

# Project One

Benny Wen  
CSC 21100  
11/4/25

## Function Specification

The `toUpperCase()` function converts the lowercase ASCII letters to their corresponding uppercase letter. An 8-bit input is taken in representing an ASCII character from values 0 to 255 and produces an 8-bit output.

The function works in the following stages:

1. Check if the input is a lowercase letter (if the input ASCII values are between 97-122, for the letters 'a' through 'z')
2. If yes, subtract 32 from the input value to get the uppercase version
3. If no, output the input unchanged

Example:

- Input 97 ('a') > outputs 65 ('A')
- Input 65 ('A') > outputs 65 ('A') since the input is not between 97-122, already an uppercase letter
- Input 48 ('0') > outputs 48 ('0') since it's not a letter

# Circuit Design

Input:  $I_7 I_6 I_5 I_4 I_3 I_2 I_1 I_0$  (8 bits)

Output:  $O_9 O_8 O_7 O_6 O_5 O_4 O_3 O_2 O_1 O_0$  (8 bits)

Lower 5 bits:

$$A = I_4$$

$$B = I_3$$

$$C = I_2$$

$$D = I_1$$

$$E = I_0$$

Lowercase detection, bits  $I_7 I_6 I_5$  must equal 011, and lower 5 bits (A-E) must represent values 1-26

Range check ABCDE:

$$\text{InRange}(A, B, C, D, E) = \sum m(1, 2, 3, 4, 5, 6, 7, \dots, 26)$$

Expanded canonical:

$$\begin{aligned} \text{InRange} = & \bar{A}\bar{B}\bar{C}\bar{D}E + \bar{A}\bar{B}\bar{C}D\bar{E} + \bar{A}\bar{B}\bar{C}DE + \bar{A}\bar{B}C\bar{D}E + \bar{A}\bar{B}C\bar{D}\bar{E} + \bar{A}\bar{B}CDE + \bar{A}\bar{B}C\bar{D}\bar{E} + \\ & \bar{A}\bar{B}\bar{C}\bar{D}\bar{E} + \bar{A}\bar{B}\bar{C}D\bar{E} + \bar{A}\bar{B}\bar{C}DE + \bar{A}\bar{B}C\bar{D}\bar{E} + \bar{A}\bar{B}C\bar{D}E + \bar{A}\bar{B}CD\bar{E} + \bar{A}\bar{B}CDE + \\ & \bar{A}\bar{B}\bar{C}\bar{D}\bar{E} + A\bar{B}\bar{C}DE + A\bar{B}\bar{C}D\bar{E} + A\bar{B}\bar{C}D\bar{E} + A\bar{B}\bar{C}DE + A\bar{B}\bar{C}D\bar{E} + A\bar{B}\bar{C}D\bar{E} + \\ & A\bar{B}\bar{C}D\bar{E} + A\bar{B}\bar{C}D\bar{E} + A\bar{B}\bar{C}D\bar{E} \end{aligned}$$

$$ISL_{\text{lower}} = \bar{I}_7 \cdot \bar{I}_6 \cdot \bar{I}_5 \cdot \text{InRange}$$

Unmin implementation

↳ 26 5 input AND gates (each per minterm)

↳ 1 26 input OR gate

↳ gates for upper bits check

K-Map A=0 (values 0-15)

	DE	00	01	11	10
BC	00	0	1	1	1
	01	1	1	1	1
	11	1	1	1	1
	10	1	1	1	1

all zeros =  $\bar{A}\bar{B}\bar{C}\bar{D}\bar{E}$  (detect value 0)

$$\text{AllZeros} = \bar{A}\bar{B}\bar{C}\bar{D}\bar{E}$$

$$\text{TooLarge} = ABC + ABCDE$$

$$\text{InRange} = (\text{AllZeros} + \text{TooLarge})'$$

$$ISL_{\text{lower}} = \bar{I}_7 \cdot \bar{I}_6 \cdot \bar{I}_5 \cdot \text{InRange}$$

$$O_7 = I_7$$

$$O_6 = I_6$$

$$O_5 = I_5 \leftarrow (ISL_{\text{lower}})$$

$$O_4 = I_4$$

$$O_3 = I_3$$

$$O_2 = I_2$$

$$O_1 = I_1$$

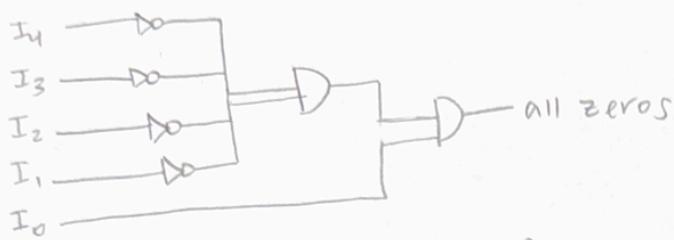
$$O_0 = I_0$$

A=1 (values 16-31)

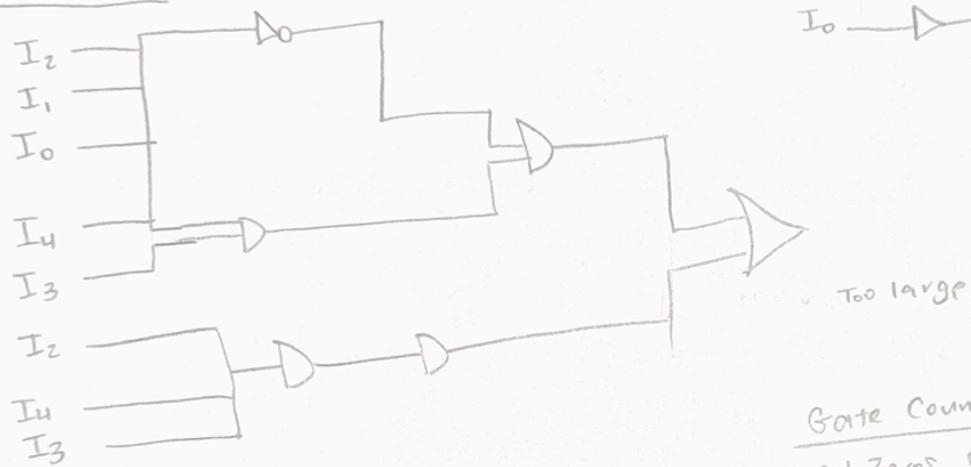
	DE	00	01	11	10
BC	00	1	1	1	1
	01	1	1	1	1
	11	0	0	0	0
	10	1	1	0	1

TooLarge =  $ABC + ABCDE$

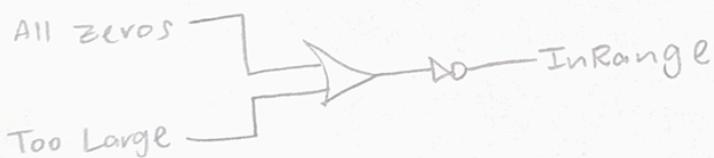
Circuit 1 (All zero detection)



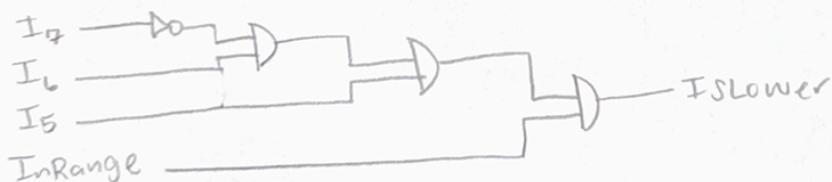
Circuit 2 (Too Large Detection)



Circuit 3 (In Range)



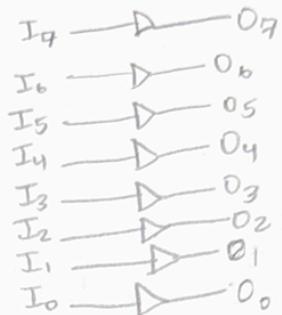
Circuit 4 (ISlower)



Circuit 5 (Output Bit 5)



Circuit 6



Gate Count

All Zeros, 5 NOT & 3 AND  
 TooLarge, 1 NOT & 4 AND & 1 OR  
 InRange, 1 OR & 1 NOT  
 ISlower, 1 NOT & 3 AND  
 Output, 7 BUF  
 Output(5), 1 NOT & 1 AND

29 gates used

# Minimum Delay Investigation

For toUpper\_tb.v, a safe inner input delay of 100 ns was used between test cases to ensure the circuit had sufficient time to settle.

To find the minimum delay, I adjusted the delay parameter in delay\_tb.v, starting from 100 ns and decreasing until failures occurred.

Delay (ns)	Outcome	Failed Tests	Notes
100	All pass	0	All conversions correct
65	Fail	3	Inputs 124, 123, 127 fail
90	Fail	1	Inputs 123 outputs 91 (wrong)
91	All pass	0	<b>Minimum Valid Delay Found</b>

Screenshots 3 & 4: Terminal output and waveform showing all tests pass with DELAY = 100 ns.

Screenshots 5 & 6: Terminal output and waveform showing 3 failures with DELAY = 65 ns.

Screenshots 7 & 8: Terminal output and waveform showing 1 failure with DELAY = 90 ns (input 123 outputted as 91 instead of 123)

Screenshots 9 & 10: Terminal output showing all tests pass with DELAY = 91 ns, confirming the minimum valid delay is 91 ns.

# Screenshots

## Screenshot 1

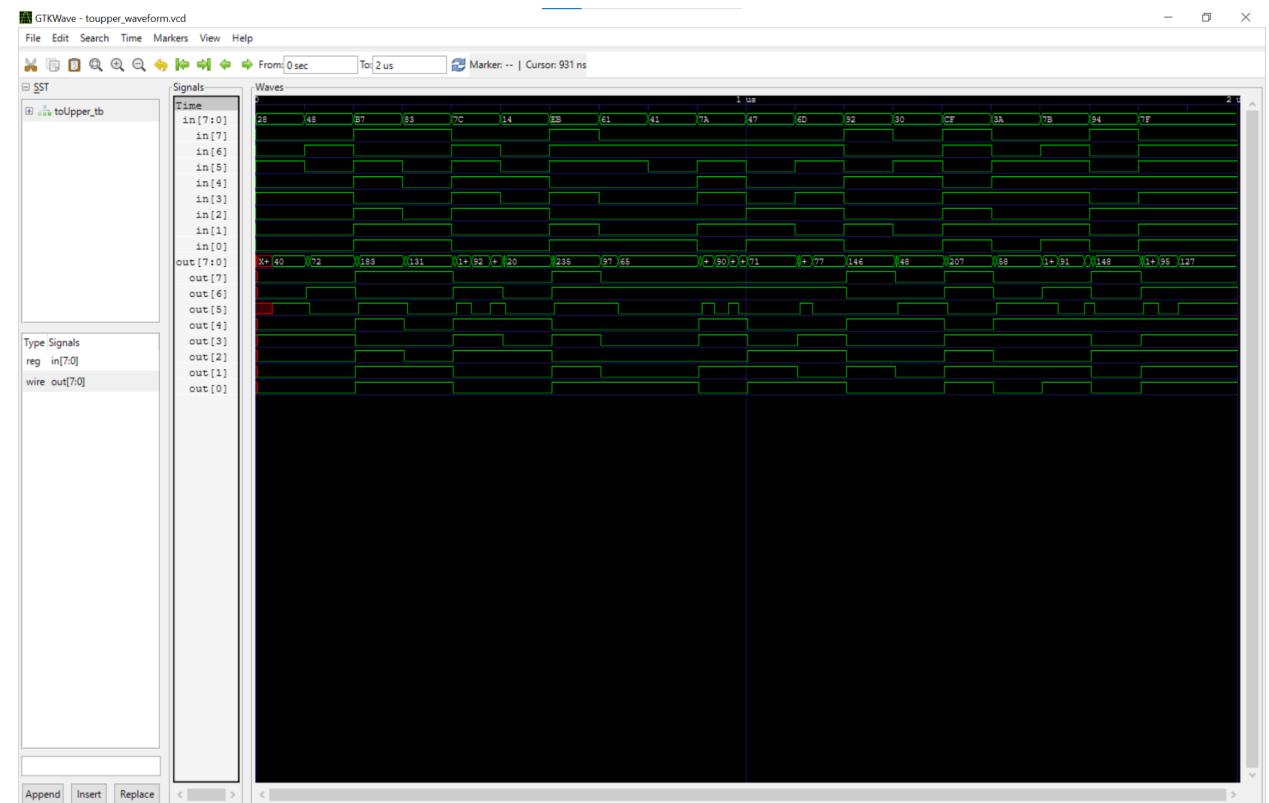
The screenshot shows a terminal window with the following content:

```
PS C:\Users\benny\Downloads\Project_One> rm *.vcd
PS C:\Users\benny\Downloads\Project_One> Remove-Item *.out, *.vcf
PS C:\Users\benny\Downloads\Project_One> iverilog -o sim toUpper.v toUpper_tb.v
PS C:\Users\benny\Downloads\Project_One> vvp sim
VCD info: dumpfile toupper_waveform.vcd opened for output.
Time Input Output Expected
==== ====== ====== ======
100000 40 40 40
200000 72 72 72
300000 183 183 183
400000 131 131 131
500000 124 124 124
600000 20 20 20
700000 235 235 235
800000 97 65 65
900000 65 65 65
1000000 122 90 90
1100000 71 71 71
1200000 109 77 77
1300000 146 146 146
1400000 48 48 48
1500000 207 207 207
1600000 58 58 58
1700000 123 123 123
1800000 148 148 148
1900000 127 127 127

Simulation complete!
toUpper_tb.v:97: $finish called at 2000000 (1ps)
PS C:\Users\benny\Downloads\Project_One> ]
```

- Screenshot 1 shows the output from running `toUpperCase_tb.v` with the default 100 ns delay between inputs. It shows all test cases pass and the output values match the expected ASCII conversions, proving the logic inside `toUpperCase.v` works as intended.

## Screenshot 2



- GTKWave Waveform view of the 100 ns functional test using toupper\_waveform.vcd. Where signals in[7:0] and out[7:0] are visible, shown in decimal format.

## Screenshot 3

The screenshot shows an IDE interface with multiple tabs open. The terminal tab displays the command-line session for running a simulation and capturing waveforms. The output table below shows the results of a test comparing input and output values over time.

```
File Edit View Go Run Terminal Help ← → Q Project_One
```

PROJECT\_ONE

- find\_min\_delay\_tb.v
- min\_delay\_test.vcd
- sim
- test\_sim
- toUpperCase\_tb.v
- toUpperCase\_waveform.vcd
- toUpperCase.v

TERMINAL

```
toUpperCase_tb.v:97: $finish called at 2000000 (ips)
PS C:\Users\benny\Downloads\Project_One> gtkwave toupper_waveform.vcd
>>
GTKWave Analyzer v3.3.118 (w)1999-2023 BSI

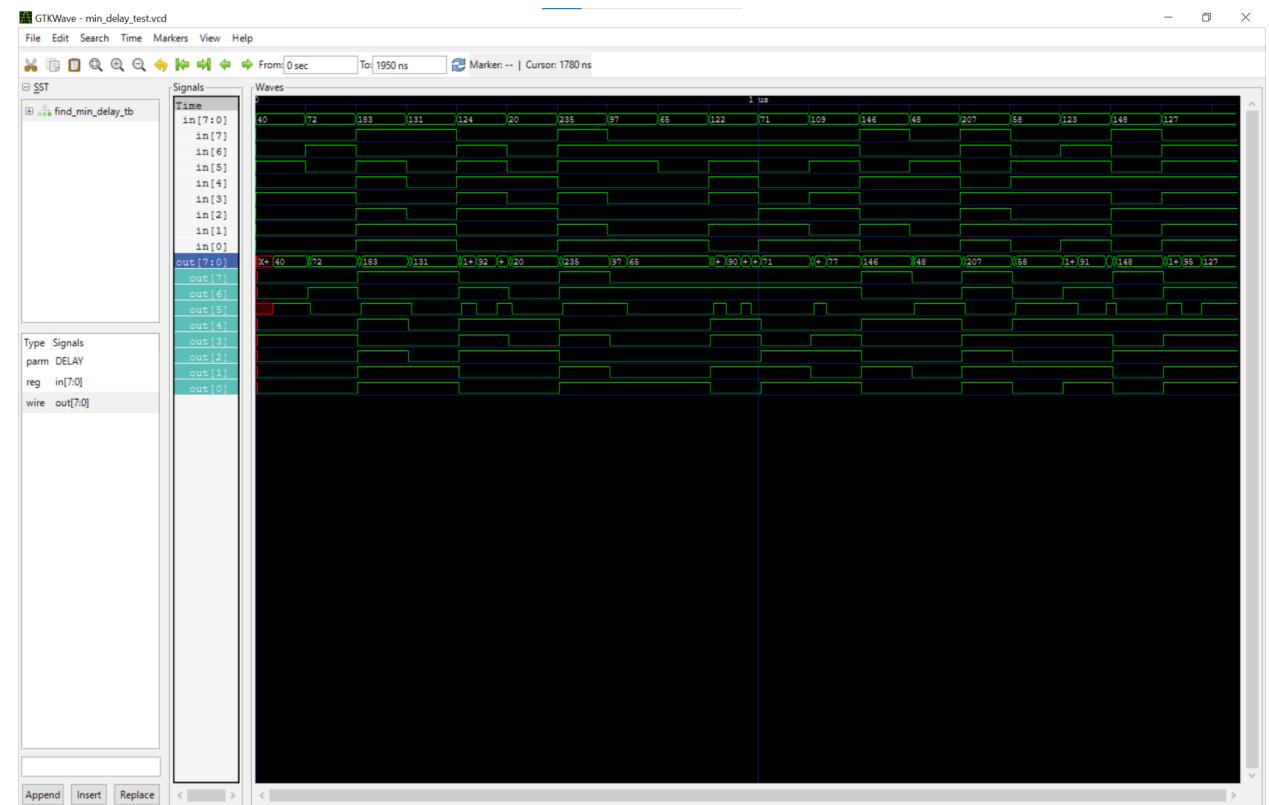
[0] start time.
[2000000] end time.
WM Destroy
PS C:\Users\benny\Downloads\Project_One> iverilog -o test_simtoUpperCase.v find_min_delay_tb.v
>>
PS C:\Users\benny\Downloads\Project_One> vvp test_sim
>>
VCD info: dumpfile min_delay_test.vcd opened for output.
=====
Testing with inter-input delay = 100 ns
=====
```

Time(ns)	Input	Output	Expected	Result
100000	40	40	40	PASS
200000	72	72	72	PASS
300000	183	183	183	PASS
400000	131	131	131	PASS
500000	124	124	124	PASS
600000	20	20	20	PASS
700000	235	235	235	PASS
800000	97	65	65	PASS
900000	65	65	65	PASS
1000000	122	98	98	PASS
1100000	71	71	71	PASS
1200000	189	77	77	PASS
1300000	146	146	146	PASS
1400000	48	48	48	PASS
1500000	287	287	287	PASS
1600000	58	58	58	PASS
1700000	123	123	123	PASS
1800000	148	148	148	PASS
1900000	127	127	127	PASS

```
=====
Test complete with delay = 100 ns
=====
find_min_delay_tb.v:85: $finish called at 1950000 (ips)
PS C:\Users\benny\Downloads\Project_One> ]
```

- Screenshot 3 shows the terminal output with all PASS values when parameter DELAY = 100 for find\_min\_delay\_tb.v

## Screenshot 4



- Screenshot 4 shows the GTKWave waveform view at 100 ns delay showing all signals settling properly between transitions.

## Screenshot 5

The screenshot shows a software interface with a terminal window displaying simulation results and a waveform viewer.

**Terminal Output:**

```
Test complete with delay = 100 ns
=====
find_min_delay_tb.v:85: $finish called at gtkwave min_delay_test.vcd
>> C:\Users\benny\Downloads\Project_One>

GTKWave Analyzer v3.3.118 (w)1999-2023 BSI

[0] start time.
[1950000] end time.
WM Destroy         iverilog -o test_sim toUpper.v find_min_delay_tb.v
>> C:\Users\benny\Downloads\Project_One>
PS C:\Users\benny\Downloads\Project_One> vvp test_sim
VCD info: dumpfile min_delay_test.vcd opened for output.
=====
Testing with inter-input delay = 65 ns
=====
```

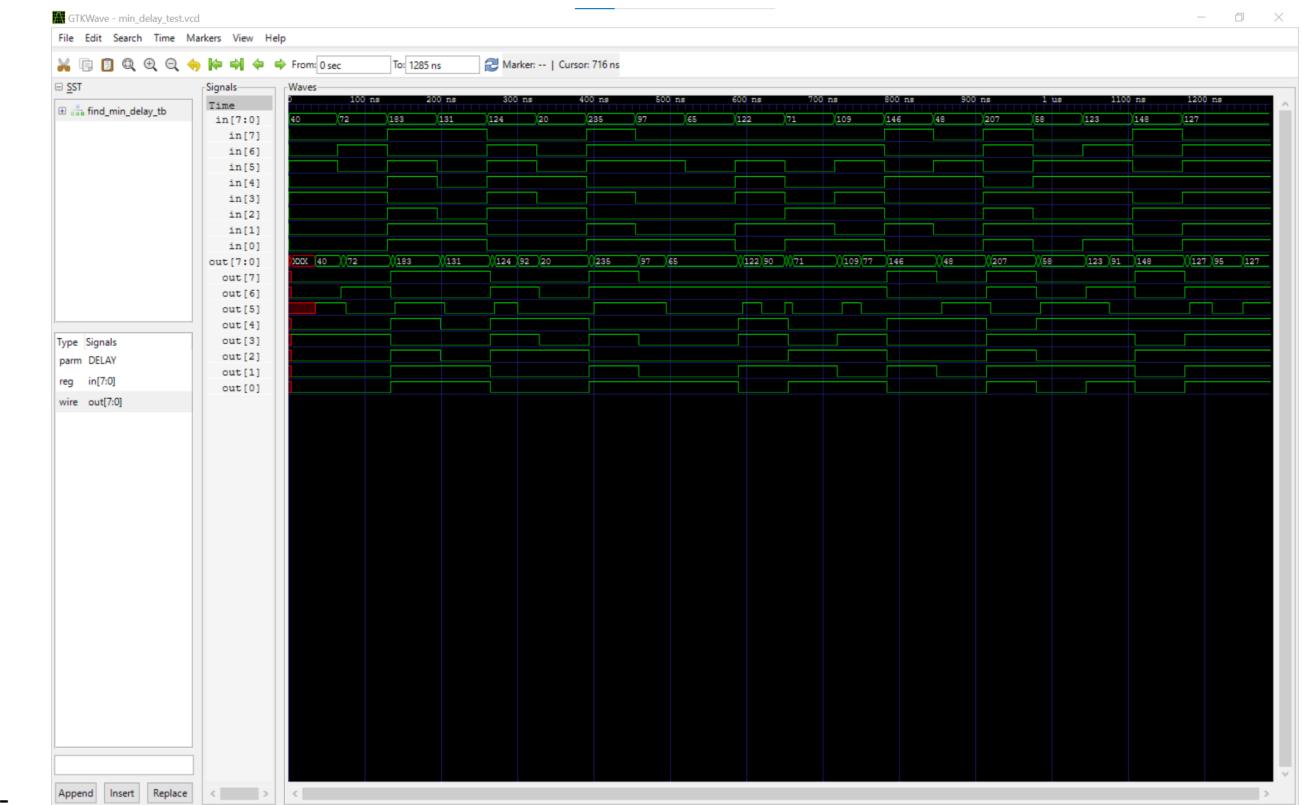
**Waveform Viewer:**

Time(ns)	Input	Output	Expected	Result
650000	40	48	48	PASS
1300000	72	72	72	PASS
1950000	183	183	183	PASS
2600000	131	131	131	PASS
3250000	124	92	124	FAIL
3900000	20	20	20	PASS
4550000	235	235	235	PASS
5200000	97	65	65	PASS
5850000	65	65	65	PASS
6500000	122	98	98	PASS
7150000	71	71	71	PASS
7800000	169	77	77	PASS
8450000	146	146	146	PASS
9100000	48	48	48	PASS
9750000	287	287	287	PASS
10400000	58	58	58	PASS
11050000	123	91	123	FAIL
11700000	148	148	148	PASS
12350000	127	95	127	FAIL

=====
Test complete with delay = 65 ns
=====
find\_min\_delay\_tb.v:85: \$finish called at 1285000 (1ps)
PS C:\Users\benny\Downloads\Project\_One>

- Screenshot 5 shows the terminal output when parameter DELAY = 65 ns with three failures: inputs 123, 124, and 127.

## Screenshot 6



- Screenshot 6 shows GTKWave waveform at 65 ns delay showing failed conversions.

## Screenshot 7

The screenshot shows a terminal window within a software interface. The terminal tab is active, displaying the following output:

```
Project_One
File Edit Selection View Go Run Terminal Help ← → ⌂ Project_One
find_min_delay_tb.v ×
find_min_delay_tb.v > {} find_min_delay_tb
=====
2
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS TEROSIDL LOG REPORT TEROSIDL TIMING
powershell + ⌂ find_min_delay_tb.v

=====
Test complete with delay = 65 ns
=====
find_min_delay_tb.v:85: $finish called at 1285000 (1ps)
PS C:\Users\benny\Downloads\Project_One> gtkwave min_delay_test.vcd

GTKWave Analyzer v3.3.118 (w)1999-2023 BSI

[8] start time,
[1285000] end time.
W# Destroy
PS C:\Users\benny\Downloads\Project_One> iverilog -o test_sim toUpper.v find_min_delay_tb.v
PS C:\Users\benny\Downloads\Project_One> vvp test_sim
VCD info: dumpfile min_delay_test.vcd opened for output.
=====
Testing with inter-input delay = 90 ns
=====

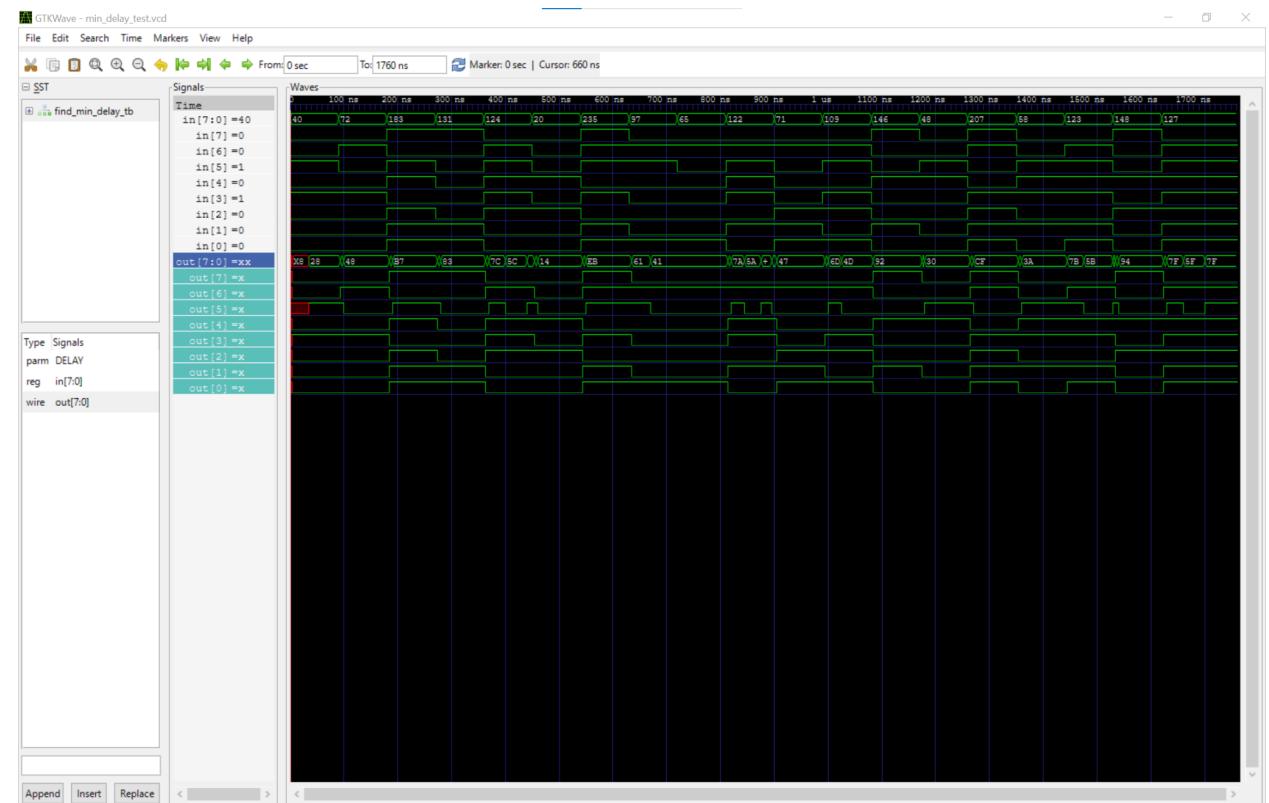
Time(ns) Input Output Expected Result
----- ---- ----- -----
90000 40 40 40 PASS
180000 72 72 72 PASS
270000 183 183 183 PASS
360000 131 131 131 PASS
450000 124 124 124 PASS
540000 20 20 20 PASS
630000 235 235 235 PASS
720000 97 65 65 PASS
810000 65 65 65 PASS
900000 122 96 96 PASS
990000 71 71 71 PASS
1080000 189 77 77 PASS
1170000 146 146 146 PASS
1260000 48 48 48 PASS
1350000 287 287 287 PASS
1440000 58 58 58 PASS
1530000 123 91 123 FAIL
1620000 148 148 148 PASS
1710000 127 127 127 PASS

=====
Test complete with delay = 98 ns
=====
find_min_delay_tb.v:85: $finish called at 1760000 (1ps)
PS C:\Users\benny\Downloads\Project_One>
```

At the bottom of the terminal window, status indicators show: In 14, Col 18, Spaces: 4, UTF-8, CR/LF, Verilog, Prettier.

- Screenshot 7 shows terminal output when parameter DELAY = 90 ns showing a single failure (input 123 outputs as 91 instead of 123).

## Screenshot 8



- Screenshot 8 shows GTKWave waveform failure when parameter  $\text{DELAY} = 90 \text{ ns}$ .

## Screenshot 9

The screenshot shows a terminal window within a software interface. The terminal window title is "Project\_One". The command run is "vvp test\_sim". The output shows VCD info for a dumpfile and testing with an inter-input delay of 91 ns. A table of test results is displayed:

Time(ns)	Input	Output	Expected	Result
91000	40	40	40	PASS
182000	72	72	72	PASS
273000	183	183	183	PASS
364000	131	131	131	PASS
455000	124	124	124	PASS
546000	20	20	20	PASS
637000	235	235	235	PASS
728000	97	65	65	PASS
819000	65	65	65	PASS
910000	122	90	90	PASS
1001000	71	71	71	PASS
1092000	189	77	77	PASS
1183000	146	146	146	PASS
1274000	48	48	48	PASS
1365000	287	287	287	PASS
1456000	58	58	58	PASS
1547000	123	123	123	PASS
1638000	148	148	148	PASS
1729000	127	127	127	PASS

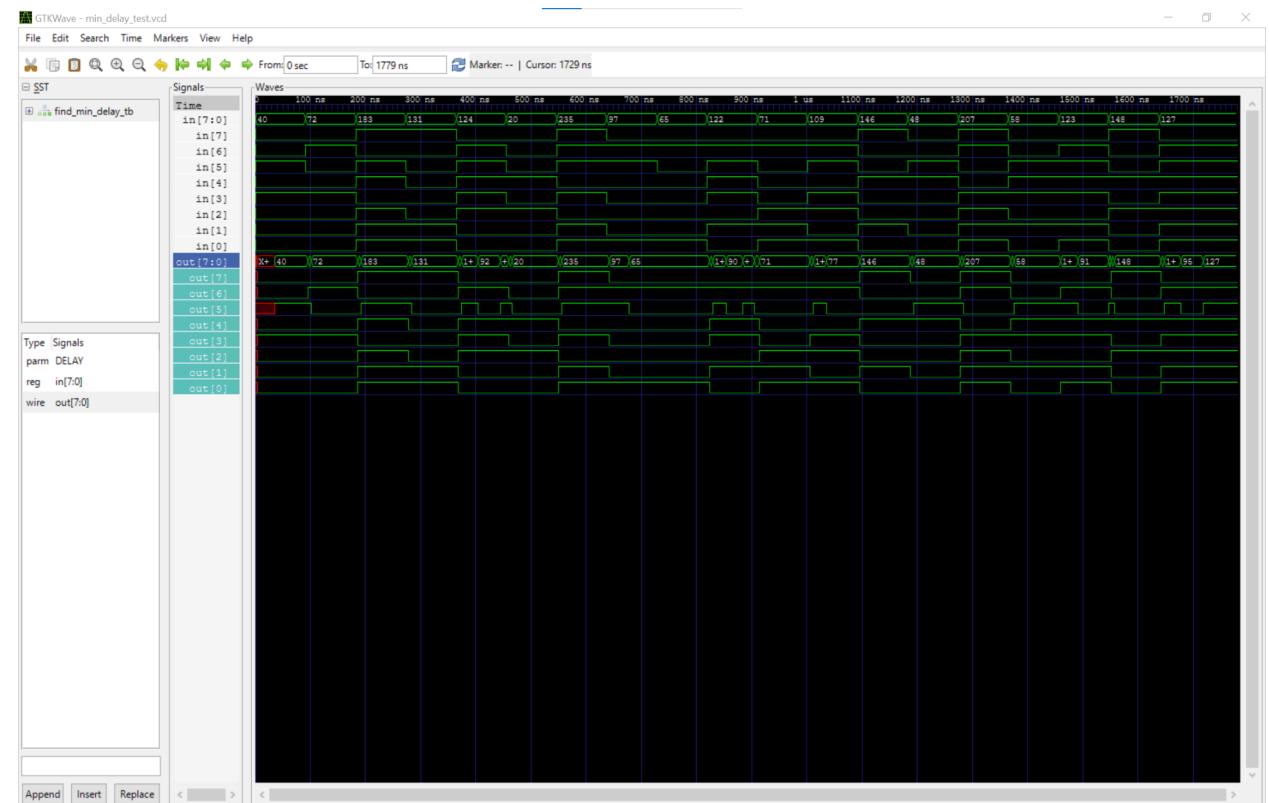
Test complete with delay = 91 ns

find\_min\_delay\_tb.v:85: \$finish called at 1779000 (ips)

PS C:\Users\benny\Downloads\Project\_One> [ ]

- Screenshot 9 shows terminal output when parameter **DELAY** = 91 ns with all 19 test cases passing. This confirms 91 ns as the minimum delay.

## Screenshot 10



- Screenshot 10 shows GTKWave waveform at 91 ns delay showing correct operations.