Arithmetic

• Learning Objectives
  • Explain how to encode both positive and negative numbers in binary
  • Add and subtract binary numbers
  • Explain why 2’s complement is a good representation
int main(int argc, char ** argv)
{
    /* Suppress compiler warnings. */
    (void)argc;
    (void)argv;

    int negs[4] = {-1, -2, -3, -4};
    hexdump(negs, sizeof(negs));
}
Negative Numbers

• What do we know so far?
  • Negative (small) numbers appear very large (they start with 0xff).
  • Counting in negative numbers does seem to follow a regular counting pattern.
• You might hypothesize that negative numbers are positive numbers with the bits flipped.
• How might you test that?
2’s Complement Arithmetic

- We can formalize what we observed, simply by examining values in memory:
- We represent \(-n\) by taking the inverse of \(n\) (\(~n\)) and adding 1 to it:
  - \(-n = (~n) + 1\)
- Example (in 8 bits):
  - \(N = 5; 5 = 00000101\)
  - \(~N = 11111010\) (\(0xFF\))
  - \(~N + 1 = 11111011 = 0xFB\)
Why 2’s Complement

• “It makes arithmetic operations just work.”
• How do we add binary numbers?
  • 0 + 0 = 0
  • 0 + 1 = 1
  • 1 + 1 = 0 (carry the 1)
• Example (again in 8 bits)
  • 35 + 5
    \[
    \begin{array}{c}
    00100011 \\
    00000101 \\
    \hline
    01100100 \\
    \hline
    01100110
    \end{array}
    \]
    \[
    \begin{array}{c}
    0 \times 2^3 \\
    0 \times 2^2 \\
    0 \times 2^1 \\
    0 \times 2^0
    \end{array}
    \]
Addition with Negative Numbers

• Let’s see what happens when we try to add two negative numbers.

• Example: \(-3 + -10\)

\[
\begin{align*}
(-3) & \quad 1111101 \\
(+1) & \quad 11110110 \\
\hline \\
\text{Sum} & \quad 00001010 \\
\text{Overflow} & \quad 01110110 \\
\text{Corrected} & \quad 11110101 \\
\end{align*}
\]

• Final check: \(-1 + 1\) ought to equal 0!

\[
\begin{align*}
(-1) & \quad 1111111 \\
(+1) & \quad 0000001 \\
\hline \\
\text{Sum} & \quad 00001100 \\
\text{Overflow} & \quad 10000110 \\
\text{Corrected} & \quad 0000001 \\
\end{align*}
\]

\[= 01110110 + 1 = 00001010\]