Lecture 4: Integer arithmetic

September 9, 2019

601.229 Computer System Fundamentals

Integer arithmetic

- Integer representations based on fixed-size machine words are *finite*
- I.e., only a finite number of possible values can be represented
  - For word with w bits, can represent 2^w possible values
- So, we should expect some (potentially) strange results when doing arithmetic using machine words
- These strange results can lead to surprising program behavior, including security vulnerabilities

Addition of unsigned values

- Same idea as what you learned in grade school
  - Start with least significant digit
  - As needed, carry excess into next-most-significant digit

Addition of unsigned values (no overflow)

Example: $0110 + 0111$

$$
\begin{array}{rr}
 & 0 \\
 & 0110 \\
+ & 0111 \\
\hline
\end{array}
$$

Addition of unsigned values (no overflow)

Example: $0110 + 0111$

$$\begin{array}{rr} & \mathbf{0}0 \\ & 011\mathbf{0} \\ + & 011\mathbf{1} \\ \hline & \mathbf{1} \end{array} \quad \text{no carry}$$

Addition of unsigned values (no overflow)

Example: 0110 + 0111

$$
\begin{array}{rr}
 & \mathbf{1}00 \\
 & 01\mathbf{1}0 \\
+ & 01\mathbf{1}1 \\
\hline
 & \mathbf{0}1
\end{array}
\quad \text{carry 1}
$$

Addition of unsigned values (no overflow)

Example: 0110 + 0111

$$
\begin{array}{rr}
 & \mathbf{1}000 \\
 & 0\mathbf{1}10 \\
+ & 0\mathbf{1}11 \\
\hline
 & \mathbf{1}01 \quad \text{carry } 1
\end{array}
$$

Addition of unsigned values (no overflow)

Example: 0110 + 0111

$$\begin{array}{rr} & \mathbf{0}1000 \\ & \mathbf{0}110 \\ + & \mathbf{0}111 \\ \hline & \mathbf{1}101 \end{array} \quad \text{no carry}$$

Addition of unsigned values (no overflow)

Example: $0110 + 0111$

$$\begin{array}{rr} & 0110 \\ + & 0111 \\ \hline & 1101 \end{array} \quad \text{done}$$

Overflow

- If the sum of w-bit (unsigned) integer values is too large to represent using a w-bit word, *overflow* occurs
- Effective sum of w bit integers a and b is

$$(a + b) \bmod 2^w$$

Addition of unsigned values (overflow)

Example: $1110 + 0111$

$$
\begin{array}{rr}
 & 0 \\
 & 1110 \\
+ & 0111 \\
\hline
\end{array}
$$

Addition of unsigned values (overflow)

Example: 1110 + 0111

$$\begin{array}{rr} & \mathbf{0}0 \\ & 111\mathbf{0} \\ + & 011\mathbf{1} \\ \hline & \mathbf{1} \end{array}$$ no carry

Addition of unsigned values (overflow)

Example: 1110 + 0111

$$
\begin{array}{rr}
 & \mathbf{1}00 \\
 & 11\mathbf{1}0 \\
+ & 01\mathbf{1}1 \\
\hline
 & \mathbf{0}1
\end{array}
\quad \text{carry } 1
$$

Addition of unsigned values (overflow)

Example: $1110 + 0111$

$$\begin{array}{rr} & \mathbf{1}100 \\ & 1\mathbf{1}10 \\ + & 0\mathbf{1}11 \\ \hline & \mathbf{1}01 \end{array} \quad \text{carry } 1$$

Addition of unsigned values (overflow)

Example: 1110 + 0111

$$\begin{array}{rr} & \mathbf{1}1100 \\ & \mathbf{1}110 \\ + & \mathbf{0}111 \\ \hline & \mathbf{0}101 \end{array} \quad \text{carry } 1$$

Addition of unsigned values (overflow)

Example: 1110 + 0111

$$
\begin{array}{rr}
 & 11100 \\
 & 1110 \\
+ & 0111 \\
\hline
 & 1\underline{0101}
\end{array}
$$

True sum is 10101 (21), effective sum is 101 (5)
(note 21 mod 16 = 5)

Clicker quiz

Clicker quiz omitted from public slides

Addition of signed values

Useful property of two's complement: addition is carried out *exactly the same way* for signed values as for unsigned values

Addition of signed values

Useful property of two's complement: addition is carried out *exactly the same way* for signed values as for unsigned values

Signed addition example

Example: 0101 (5) + 1110 (-2)

$$\begin{array}{r} 0101 \\ + \quad 1110 \\ \hline \end{array}$$

Signed addition example

Example: 0101 (5) + 1110 (-2)

$$\begin{array}{r} 0101 \\ + \quad 1110 \\ \hline 1\underline{0011} \end{array}$$

After truncating (discarding high bit of sum),
effective sum is 0011 (3)

Signed overflow

What happens when sum of signed w-bit values can't be represented?

- If sum exceeds $2^{w-1} - 1$, it becomes negative (overflow)
- If sum is less than -2^{w-1}, it becomes positive (negative overflow)

Signed addition example (overflow)

Example: 0100 (4) + 0101 (5)

$$\begin{array}{rr} & 0100 \\ + & 0101 \\ \hline \end{array}$$

Signed addition example (overflow)

Example: 0100 (4) + 0101 (5)

$$
\begin{array}{rr}
 & 0100 \\
+ & 0101 \\
\hline
 & 1001
\end{array}
$$

Result is -7 (-8 + 1)

Signed addition example (negative overflow)

Example: 1100 (-4) + 1011 (-5)

$$\begin{array}{r} 1100 \\ +\quad 1011 \\ \hline \end{array}$$

Signed addition example (negative overflow)

Example: 1100 (-4) + 1011 (-5)

$$\begin{array}{rr} & 1100 \\ + & 1011 \\ \hline & 1\underline{0111} \end{array}$$

Result (after truncating) is 7

Clicker quiz

Clicker quiz omitted from public slides

Two's complement negation and subtraction

- Negation: if x is a two-complement integer value, $-x$ can be computed by inverting bits of x, then adding 1
  - Why?
- Subtraction:

$$a - b = a + -b$$

  I.e., to compute $a - b$, compute $-b$, then add $-b$ to a

Clicker quiz

Clicker quiz omitted from public slides

Integer arithmetic in C

Yeah