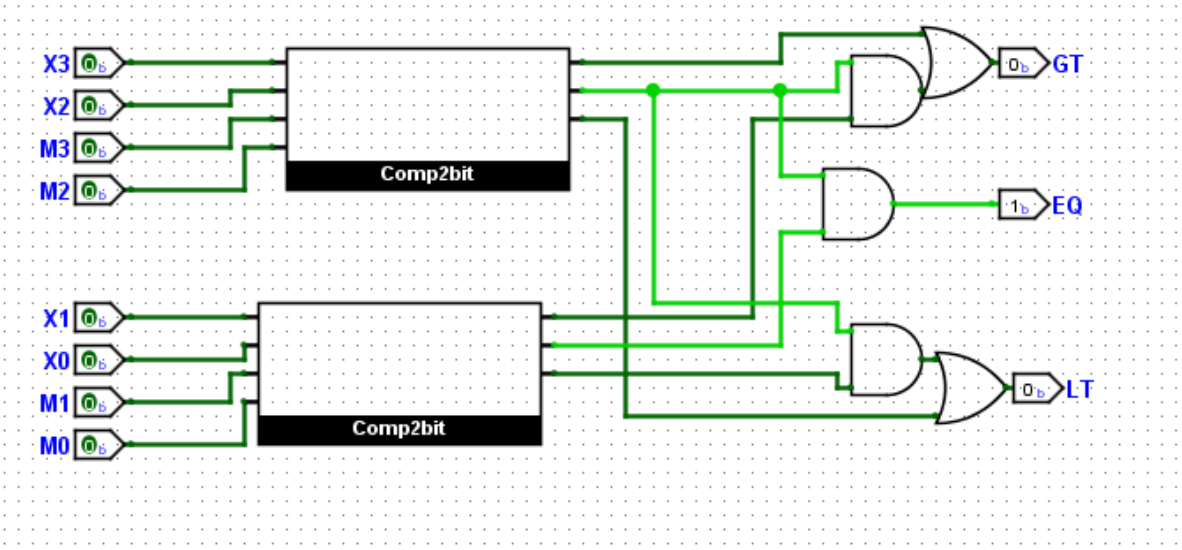
2 Bit Comparator Logic from Lab 2



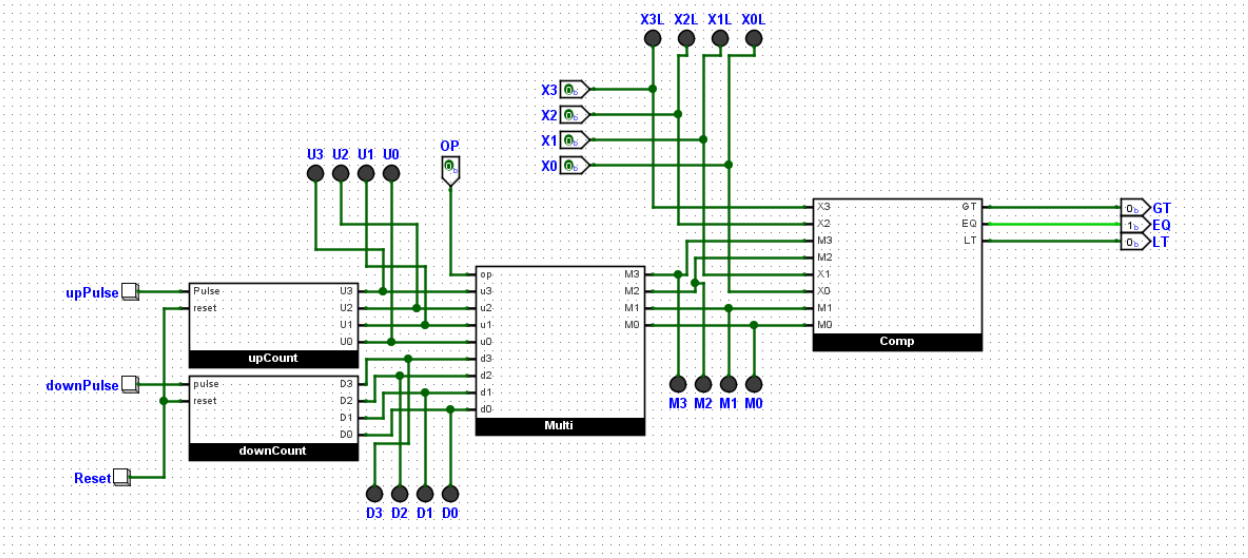
2 Bit Comparator



Final Comparator



Final Circuit



Step 9: Convert to NANDs

Due to the overall complexity of this lab, we decided not to convert to NANDS to both save time and remove the need to overcomplicate things.

Step 10-11: Analyze any unused states, Revise state diagram

We had no unused states and therefore had no need to revise our state diagram.

Step 12: Check assumptions
We would assume our clock pulses work for our circuit trainer but we cannot confirm this assumption at this time.

Step 13: Chip circuit diagram
Like the last lab, our space on the circuit trainer had been extremely limited. Therefore we used a comparator chip again to decrease the amount of space we needed on the trainer. We did create the multiplexer and counters with gates and flip-flops however. (We did create the circuit in the lab before online classes began but we have no video/pictures).

4. Options

Logisim:
The options came from the multiplexer and the comparator. For the multiplexer, we could have used OR gates instead of XORs for the final output. Both gave the desired result but the XOR felt more correct and accurate to us.
For the comparator, we could have tried to make a 4-bit comparator from scratch but it would have been difficult to do. So we chose the easy option of combining two 2-bit comparators instead.

5. Problems Encountered

Logisim Simulations:

The main problems we encountered came from the 4-bit comparator. When we were unable to solve a truth table for the comparator, we looked up what the circuit should look like. This was for us to try to reverse engineer a truth table out of it. In the end it proved too difficult for us to work with and when we changed to the multi 2-bit comparator, some of the inputs were messed up from the previous attempt.

6. Testing Methodology

Logisim Circuit Simulations:

For this lab, we had a total of 512 possible outcomes. It was not feasible to test each outcome due to the immense number. We got an approximate 25% certainty by completing 128 tests. These tests are the lowest state of the User Input (0000), the highest state (1111), and two opposite middle states (1010 and 0101).

This way we confirm that the Less Than and Greater Than outputs work in their respective extremes and work somewhere in the middle. However, the Equals To output may be ineffective, but we do get 4 outputs on each counter to feel comfortable with.

Our certainty could likely be higher if we assume that our outputs from lab 2 are counted or considered. But due to them being needed to be checked again, we can't guarantee that.

7. References

DM74LS73AN Fairchild 14P/DIP. [Online]. Available: http://us.100y.com.tw/viewproduct.asp?MNo=33000. [Accessed: 03-Feb-2019].

K. Carman and C. Buxton, "nanoLC Programmable Logic Controller (PLC) Manual Including nanoNavigator Software." .

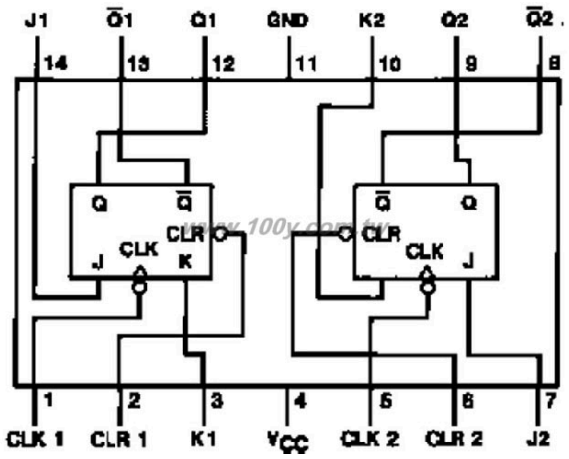
Admin, "SN74LS86AN DOWNLOAD," *Atelier*, 18-Oct-2018. [Online]. Available: http://atelier.cc/sn74ls86an-42/. [Accessed: 03-Feb-2019].

Pandey, Harshita. "Magnitude Comparator in Digital Logic." *GeeksforGeeks*, 25 Nov. 2019, www.geeksforgeeks.org/magnitude-comparator-in-digital-logic/.

8. Appendices

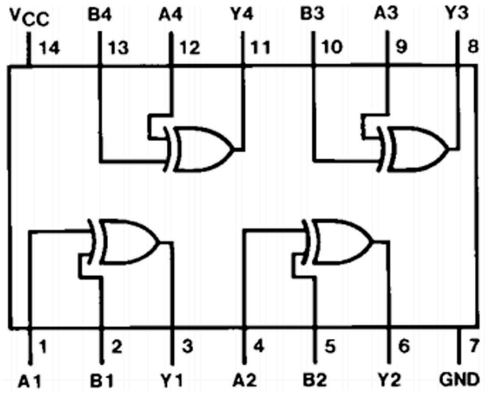
Diagrams used:

JK Flip Flop Chip Circuit Diagram



Source: http://us.100y.com.tw/viewproduct.asp?MNo=33000

XOR Chip Circuit Diagram



Source: http://atelier.cc/sn74ls86an-42/