[This is a lab report template with helpful comments in brackets like this. The principles shown in these comments are more important than the specific text]

[Cover Page, or if you like to save paper it is OK at the top of the first page like here.]

CPEN 230L: Introduction to Digital Logic Laboratory
Lab #: Lab Title
Firstname Lastname, Lab Date

[**Key Concept for what follows**: The **target reader** of your report is another student in the class that has the lab assignment and other lab handouts, but wasn't at the lab session and hasn't done the lab. They should be able to **replicate your results** from the lab handouts, your lab report, and the equipment you used. **There is no need to copy anything from the lab assignment or handouts unless you modified that content or you are explicitly asked for.** This frequently happens, if for example your instructor changes the assignment during the lab session or you found that you had to do things not detailed in the handouts. Your target reader will read the report from start to end, so organize it like this:

- Part 1 Pre-Lab Work, Procedure Changes, Results, Discussion
- Part 2 Pre-Lab Work, Procedure Changes, Results, Discussion
- …
- Summary and Conclusions]

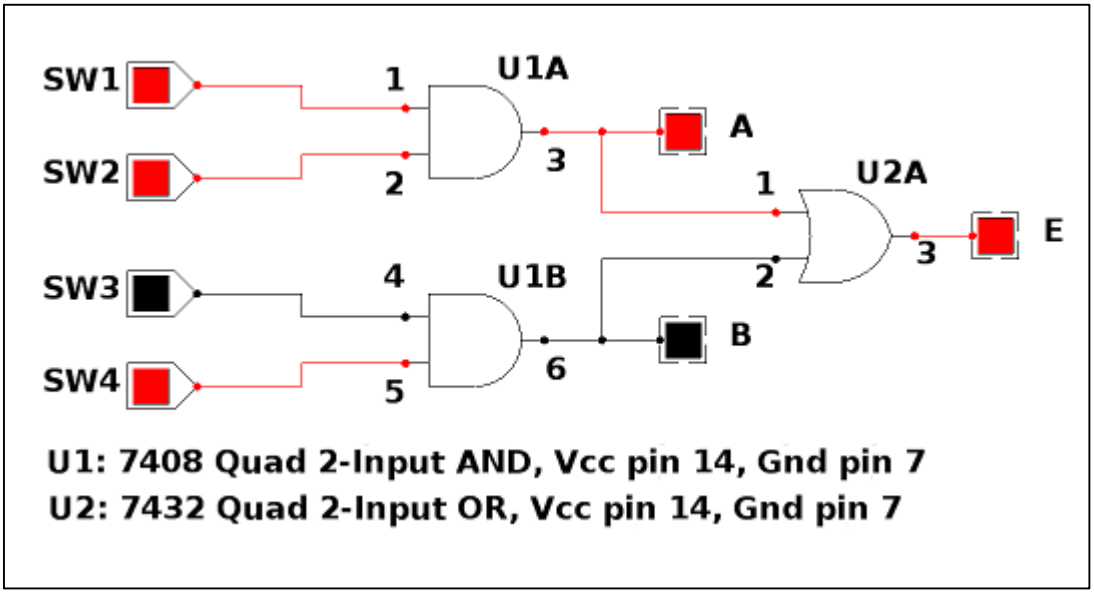
Part 1: Title

- Pre-Lab Work
  - Present the outcomes/deliverables of your pre-lab work. [Unless the lab. explicitly does not require any pre-lab work. Note: this is not going to happen very often!]
- Procedure Changes
  - I skipped … because … [an example deletion]
  - I found that … helps … [an example addition]
    [If you have nothing to say here, say "None". But helping your target reader have a smoother time doing the lab is rewarded. It also helps your instructor improve the lab handouts in the future.]
- Results: Here you should present the outcome of your during-lab work for example:

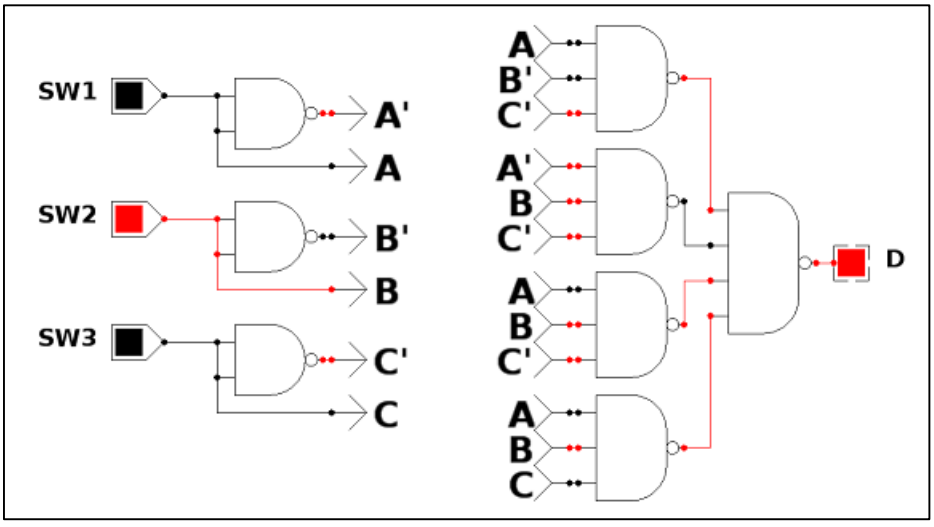
  - Truth table

SW1	SW2	SW3	SW4	A = SW1*SW2	B = SW3*SW4	E = SW1*SW2 + SW3*SW4
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

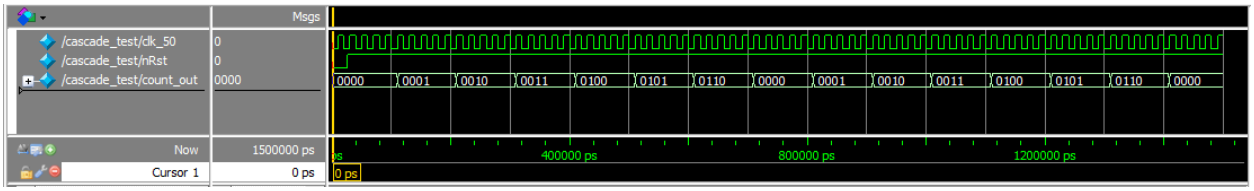
- Schematic diagram [example: E = SW1*SW2 + SW3*SW4, with A and B indicating mid-circuit values]



[Unless otherwise specified, schematic diagrams should include everything required by your reader to **understand and recreate your circuit**: Reference designators on each gate (U1A, U1B, U2A), pin numbers for all IC connections including Vcc and Ground, labels showing Logic Trainer I/O connections (SW1 to SW4, LEDs A, B, and E), a parts list describing each component, and anything else that will help your reader. For example, on more complicated circuits use To (-->) and From (>--) connectors to avoid many overlapping and confusing wires. This is shown on the following example that was part of pre-lab work not requiring pin numbers or other data.]



- ModelSim waveform – Notice that count_out counts 0 to 6 and then wraps around to 0, demonstrating that it is a mod-7 counter.



[The screen capture image has been cropped to show only the things of interest to the reader, specifically the waveforms and signal names in this case. The image is big enough to read the important details. (The "400000 ps" values on the time scale would be clearer if formatted as "400 ns".)]

- ModelSim Transcript window showing BCD count output from 0 through 15:

```
VSIM 3> run
# time  count BCD1:0
#    0      0    0 0
#   10      1    0 1
#   20      2    0 2
#   30      3    0 3
#   40      4    0 4
#   50      5    0 5
#   60      6    0 6
#   70      7    0 7
#   80      8    0 8
#   90      9    0 9
#  100     10    1 0
#  110     11    1 1
#  120     12    1 2
#  130     13    1 3
#  140     14    1 4
#  150     15    1 5
# ** Note: $finish
```

- Verilog code demonstrating two ways to code the same logic:

```
//CPEN 230L Lab 6a, Combinational Logic
//Rick Nungester, 10/15/15

module logic1(W, X, Y, Z);  // model A
output W;
input X, Y, Z;
wire P, Q;

not(P, Y);      // P = !Y
or(Q, X, P);    // Q = X || P
and(W, Q, Z);   // W = (X || !Y) && Z
endmodule  // logic1

/* commented-out alternate implementation
module logic1(W, X, Y, Z); // model B
output W;
input X, Y, Z;

assign W = (X || !Y) && Z; // same result as model A
endmodule // logic1
*/
```

[**Code formatting and comments are important!** The code above uses Courier New (fixed-width) 9-point font, left-justification, 2 spaces per indentation level, no too-long lines that wrap around from the right of the

page to the left, blank lines to separate logical sections, clear helpful comments, and a header showing class, title, name and date. Write code so your target reader can easily understand it and use it. Also notice that this short code example and its label "Verilog code demonstrating …" are all on one page. Avoid page breaks in code that is less than about ¾ of a page in length.]

[End of "Results" examples.]

- Discussion
    - The **IC Tester** verifies that chips are working well, before using them in breadboarded circuits. I had problems with getting it to work the first try because I forgot to put my chip in the socket as shown in the diagram next to the socket. Once I got it placed properly it worked as expected.
    - The Elenko XK-550 Digital/Analog Trainer ("**Logic Trainer**") was used to provide power to the IC, input signals from the switches, and output indications on its LEDs. Be sure to understand what wire holes are connected to each other in the breadboard. Getting this wrong (as I did) led to significant time wasted troubleshooting the circuit.
    - The ICs used (**7408 AND, 7432 OR**) worked as expected, but only after realizing I had pin 1 mis-identified so the IC was wired backwards.
    - Be sure to turn the Logic Trainer power switch on before using it.

[Use Discussion sections to explain the significance of what you just did. If testing is involved, make it clear how you tested your circuit or code and what the results were. For Results like simulation waveforms, explain how they demonstrate that the circuit is working correctly. Add anything that helps your target reader understand **why** they are doing this part of the lab. This will also show that you understand.]

Part 2: Title

- Procedure Changes
    - …
- Results
    - …
- Discussion
    - …

Summary and Conclusions

- This lab was really more about lab processes and report writing than the details of what an AND gate and OR gate do. The groundwork has been laid to move on to more complicated circuits and digital logic principles in future labs.
- It was very helpful reading the lab handout 5 days before the lab and knowing what to expect so the lab time could be spent solely on what had to be done in Herak 214.
- What could I have done better to improve my work effectiveness …
- What could I have done better to improve my work quality …

[Spend some time in Summary and Conclusions telling your target student reader anything else that might make the lab go smoother. Help them understand **what they were supposed to learn** in the lab. Be sure they have everything they need to **replicate your results** from just the lab assignment, other pre-lab handouts, and your report. Again, there is no need to repeat procedure steps given in the lab assignment and handouts.]

[Appendix

If possible put all data in the body of the report in the order your reader would want to encounter it – like in a textbook. But if necessary for full-page detailed images or long code listings, use an Appendix. If this is done, **a reference to the Appendix must be included in the main body of the report** (for example, "See the ModelSim Waveform results in the Appendix.") to indicate to your reader "quit reading here, go see the Appendix, then return to reading here".]

[Other things to keep in mind when writing reports:

- **Proofread**: Play the role of the target reader and read your report to yourself completely from start to end before handing it in. Imagine being your target student reader and sitting in the lab alone with only the pre-lab handouts and your report. Will that reader be able to do everything you did? Eliminate spelling and grammar errors. Check that your truth tables, schematic diagrams, state diagrams, screenshots, photographs, etc. show what you want them to show.
- **Re-read this template** before starting each lab report.