LABORATORY #3
DIGITAL LOGIC MINIMIZATION: EXCESS-3 TO 2-OF-5 CODE CONVERTER

Austin Guerrero
Ben Mettler
ECE 221 Digital Logic Design | Fall 2026 | October 8, 2026
Department of Electrical and Computer Engineering
Valparaiso University

ABSTRACT

In this laboratory, the Excess-3 to 2-of-5 code converter from Design Project #2 was moved off the screen and onto a breadboard. The circuit followed the instructor-provided minimum-IC wiring diagram, which builds the multiple-output SOP equations from Minilog entirely out of NAND gates and inverters on six ICs: one 7404, three 7410s, and two 7400s. Four switches drove the inputs, and five LEDs displayed the outputs. The converter produced the correct 2-of-5 code for all ten valid inputs. The longest path in the circuit passes through five gates on its way to output V, and its propagation delay was measured at 23.2 ns.

♦ INTRODUCTION

The goal of this lab is to implement the Excess-3 to 2-of-5 code converter with small scale integration (SSI) circuits [1]. Design Project #2 proved the logic in PSpice, but a simulation never has a loose wire, a bent pin, or a chip that runs a few nanoseconds slower than its datasheet claims. A breadboard has all three.

The converter takes a decimal digit in Excess-3 form and returns its five-bit 2-of-5 code. On the breadboard, the inputs were labeled A, B, C, and D and the outputs V, W, X, Y, and Z, matching the names Minilog assigned in Design Project #2 [2]. The lab had three parts: build the minimum-IC design with switches and LEDs, check all ten valid inputs, and measure the propagation delay of the longest path through the circuit.

♦ THEORY

The Minimum-IC Design

The wiring diagram provided for this lab (Fig. 3) is a NAND-NAND version of Minilog's multiple-output SOP result from Design Project #2 [2]:

$$V = A'C'D + A'C'D' + A'CD' + A'B'$$

$$W = BCD + B'C' + A'B'$$

$$X = BCD + A'C'D' + AC$$

$$Y = A'C'D + AD'$$

$$Z = A'CD' + AB + AD$$

Multiple-output mode earns its keep here. Five product terms, A′C′D, A′C′D′, A′CD′, A′B′, and BCD, each feed two outputs, so each one is built once and shared. Any SOP converts directly into two levels of NAND gates, which lets the whole circuit run on two gate types plus inverters.

Output V is the exception. It is a sum of four products, and no 4-input NAND was available, so it is built as

$$V = NAND(NOT(NAND(p_1, p_2, p_3)), p_4)$$

where p_1 through p_3 are the NAND outputs for the three A′ product terms and p_4 is the NAND output for A′B′. The workaround costs one 3-input NAND and one inverter, and, as the results show, two extra gate delays.

Switch and LED Interface

The inputs used the same switch circuit as Laboratory #2. A 10 kΩ resistor pulls each input up to Vcc, and closing the switch shorts it to ground, so an open switch reads as 1 and a closed switch reads as 0. Each output drives an LED through a 180 Ω resistor, which keeps the current within what a TTL output can safely supply. For any valid input, exactly two LEDs should light; one or three lit at once means something is wrong.

Propagation Delay

Every gate takes a few nanoseconds to respond, and those delays stack up along a path. The worst-case delay of the circuit is therefore set by the path with the most gate levels. Outputs W, X, Y, and Z are at most three levels deep: an inverter and two NANDs. Output V is five levels deep, so it sets the speed limit for the entire converter.

♦ PROCEDURE

The procedure is given in the Laboratory #3 handout [1]. The circuit photograph, oscilloscope capture, wiring diagram, and signed cover page are included in the Appendix.

I. The six ICs were seated across the center channel of the breadboard, and pins 14 (Vcc) and 7 (GND) of each chip were wired to the power rails. The bench supply was set to 5.05 V.

II. Inputs A through D were wired to switches with 10 kΩ pull-up resistors, as in Laboratory #2.

III. The gates were wired according to the diagram one output at a time, from V to Z, and each net was checked against the schematic as it went in.

IV. Outputs V through Z were connected to five LEDs, each through a 180 Ω current-limiting resistor (Fig. 1).

V. All ten valid Excess-3 inputs, 0011 through 1100, were set on the switches, and the lit LEDs for each input were compared against the truth table (Table 2).

VI. The longest path was identified from the schematic: an input passes through its inverter, a 3-input NAND, U3A, inverter U1E, and finally U5B before it reaches V.

VII. Input D was removed from its switch while A, B, and C were held at 0, 1, and 1. A 0 V to 5 V, 100 kHz square wave was checked on the oscilloscope before it was connected, as the manual warns. The circuit then toggled between digit 3 (0110) and digit 4 (0111), a change V only sees after all five gate levels.

VIII. Channel 1 probed input D and channel 2 probed output V. Cursors were placed on the rising input edge and the falling output edge to read the delay (Fig. 2).

♦ RESULTS

Table 1 lists the chips used by the implemented design. The packages are nearly full; only one inverter and one 3-input NAND are left over.

Table 1: IC Usage of the Implemented Design

Chip	Function	Gates Used	Gates per Chip	ICs
7404	Inverter	5	6	1
7410	3-input NAND	8	3	3
7400	2-input NAND	8	4	2
Total		**21**		**6**

Table 2 records the hardware test of every valid input. All ten produced the expected code, and exactly two LEDs lit each time. The working circuit was demonstrated to the instructor and signed off as functionally verified (Fig. 4).

Table 2: Hardware Test of All Valid Inputs

Digit	ABCD	Expected VWXYZ	Observed VWXYZ	Result
0	0011	11000	11000	✓
1	0100	10100	10100	✓
2	0101	10010	10010	✓
3	0110	10001	10001	✓
4	0111	01100	01100	✓
5	1000	01010	01010	✓
6	1001	01001	01001	✓
7	1010	00110	00110	✓
8	1011	00101	00101	✓
9	1100	00011	00011	✓

Table 3 lists the oscilloscope settings and readings for the longest path. The delay from the rising edge of input D to the falling edge of output V was 23.2 ns; the full capture is shown in Fig. 2.

Table 3: Longest-Path Propagation Delay Measurement

Quantity	Value
Channel 1 (input D)	5.00 V/div
Channel 2 (output V)	1.00 V/div
Time base, sample rate	40.0 ns/div, 2.50 GS/s
Trigger	Channel 1, rising, 5.10 V
Cursor b (input edge)	1.600 ns
Cursor a (output edge)	24.80 ns
Channel 1 amplitude (max)	6.40 V (7.20 V)

Quantity	Value
Channel 2 peak-to-peak	1.76 V
Propagation delay, Δt	**23.20 ns**

♦ DISCUSSION

Propagation Delay

Spread across five gate levels, 23.2 ns works out to about 4.6 ns per gate. That is quicker than the typical figures printed on 74LS datasheets, but it makes sense; datasheet numbers are written to cover heavy loading and the full temperature range, and a lightly loaded chip on a lab bench usually beats them.

The capture in Fig. 2 also shows that a breadboard is not a perfect conductor of good intentions. The input edge overshoots to 7.20 V and rings for roughly 100 ns before it settles. Output V shows a small bump at the instant the input rises, a full 23 ns before its real transition begins. Both come from the long jumper wires visible in Fig. 1, which behave like small inductors and couple neighboring signals together. Neither effect lasted long enough to flip a logic level, so the outputs stayed correct.

Comparison with Design Project #2

The circuit built here is not the one designed in Design Project #2. The instructor's version uses six ICs and 21 gates, all NANDs and inverters. My XOR-based design used five ICs and 19 gates by treating the don't cares as free logic [2]. Both beat the seven-IC AND-OR-NOT version comfortably, and both make the same point: once gates are bought four or six to a package, how well each package is filled matters more than the raw gate count.

The two designs also differ in speed. No output in the XOR design is more than three gate levels deep, while the instructor's design spends two extra levels on V to make up for the missing 4-input NAND. The extra chip is not the only price of the six-IC version; it is also the slower of the two.

Questions

1. Other than component cost, why would you want to limit the number of ICs that you use?

Every IC carries overhead beyond its price. Each chip needs its own power and ground connections, takes up board space, and adds 14 pins that must be seated, wired, or soldered correctly. Fewer chips mean fewer connections, and fewer connections mean fewer places for a fault to hide; one loose jumper in Fig. 1 could easily cost an hour of searching. Every package also draws supply current whether its gates are busy or not, so a half-empty chip wastes power and adds heat. On a printed circuit board, fewer packages allow a smaller board, shorter traces, and simpler routing, and signals that stay inside one package travel shorter distances, which helps both speed and noise.

2. Are there disadvantages to minimizing the number of gates in a design?

Yes. The fewest gates rarely means the fewest chips; the gate-minimized design in Design Project #2 needed only 22 gates but still filled seven ICs [2]. Minimization also tends to factor logic into more levels, and every level adds delay. Output V is the proof: sharing product terms saved gates, but it made V five levels deep and the slowest signal on the board.

A fully minimized SOP also drops the redundant terms that would otherwise cover hazards. The PSpice simulations in Design Project #2 showed short glitches whenever several inputs changed at once [2], which is exactly the behavior those missing terms allow. A design that leans on don't cares outputs whatever is convenient on the unused inputs, which is only safe if those inputs truly never occur. Finally, heavily shared terms increase fan-out, and a tightly packed circuit is harder to read, debug, and modify. A few extra gates can be a fair price for a circuit that works the first time.

Problems Encountered

The main challenge was simply managing the volume of wiring. Six chips and roughly sixty connections turned into the tangle seen in Fig. 1 faster than expected. Wiring one output at a time and checking each net before moving on kept errors from stacking up. The same tangle explains the ringing and crosstalk in Fig. 2; shorter, neatly routed wires would have cleaned up both.

♦ CONCLUSION

The six-IC NAND-NAND converter worked in hardware and produced the correct 2-of-5 code for all ten valid Excess-3 inputs. Its longest path, five gates from input D to output V, took 23.2 ns, which is fast enough that nobody flipping switches would ever notice. The lab also made the trade-offs from Design Project #2 tangible. Sharing product terms saved both gates and chips, yet it pushed V two levels deeper than every other output. The leanest design on paper ended up carrying the slowest signal on the board; chip count, speed, and wiring effort all pull on the same circuit, and the engineer has to decide which one wins.

♦ ACKNOWLEDGEMENTS

The circuit was built and tested with my lab partner, Ben Mettler. The objective, procedure, and questions were given and paraphrased from the ECE 221 Laboratory #3 handout [1]. The wiring diagram in Fig. 3 was provided by the course instructor, and the equations and comparison design come from my Design Project #2 report [2].

♦ REFERENCES

[1] Dept. of Electrical and Computer Engineering, *ECE 221: Digital Logic Design Laboratory & Design Projects User Manual*, Laboratory #3. Valparaiso, IN, USA: Valparaiso University.

[2] A. Guerrero, "Design Project #2: Logic Minimization: Excess-3 to 2-of-5 Code Converter," ECE 221, Valparaiso University, Oct. 8, 2026.

APPENDIX

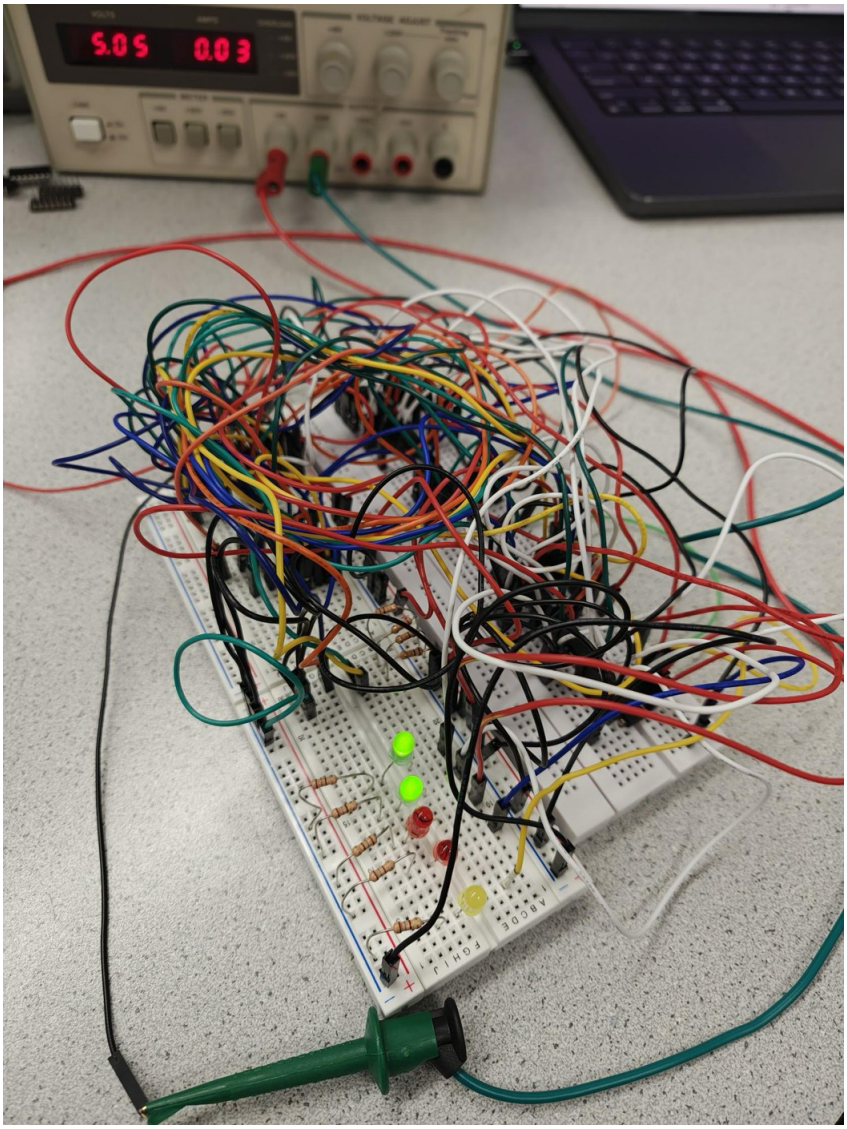


Fig 1. ***Breadboard implementation of the six-IC converter at 5.05 V and 0.03 A, with two of five output LEDs lit***

APPENDIX

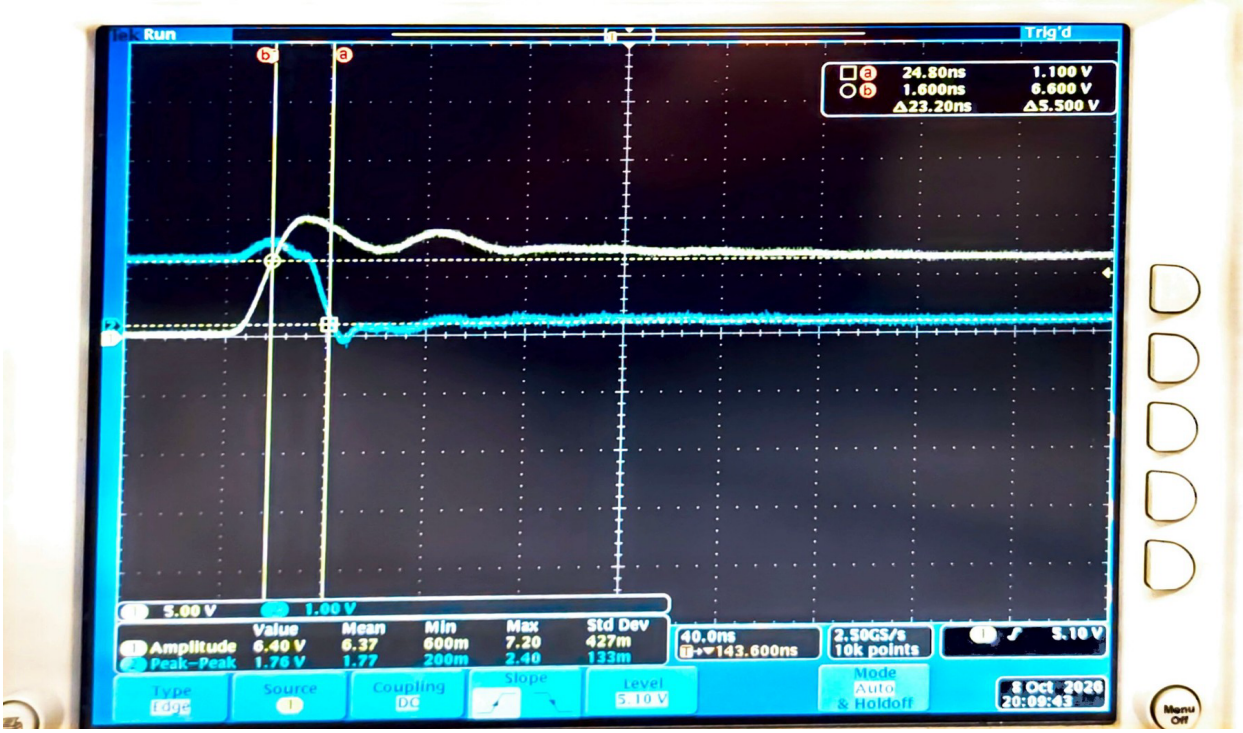


Fig 2. ***Longest-path propagation delay, input D (channel 1, white) rising to output V (channel 2, cyan) falling: Δt = 23.20 ns***

APPENDIX

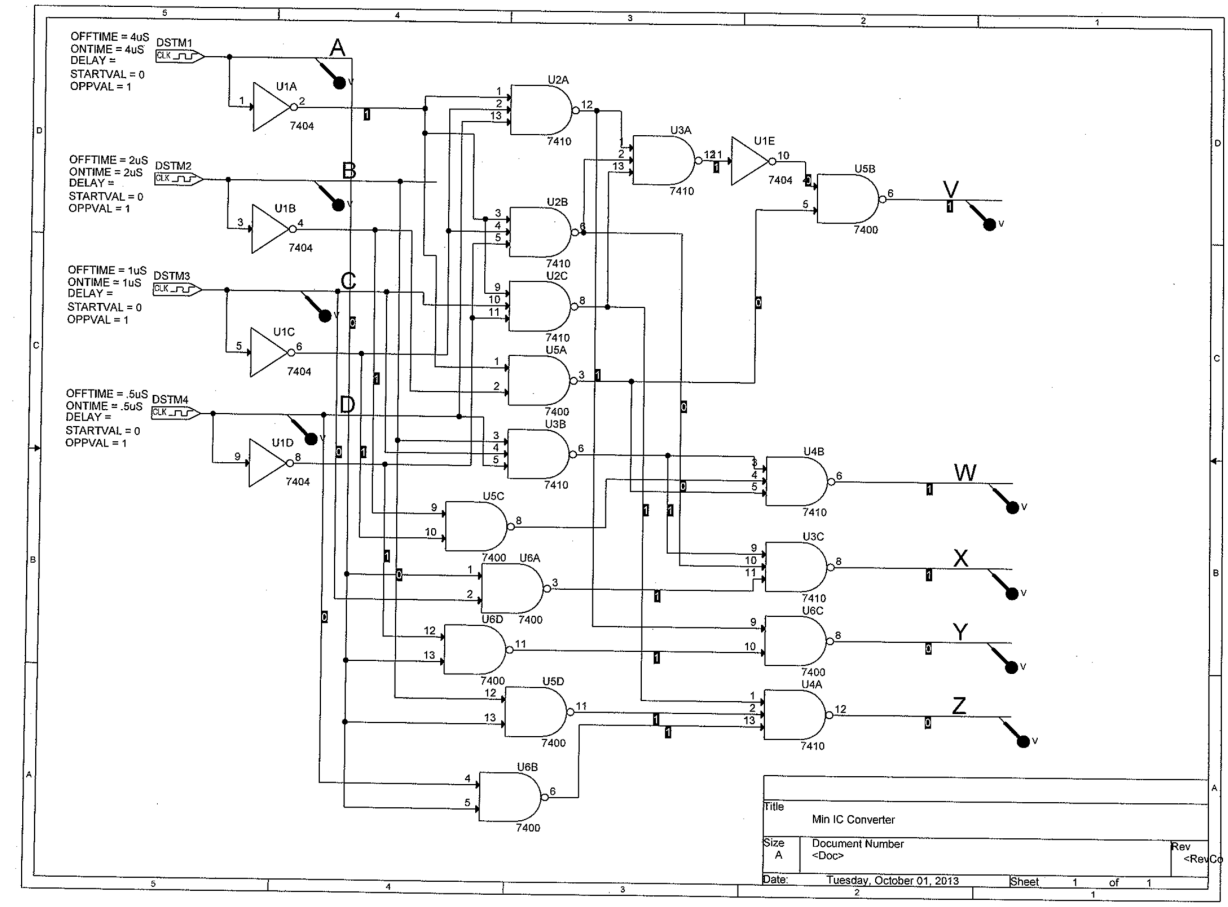


Fig 3. ***Instructor-provided minimum-IC wiring diagram (7404 ×1, 7410 ×3, 7400 ×2)***

Valparaiso University
Department of Electrical and Computer Engineering
ECE 221: Digital Logic Design

Laboratory #3

Name: Austin

Honor Code Pledge:

"I have neither given nor received nor tolerated others' use of unauthorized aid."

Signature:

Assignment milestones:

Project functionally verified

Project is partially verified

23