Computer Architecture Laboratory

1st Experiment: 3-Digit BCD Adder

Name	Student #
Amirreza Aranpour	99170348
Ali Safarafard	99105583
Soroush Sherafat	99105504
Mehrad Milanloo	99105775

Design

Validating Input

At first, we know that if a digit of any of inputs is greater than 9, then the input is not a valid BCD number. So we design a `INPUT_VALIDATOR` subcircuit which generates an `IS_INPUT_VALID` output bit.

To implement this subcircuit we need to implement a `IS_GT9` subcircuit first. This subcircuit must return a single bit, showing wether a 5-bit input is greater than 9 or not. Here's the truth table:

Number	a_4	a_3	a_2	a_1	a_0	out
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	1	0	0
3	0	0	0	1	0	0
4	0	0	1	0	0	0
5	0	0	1	0	0	0
6	0	0	1	1	0	0
7	0	0	1	1	0	0
8	0	1	0	0	0	0
9	0	1	0	0	0	0
10	0	1	0	1	0	1
11	0	1	0	1	0	1
12	0	1	1	0	0	1
13	0	1	1	0	0	1
14	0	1	1	1	0	1
15	0	1	1	1	0	1
16	1	0	0	0	0	1
17	1	0	0	0	0	1
18	1	0	0	1	0	1
19	1	0	0	1	0	1
20	1	0	1	0	0	1

Number	a_4	a_3	a_2	a_1	a_0	out
21	1	0	1	0	0	1
22	1	0	1	1	0	1
23	1	0	1	1	0	1
24	1	1	0	0	0	1
25	1	1	0	0	0	1
26	1	1	0	1	0	1
27	1	1	0	1	0	1
28	1	1	1	0	0	1
29	1	1	1	0	0	1
30	1	1	1	1	0	1
31	1	1	1	1	0	1

Here's the equation:

$$out := a_4 \lor (a_3 \land a_2) \lor (a_3 \land a_1)$$

Adding

We realize that just like binary adders, n-digit BCD adders are composed of n 1-digit BCD adders with connected Cins and $Cout$s and therefore, we need to design 1-digit BCD adders.

A 1-digit BCD adder is a 4-bit binary adder under the hood, with one difference: if the sum is greater than 9, we will output $Sum - 10$ as Sum and 1 as $Cout$. We reuse the IS_GT9 subcircuit to detect these cases.

To generate $Sum - 10$, we use another 4-bit binary adder. Here is its second operand's truth table:

$gt9$	out_3	out_2	out_1	out_0
0	0	0	0	0
1	0	1	1	0

So we get the following equations:

$$out_3 := 0, out_2 := gt9, out_1 := gt9, out_0 := 0$$

To design a 4-bit binary adder, we need full adders and to design full adders, we need half adders.
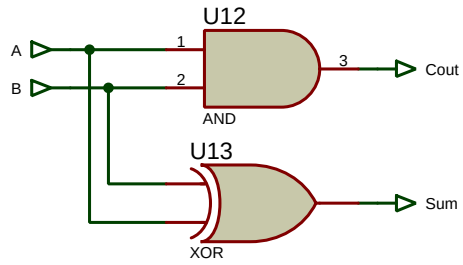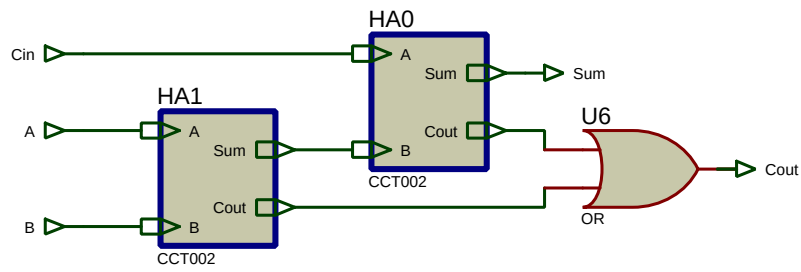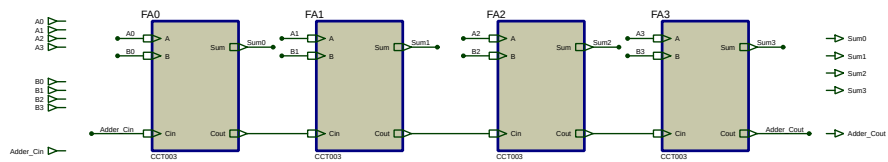
Modules

Figure 1: `HA` module.

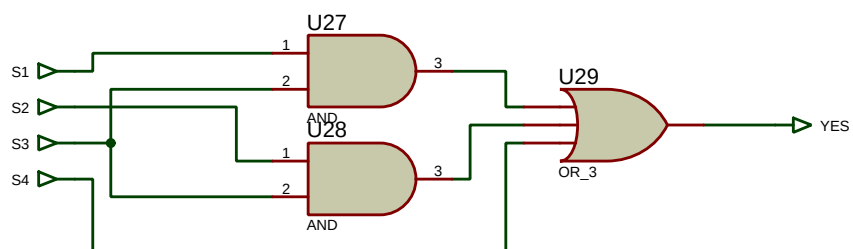Figure 2: `FA` module.

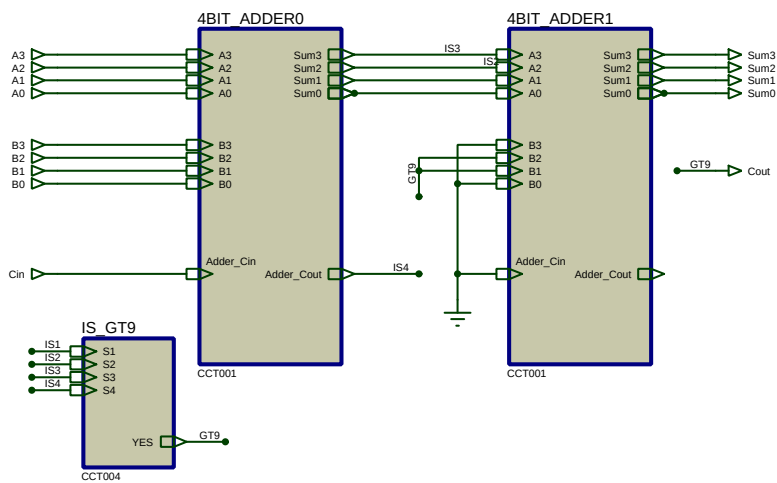Figure 3: `4BIT_ADDER` module.

Figure 4: `IS_GT9` module.

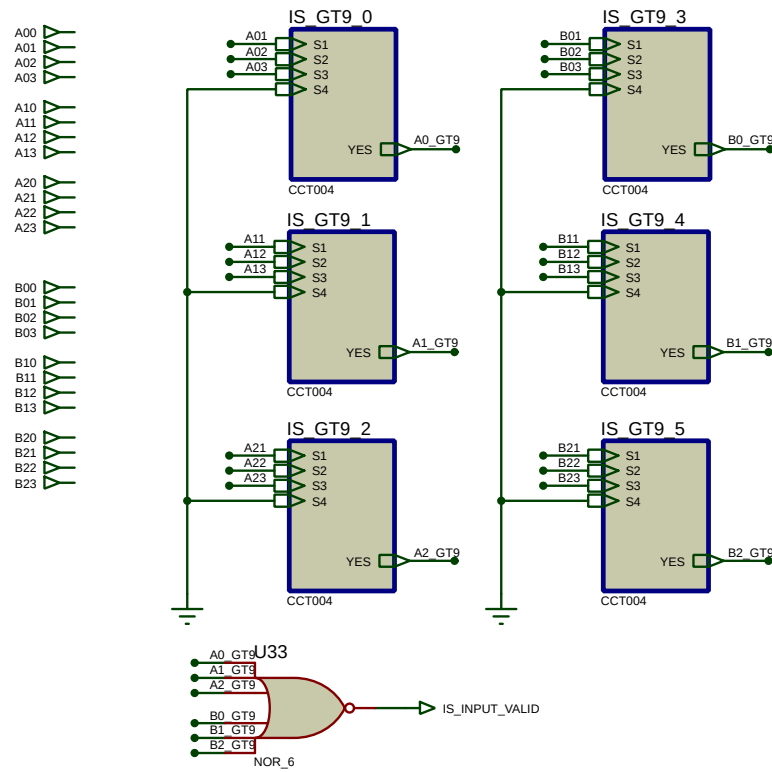Figure 5: `1DIGIT_BCD_ADDER` module.

A00
A01
A02
A03

A10
A11
A12
A13

A20
A21
A22
A23

B00
B01
B02
B03

B10
B11
B12
B13

B20
B21
B22
B23

IS_GT9_0

A01 — S1
A02 — S2
A03 — S3
— S4

YES — A0_GT9

CCT004

IS_GT9_1

A11 — S1
A12 — S2
A13 — S3
— S4

YES — A1_GT9

CCT004

IS_GT9_2

A21 — S1
A22 — S2
A23 — S3
— S4

YES — A2_GT9

CCT004

IS_GT9_3

B01 — S1
B02 — S2
B03 — S3
— S4

YES — B0_GT9

CCT004

IS_GT9_4

B11 — S1
B12 — S2
B13 — S3
— S4

YES — B1_GT9

CCT004

IS_GT9_5

B21 — S1
B22 — S2
B23 — S3
— S4

YES — B2_GT9

CCT004

U33

A0_GT9
A1_GT9
A2_GT9

B0_GT9
B1_GT9
B2_GT9

IS_INPUT_VALID

NOR_6

Figure 6: INPUT_VALIDATOR module.

5

1DIGIT_BCD_ADDER0

A03 A3 Sum3 Sum03
A02 A2 Sum2 Sum02
A01 A1 Sum1 Sum01
A00 A0 Sum0 Sum00

B03 B3
B02 B2
B01 B1
B00 B0

Cin Cin Cout

CCT005

1DIGIT_BCD_ADDER1

A13 A3 Sum3 Sum13
A12 A2 Sum2 Sum12
A11 A1 Sum1 Sum11
A10 A0 Sum0 Sum10

B13 B3
B12 B2
B11 B1
B10 B0

Cin Cout

CCT005

1DIGIT_BCD_ADDER2

A23 A3 Sum3 Sum23
A22 A2 Sum2 Sum22
A21 A1 Sum1 Sum21
A20 A0 Sum0 Sum20

B23 B3
B22 B2
B21 B1
B20 B0

Cin Cout Adder_Cout

CCT005

A00
A01
A02
A03

A10
A11
A12
A13

A20
A21
A22
A23

B00
B01
B02
B03

B10
B11
B12
B13

B20
B21
B22
B23

Cin

INPUT_VALIDATOR

A00 IS_INPUT_VALID IS_INPUT_VALID
A01
A02
A03

A10
A11
A12
A13

A20
A21
A22
A23

B00
B01
B02
B03

B10
B11
B12
B13

B20
B21
B22
B23

CCT007

Sum00
Sum01
Sum02
Sum03

Sum10
Sum11
Sum12
Sum13

Sum20
Sum21
Sum22
Sum23

Adder_Cout

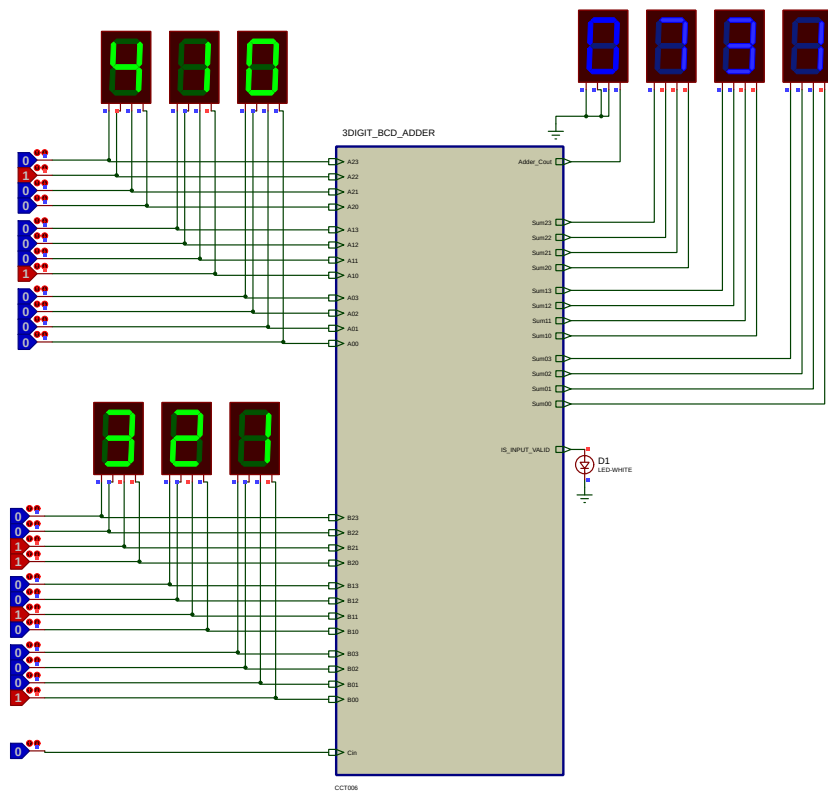Figure 7: **3DIGIT_BCD_ADDER** module.
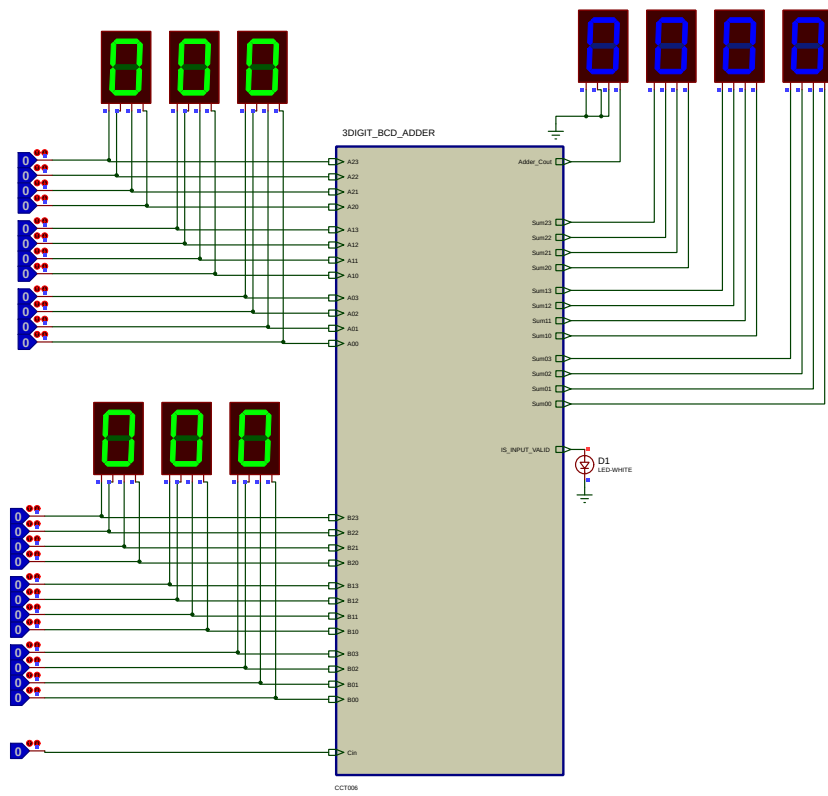
Figure 8: `TOP` module.

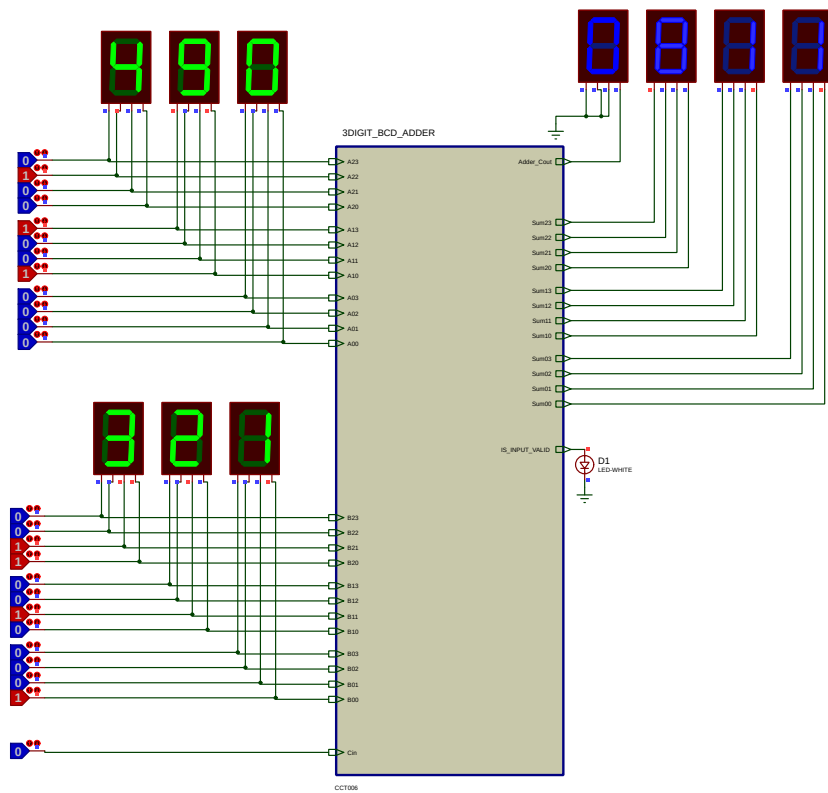Figure 9: TOP module, performing a simple addition.

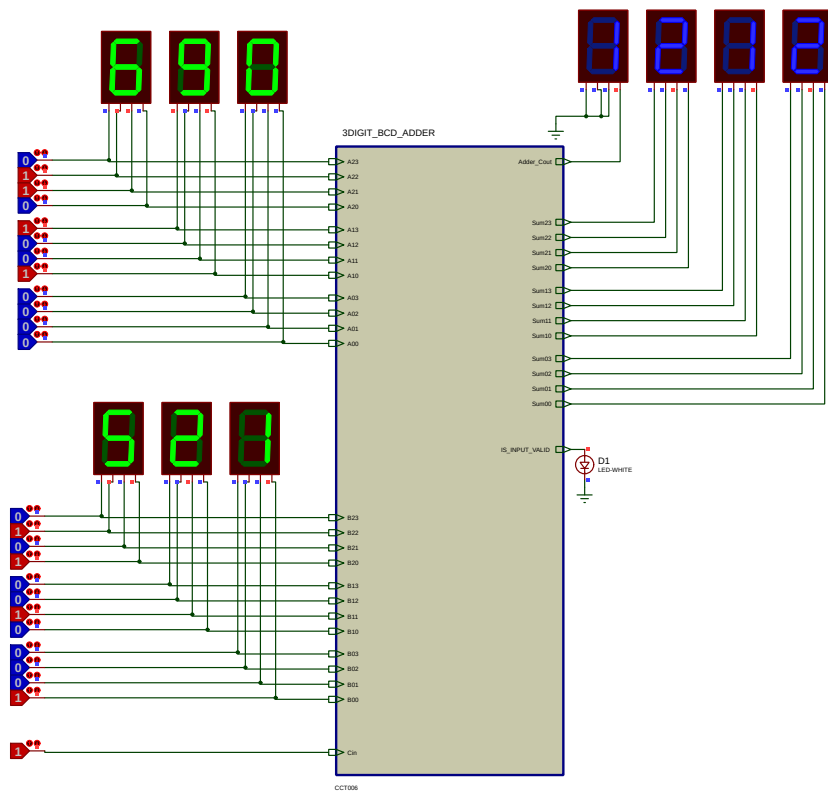Figure 10: TOP module, performing an addition with a carry.

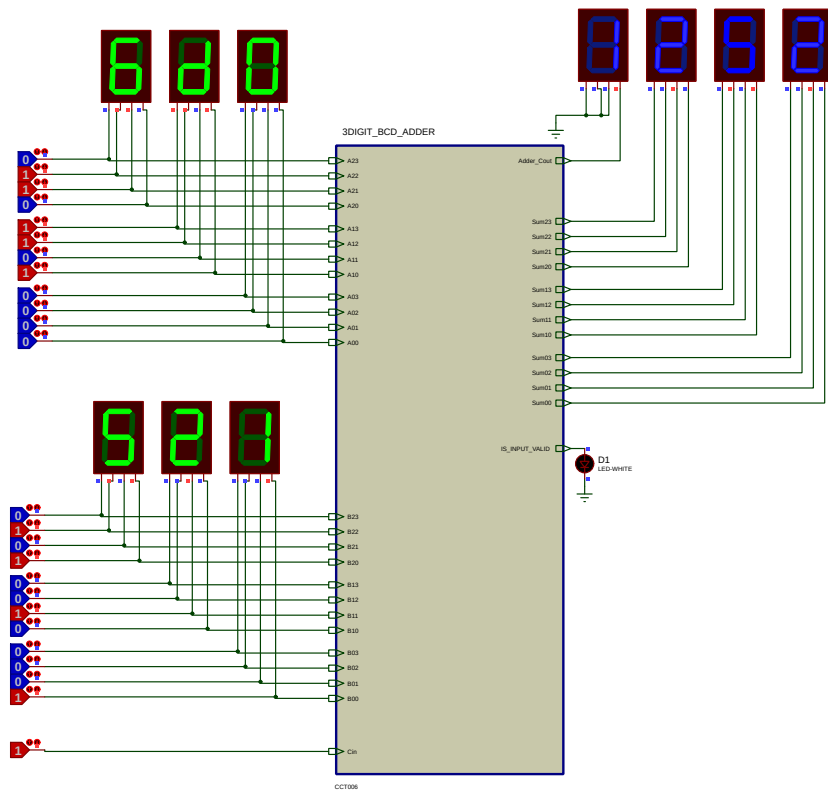Figure 11: TOP module, performing an addition with all the carry bits enabled.

Figure 12: TOP module, setting IS_INPUT_VALID to 0 because one of the input digits is greater than 10.