Announcements

- C Workshop on C functions
  - Thu 7-9 p.m.

Signed Multiplication

- Operands: w bits
- True Product: 2*w bits
- Discard w bits: w bits

- Ignores high order w bits
- Different interpretation for signed vs. unsigned multiplication
- Lower bits are the same

Signed Multiplication in Practice

- Method 1
  - Find the magnitude of the two multiplicands
  - Multiply them together
  - Determine the sign
    - Same sign → positive
    - Different sign → negative
- Method 2
  - Sign-extend the two multiplicands
  - Multiply them together
  - Extract low w bits

Pop Quiz

Assume a 3-bit signed binary representation.

111 * 011 = ?
-1 * 3

Pop Quiz

Assume a 3-bit signed binary representation.

111 * 011 = 101
-1 * 3 = -3
Pop Quiz

Assume a 3-bit signed binary representation.

100 * 111 = ?

-4 * -1 = -4 (?)

---

**Power of-2 Multiply with Shift**

- **Operation**
  - $u << k$ gives $u \times 2^k$
  - for both signed and unsigned

- **Operands:** $u$ bits
  - $u$ = [*** ... ***]

- **True Product:** $w+e$ bits
  - $w$ = [*** ... ***]
  - $e = 2^k$

- **Discard $k$ bits:** $w$ bits
  - $(UMult(u, 2^k))$ = [*** ... ***]

- **Examples**
  - $u << 3 = u \times 8$
  - $u \times 12 = ?$

---

**Dividing by Powers of 2**

- **Integer division much slower than multiplication**
- **Dividing by $2^k$ can be done by a right shift**
  - logical right shift
  - arithmetic right shift
- **Integer division always rounds toward zero.**

---

**Unsigned Power of-2 Divide with Shift**

- **Quotient of unsigned by power of 2**
  - $u >> k$ gives $\lfloor u / 2^k \rfloor$
  - Uses logical shift

---

**Signed Power of-2 Divide with Shift**

- **Quotient of signed by power of 2**
  - $u >> k$ gives $\lfloor u / 2^k \rfloor$
  - Uses arithmetic shift
  - Rounds wrong direction when $u < 0$ (round down!)
Correct Power-of-2 Divide

- Quotient of Negative Number by Power of 2
  - Want \( \lceil x / 2^k \rceil \) (round toward 0)

- Use the property (from CPSC 203)
  \[ \lceil a / b \rceil = \lceil (a + b - 1) / b \rceil \] (adding a bias)
  which gives

- Compute \( \lceil x / 2^k \rceil \) as
  \[ \lfloor (x + 2^k - 1) / 2^k \rfloor \]

- In C:
  \( (x + (1 << k) - 1) >> k \)