ECE 2372 Modern Digital Design July 17

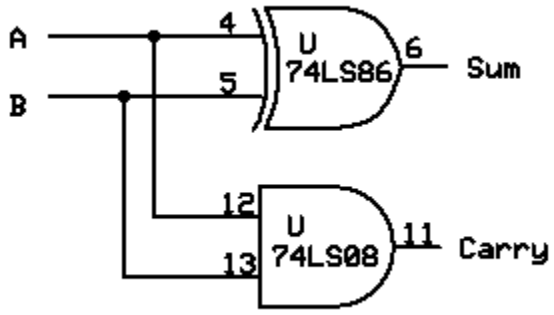
Reading assignment for today: Ch 3: section 3-9, 3-10, 3-11 Binary Adders

One thing that computers do frequently is add and subtract numbers. Since modern digital computers store information in binary format, those computers have to be able to add and subtract binary numbers. This turns out to be easy to do with combinational logic circuits. First, let's consider a truth table that describes the behavior of a binary adder.

Let's assume we have two operands, A and B, that are each a 1-bit operand, and these need to be added together. Since we can get a Carry bit from adding two single bit binary numbers, we have to include the capability for calculating a Carry bit as well as the Sum. We can see by inspection of the truth table that the Sum is simply the XOR (exclusive OR) function of A, B, and the Carry bit is simply the AND function of A, B.

operands		AND	XOR
A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

We can form what is referred to as a half-adder with simply an XOR gate and an AND gate as shown below:

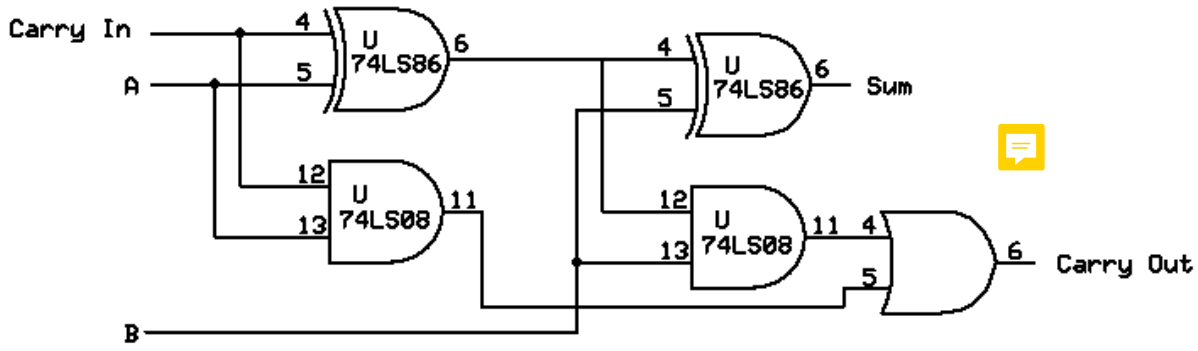


Although this certainly works, it is lacking something we need in a practical system. We also need a way to include a Carry input from a prior stage. Note that in the very first stage (Least Significant Bit) we wound not normally need a Carry input since there is no stage for it to come from, but we will see that it is handy when we want to make a subtractor from an adder. So we need to extend the idea of a half adder just developed to form a more complex circuit called a full adder. We can build the complete truth table for a full adder below:

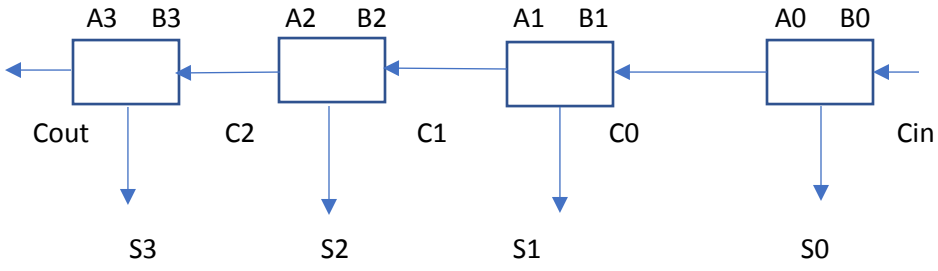
Carry In	operands		AND	XOR
	A	B	Carry out	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Referring to your textbook on page 158, this truth table can be reduced to:
Sum = ~C~AB + ~CA~B + C~A~B + CAB
Carry = CA +CB + AB

The textbook shows that with some study of the truth table and some further manipulation, it can be shown that combining two half adder circuits in the correct way forms a full adder that has all the capability we need. The full adder, formed from two half adders and an extra OR gate, is shown below:



We can connect several of these units (full binary adders) together to form a multi-bit binary adder. For example, we could connect 4 of these together to for a 4-bit ripple carry adder.



This is called a ripple carry adder because the value of the Carry bit at any stage has to be determined over a small period of time necessary for the signals (A, B operands, and Carry bits) to propagate thru

the logic gates. Suppose each of the logic gates in the full adder shown above requires 10 nS for signals to propagate thru each individual gate. If that is the case, then it will take 20 nS for the Sum to be calculated and 30 nS for the Carry out to be calculated. But, if there are 4 of these ripple carry binary adders cascaded as shown, then Carry out from the first stage takes 30 nS, then another 30 nS thru the next stage, etc, the final Carry Out takes 120 nS before it is known to be valid. The final Sum, S3 only occurs slightly sooner. 120 nS doesn't seem like a very long time, but in the world of computers that is a quite long time. Suppose we wanted to extend the ripple carry adder concept to 32 bits. 32 x 30 nS yields 960 nS to calculate Carry out for the final stage. For a 32 bit system, that means you can only do about one addition every 960 nS, or about once per microsecond. That is only 1 million additions per second. Not very fast in the modern scale of things. So, how do we solve this problem in practice? There is a somewhat complex combinational logic circuit called a look ahead carry generator. This topic is apparently not covered in the current edition of the official textbook. It is an interesting circuit, uses combinational logic circuits to in parallel generate the intermediate carry bits in a quicker way than the ripple scheme. There are also some other methods to make circuits operate faster using different approaches. Some of that we have skipped over already and will go back and pick up later.

We will stop here for today. Next time we will finish up some topics on binary ripple adders, in particular how to easily convert a binary adder to a binary subtractor by applying 2's complement conversion. Then we will move into chapter 4, Sequential Circuits. At some point, we will go back and pick up some additional material from ch 3 that we have skipped for now.