One's complement addition

- To add one's complement numbers:
  - First do unsigned addition on the numbers, *including* the sign bits.
  - Then take the carry out and add it to the sum.
- Two examples:

$$\begin{array}{rr} & 0111 \\ + & 1011 \\ \hline 1 & 0010 \end{array} \quad \begin{array}{rr} & (+7) \\ + & (-4) \\ \hline \end{array} \qquad\qquad \begin{array}{rr} & 0011 \\ + & 0010 \\ \hline 0 & 0101 \end{array} \quad \begin{array}{rr} & (+3) \\ + & (+2) \\ \hline \end{array}$$

$$\begin{array}{rr} & 0010 \\ + & 1 \\ \hline & 0011 \end{array} \quad (+3) \qquad\qquad \begin{array}{rr} & 0101 \\ + & 0 \\ \hline & 0101 \end{array} \quad (+5)$$

- This is simpler and more uniform than signed magnitude addition.

Two's complement addition

- Negating a two's complement number takes a bit of work, but addition is much easier than with the other two systems.
- To find A + B, you just have to:
  - Do unsigned addition on A and B, including their sign bits.
  - Ignore any carry out.
- For example, to find 0111 + 1100, or (+7) + (-4):
  - First add 0111 + 1100 as unsigned numbers:

$$\begin{array}{rr} & 0111 \\ + & 1100 \\ \hline 1 & 0011 \end{array}$$

  - Discard the carry out (1).
  - The answer is 0011 (+3).

Comparing the signed number systems

- Here are all the 4-bit numbers in the different systems.
- *Positive numbers are the same in all three representations.*
- Signed magnitude and one's complement have *two* ways of representing 0. This makes things more complicated.
- Two's complement has asymmetric ranges; there is one more negative number than positive number. Here, you can represent -8 but not +8.
- However, two's complement is preferred because it has only one 0, and its addition algorithm is the simplest.

Decimal	S.M.	1's comp.	2's comp.
7	0111	0111	0111
6	0110	0110	0110
5	0101	0101	0101
4	0100	0100	0100
3	0011	0011	0011
2	0010	0010	0010
1	0001	0001	0001
0	0000	0000	0000
−0	1000	1111	—
−1	1001	1110	1111
−2	1010	1101	1110
−3	1011	1100	1101
−4	1100	1011	1100
−5	1101	1010	1011
−6	1110	1001	1010
−7	1111	1000	1001
−8	—	—	1000

Ranges of the signed number systems

- How many negative and positive numbers can be represented in each of the different systems on the previous page?

	Unsigned	Signed Magnitude	One's complement	Two's complement
Smallest	0000 (0)	1111 (-7)	1000 (-7)	1000 (-8)
Largest	1111 (15)	0111 (+7)	0111 (+7)	0111 (+7)

- In general, with n-bit numbers including the sign, the ranges are:

	Unsigned	Signed Magnitude	One's complement	Two's complement
Smallest	0	$-(2^{n-1}-1)$	$-(2^{n-1}-1)$	-2^{n-1}
Largest	2^n-1	$+(2^{n-1}-1)$	$+(2^{n-1}-1)$	$+(2^{n-1}-1)$

Example solution

- Convert 110101 to decimal, assuming this is a number in:

  Since the sign bit is 1, this is a negative number. The easiest way to find the magnitude is to convert it to a positive number.

  (a) signed magnitude format

  Negating the original number, 110101, gives 010101, which is +21 in decimal. So 110101 must represent -21.

  (b) ones' complement

  Negating 110101 in ones' complement yields 001010 = $+10_{10}$, so the original number must have been -10_{10}.

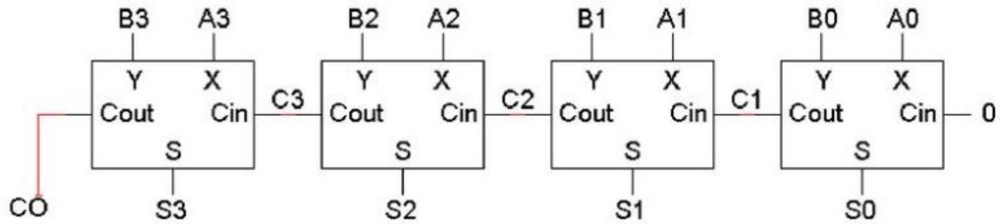
  (c) two's complement

  Negating 110101 in two's complement gives 001011 = 11_{10}, which means 110101 = -11_{10}.

- The most important point here is that a binary number has *different* meanings depending on which representation is assumed.

Our four-bit unsigned adder circuit

- Here is the four-bit unsigned addition circuit from an earlier lecture.



Making a subtraction circuit

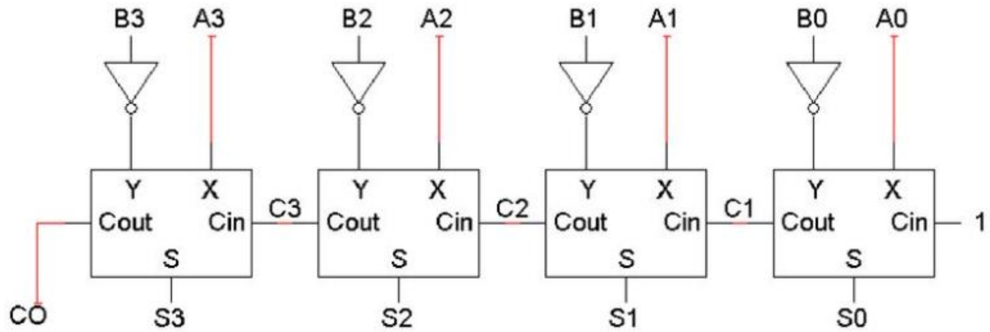
- We could build a subtraction circuit directly, similar to the way we made unsigned adders yesterday.
- However, by using two's complement we can convert any subtraction problem into an addition problem. Algebraically,

$$A - B = A + (-B)$$

- So to subtract B from A, we can instead *add* the negation of B to A.
- This way we can re-use the unsigned adder hardware from last week.

A two's complement subtraction circuit

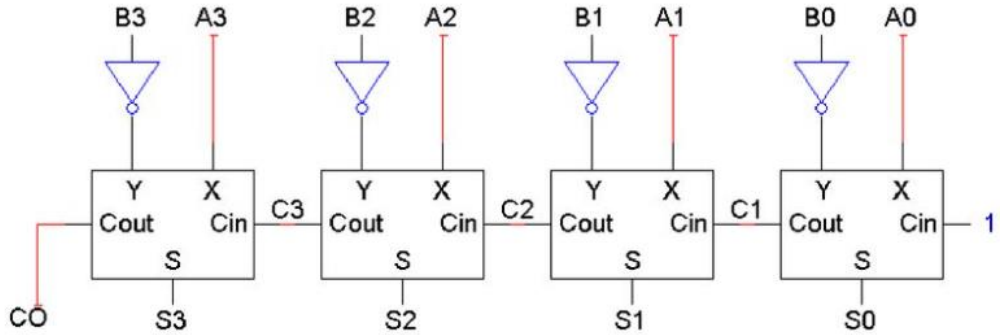
- To find A - B with an adder, we'll need to:
  - Complement each bit of B.
  - Set the adder's carry in to 1.
- The net result is A + B' + 1, where B' + 1 is the two's complement negation of B.



- Remember that A3, B3 and S3 here are actually sign bits.

Small differences

- The only differences between the adder and subtractor circuits are:
  - The subtractor has to negate B3 B2 B1 B0.
  - The subtractor sets the initial carry in to 1, instead of 0.



- It's not too hard to make one circuit that does *both* addition and subtraction.

An adder-subtractor circuit

- XOR gates let us selectively complement the B input.

$$X \oplus 0 = X \qquad\qquad X \oplus 1 = X'$$

- When Sub = 0, the XOR gates output B3 B2 B1 B0 and the carry in is 0. The adder output will be A + B + 0, or just A + B.
- When Sub = 1, the XOR gates output B3' B2' B1' B0' and the carry in is 1. Thus, the adder output will be a two's complement subtraction, A - B.