ES 4 lab 1: Saturating adder

Ben Bitdiddle
26 January 2019
10-12am Thursday section, Prof. Bell

The objective of this lab is to design and build a 2-bit saturating adder using 74-series logic ICs on a breadboard. The inputs are DIP switches, and the output is 2 LEDs.

Design

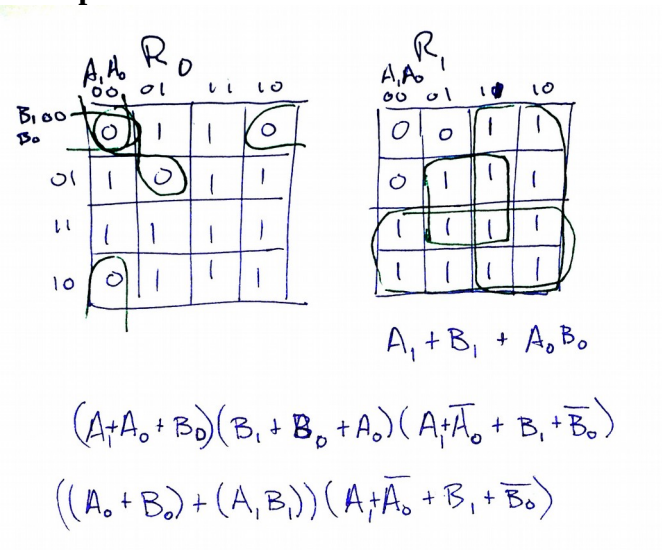
Desired operation (decimal):

A	B	R
0	0	0
0	1	1
0	2	2
0	3	3
1	0	1
1	1	2
1	2	3
1	3	3
2	0	2
2	1	3
2	2	3
2	3	3
3	0	3
3	1	3
3	2	3
3	3	3

Truth table (binary)

A1	A0	B1	B0	R1	R0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	1	1
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	1

K-map work:

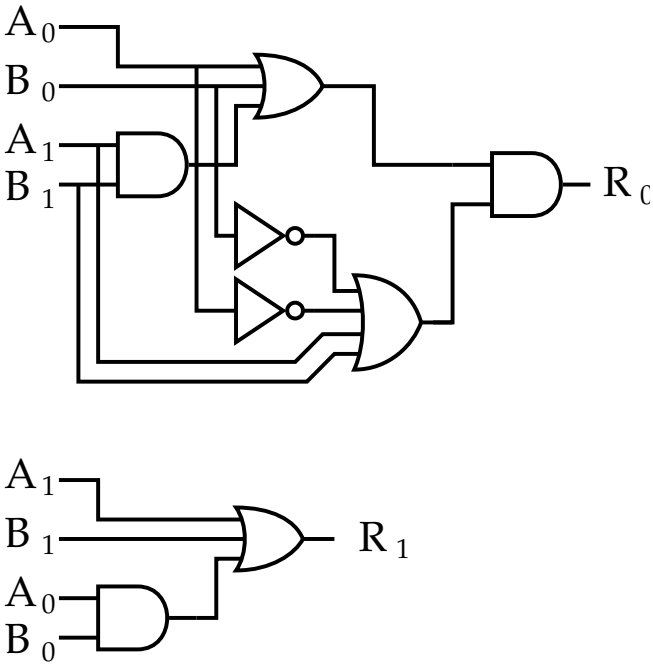


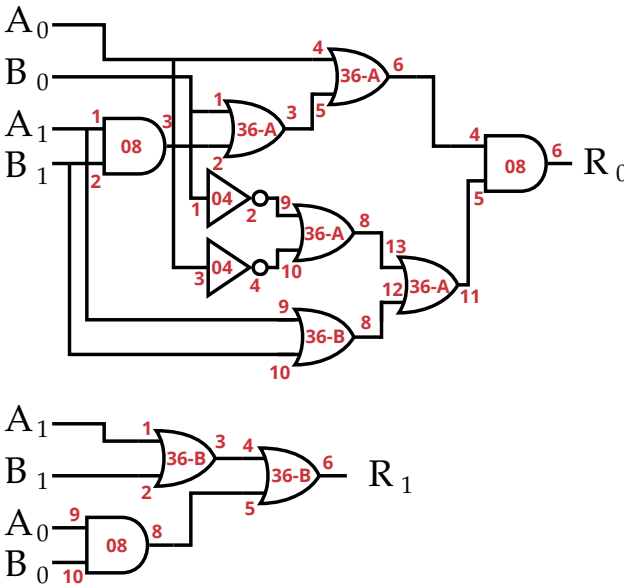
Logic equations:

R1 = (A1+A0+B1)(A1+!A0+B1+B0)
This simplifies to
R1 = ((A1 + B1) + (A0))((A1+B1) + (!A0 + B0))
= (A1 + B1) + A0(!A0 + B0)
= (A1 + B1) + A0 B0

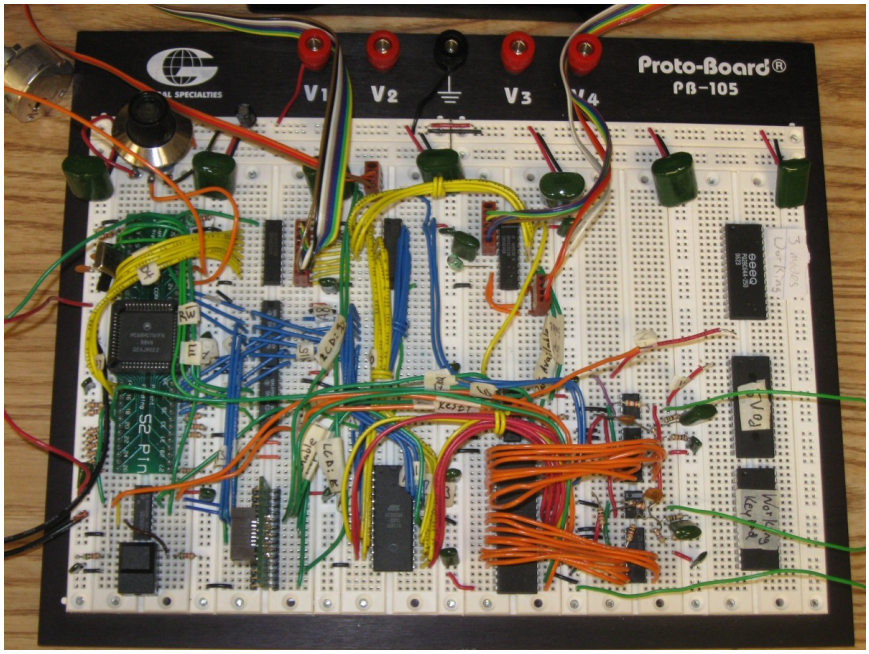
R0 = ((A0+B0)+(A1 B1))(A1 + !A0 + B1 + !B0)

Logic and schematic diagrams:





Completed circuit:



Debugging log

I started by building the high bit of the circuit (R1), since it's just two chips and three gates. Powered with 5V from bench supply.
LED is off no matter what switches I flip.
Tried checking the LED voltage; it's always 0 (so LED is probably fine). But last OR gate is always outputting 0.
Checked AND gate; everything is always 0.
OH! forgot the power supplies for the chips. Added those, now LED turns on. But it's on for all zero case → something is still wrong. Possible problems:

* Inputs aren't working correctly; check the inputs coming from the switches with the multimeter
* OR gate is broken; replace it OR gate
* Inputs to OR gate are bad; check them with the multimeter
* Breadboard is broken; tear up everything and start over

Realized that I was understanding the switches backwards; they produce logical "0" when they're in the "ON" position… weird.
So this is the 1111 state → gives 1. Correct!
Checking the other cases:

A1	A0	B1	B0	Expected	Actual
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	1	1
0	0	1	1	1	1

0	1	0	0	0	0
0	1	0	1	1	1
0	1	1	0	1	1
0	1	1	1	1	1
1	0	0	0	**1**	**0**
1	0	0	1	**1**	**0**
1	0	1	0	1	1
1	0	1	1	1	1
1	1	0	0	**1**	**0**
1	1	0	1	**1**	**0**
1	1	1	0	1	1
1	1	1	1	1	1

Several cases are bad. It seems like they're all happening when A1 should be true. Possible problems:
* OR gate is actually something else; double-check that it's a 74LS32… it is.
* A1 isn't connected to the OR gate; check input of OR gate

A1 connection was off by one pin. R1 works now.

Now let's build R0.
Testing the first term by checking the output of the second OR gate (pin 6 on the first OR chip)
(A0+B0)+(A1 B1)) It seems to be working.

Finished building the circuit. Testing with all inputs:

A1	A0	B1	B0	Expected	Actual
0	0	0	0	0	0
0	**0**	**0**	**1**	**1**	**0**
0	0	1	0	0	0
0	0	1	1	1	1
0	1	0	0	1	1
0	**1**	**0**	**1**	**0**	**1**
0	1	1	0	1	1
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	1	1	1
1	0	1	0	1	1
1	0	1	1	1	1

1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	1

This looks wrong for two cases… maybe I miswired some part of the circuit.
Check the OR gate inputs for A1 and B1. Looks good.
Check the OR gate inputs for !A0 and !B0. I missed the inverter on A0. Duh.

Now it works.

Testing

The circuit only has 4 inputs, so there are 16 possible combinations. I just tested each combination in the truth table and checked the output.

A1	A0	B1	B0	R1	R0	Correct
0	0	0	0	0	0	Y
0	0	0	1	0	1	Y
0	0	1	0	1	0	Y
0	0	1	1	1	1	Y
0	1	0	0	0	1	Y
0	1	0	1	1	0	Y
0	1	1	0	1	1	Y
0	1	1	1	1	1	Y
1	0	0	0	1	0	Y
1	0	0	1	1	1	Y
1	0	1	0	1	1	Y
1	0	1	1	1	1	Y
1	1	0	0	1	1	Y
1	1	0	1	1	1	Y
1	1	1	0	1	1	Y
1	1	1	1	1	1	Y

Reflection

1) I became much better at using the multimeter to debug my circuits. I started just clipping the negative probe to ground, and could quickly examine every node of the circuit with the positive probe. This will help me debug circuits much faster in the future.

2) I got really confused about which chips were which, and where all of my wires were going. I don't think I need more practice, but next time I will color-code my wires, and use the same colors on my logic diagram.

I also need more practice taking IC's out of the breadboard without destroying them. Next time I will try using one of the IC removal tools, and experiment on a breadboard without any wires first.

3) This lab took me an hour and a half to complete. It would have been a little faster if I hadn't miswired part of the circuit, but I felt like my debugging process was relatively efficient once I got oriented.