Subtraction Using 1's Complement

- Use addition of the 1's complement to subtract two numbers:
  1. **When an end carry occurs:** Since $(r-1)$'s complement is one less than the r's complement, the result is one less than the correct sum. $\Longrightarrow$ Discard the end carry and add one to the sum (**end-around carry**).
  2. **When no end carry occurs:** The result is negative. $\Longrightarrow$ Use 1's complement to represent the negative result.

Find 1010100 – 1000011	Find 1000011 – 1010100
Step 1: 1's complement of 1000011 is 0111100.	Step 1: 1's complement of 1010100 is 0101011.
Step 2: $\begin{array}{r} 1010100 \\ +\ 0111100 \\ \hline 1\ 0010000 \end{array}$	Step 2: $\begin{array}{r} 1000011 \\ +\ 0101011 \\ \hline 1101110 \end{array}$
Step 3: A carry occurs, add carry, end around.	Step 3: No carry occurs, 1's complement the result.
Ans: 0010001	Ans: (–) 0010001

Signed Binary Numbers

- Use the **most significant bit** as a **sign bit**: $sa_{n-2}\ldots a_0$.
  - $s=0$: positive numbers; $s=1$: negative numbers.
- Three popular ways to interpret $a_{n-2}\ldots a_0$:
  - **Signed magnitude:** The $n-1$ digits are interpreted as a positive magnitude.
  - **Signed 1's complement:** Use the 1's complement to compute the **negative** numbers.
  - **Signed 2's complement:** Use the 2's complement to compute the **negative** numbers.
- Exp: Signed-magnitude: 10001001 (-9); signed 1's complement: 11110110 (-9); signed 2's complement: 11110111 (-9).
- For positive numbers, the three ways give the same representation.

Three Signed-Number Representations

Decimal	Signed 2's Compement	Signed 1's Complement	Signed Magnitude
+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	–	1111	1000
–1	1111	1110	1001
–2	1110	1101	1010
–3	1101	1100	1011
–4	1100	1011	1100
–5	1011	1010	1101
–6	1010	1001	1110
–7	1001	1000	1111
–8	1000	–	–

Signed-Magnitude Arithmetic

- Addition
  - **If signs are the same:**
    1. Add the magnitudes.
    2. Check for **overflow** (a carry into the sign bit).
    3. The sign of the result is the same.
  - **If signs differ:**
    1. Subtract the magnitude of the smaller from the magnitude of the larger.
    2. Use the sign of the larger magnitude for the sign of the result.
    3. **Overflow will never occur.**
- Subtraction: Complement the sign bit of the number you are subtracting and follow the rules for addition.
- Same signs: $000 + 011 = 011$; $010 + 010 = x00$ (**overflow!**); $101 + 101 = 110$; $110 + 110 = x00$ (**overflow!**).
- Different signs: $001 + 110 = 101$ ($010 - 001$, take $-$ sign); $111 + 010 = 101$; $100 + 000 = ?00$ ($+$ or $-$ zero?).

Signed-Complement Arithmetic

- Addition
  1. Add the numbers (including the sign bits),
     - ∗ 2's complement: discarding a carry out of the sign bit.
     - ∗ 1's complement: using an end-around carry.
  2. If the sign bits were the same for both numbers and the result is different, an overflow has occurred.
  - – Boolean equation for overflow:

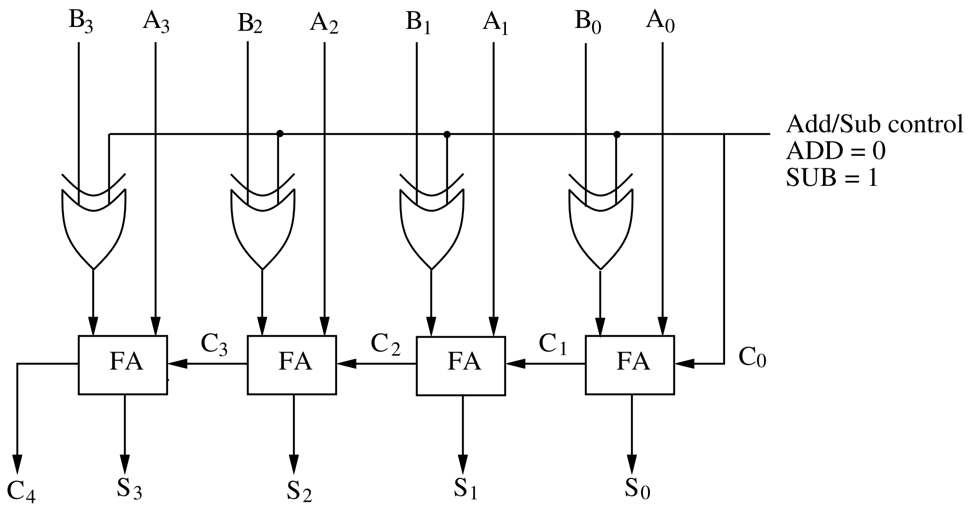
$$Overflow = A_{n-1}B_{n-1}\overline{S_{n-1}} + \overline{A_{n-1}}\ \overline{B_{n-1}}S_{n-1}.$$

$$\begin{array}{rl} +70 & 0\ 1000110 \\ +80 & 0\ 1010000 \\ \hline +150 & 1\ 0010110 \end{array} \qquad \begin{array}{rl} -70 & 1\ 0111010 \\ -80 & 1\ 0110000 \\ \hline -150 & 0\ 1101010 \end{array}$$

- Subtraction: Form the complement of the number you are subtracting and follow the rules for addition.

Binary Adder/Subtractor

- Subtraction can be achieved by adding 2's complement.
  - – Complement each bit (1's complement) and add 1 to the result.
- $C_0 = 0$: addition; $C_0 = 1$: Form the 1's complement of B using EXOR gates and add one.



Binary Multiplication

- **Compute partial products**, and **justify and sum** the partial products.
  - $m \times n$ digit multiply generates up to an $(m+n)$-digit result.
- **Compute partial products:**
  - n-digit, base-2 numbers require adding up to n digits (with carries) in a column.
  - Multiply the row of multiplicand digits by each multiplier digit, one at a time.
  - If the multiplier digit is 0 $\Longrightarrow$ all zero; if the multiplier digit is 1 $\Longrightarrow$ the same as the multiplicand.

$$
\begin{array}{rrrrrr}
 & & & 1 & 0 & 1 \\
 & & \times & 0 & 1 & 1 \\
\hline
 & & & 1 & 0 & 1 \\
 & & 1 & 0 & 1 & \\
 & 0 & 0 & 0 & & \\
\hline
0 & 0 & 1 & 1 & 1 & 1
\end{array}
$$

2×2 Multiplier

- For $m \times n$ digit multiply, we need $m \times n$ AND gates and $(n-1)$ m-bit adders to produce a product of $m+n$ bits.

$$
\begin{array}{cccc}
 & & B_1 & B_0 \\
 & & A_1 & A_0 \\
\hline
 & & A_0B_1 & A_0B_0 \\
 & A_1B_1 & A_1B_0 & \\
\hline
P_3 & P_2 & P_1 & P_0
\end{array}
$$

$P_0 = A_0 B_0$

$P_1 = A_0B_1 \oplus A_1B_0$

$P_2 = A_1B_1 \oplus (A_0B_1 \bullet A_1B_0)$

$P_3 = A_1B_1 \bullet (A_0B_1 \bullet A_1B_0)$