# Numerical Overflow (Practice)

ICS312

**Machine-Level and Systems Programming**

Henri Casanova (henric@hawaii.edu)

# (q1) Unsigned Overflow

- For each UNSIGNED operation below say whether it overflows

- 1-byte: 05 + F1
- 1-byte: FA + 60
- 1-byte: 56 - 3F
- 1-byte: B3 - FF
- 2-byte: FFFF - E410
- 4-byte: F6420BA4 + 0A121212

# (q1) Solutions

- For each UNSIGNED operation below say whether it overflows

- 1-byte: 05 + F1 — NO OVERFLOW
- 1-byte: FA + 60 — OVERFLOW
- 1-byte: 56 - 3F — NO OVERFLOW
- 1-byte: B3 - FF — OVERFLOW
- 2-byte: FFFF - E410 — NO OVERFLOW
- 4-byte: F6420BA4 + 0A121212 — OVERFLOW

# (q2) Signed Overflow

- For each SIGNED operation below say whether it overflows

- 1-byte: 05 + 7E
- 1-byte: FA + 60
- 1-byte: 76 - 81
- 1-byte: B3 - FF
- 2-byte: FFFF - E410
- 4-byte: F6420BA4 + 7FFFFFF

# (q2) Solutions

- For each SIGNED operation below say whether it overflows

- 1-byte: 05 + 7E — OVERFLOW
- 1-byte: FA + 60 — NO OVERFLOW
- 1-byte: 76 - 81 — OVERFLOW
- 1-byte: B3 - FF — NO OVERFLOW
- 2-byte: FFFF - E410 — NO OVERFLOW
- 4-byte: F6420BA4 + 7FFFFFF — NO OVERFLOW

# (q3) Carry bit / Overflow bit

- For each operation below show CF and OF

- 1-byte: AF + 70
- 2-byte: BF12 + 3FFE
- 1-byte: AF + 84
- 2-byte: 5001 + 6FFE

# (q3) Solutions

- For each operation below show CF and OF

- 1-byte: AF + 70 CF=1 OF=0
- 2-byte: BF12 + 3FFE CF=0 OF=0
- 1-byte: AF + 84 CF=1 OF=1
- 2-byte: 5001 + 6FFE CF=0 OF=1

# (q4) Carry/Overflow Bit

- For each 1-byte operation below, say whether the overflow bit (of) and the carry bit (cf) configuration is possible (given that one digit can be picked to be any digit between 0 and F)

- 05 + ?E  cf=0  of=1
- 14 + ?E  cf=1  of=1
- F2 + 0?  cf=1  of=0
- 80 + ?1  cf=1  of=0
- 80 + ?1  cf=0  of=1
- A0 + ?4  cf=1  of=1

# (q4) Solutions

- For each 1-byte operation below, say whether the overflow bit (of) and the carry bit (cf) configuration is possible (given that one digit can be picked to be any digit between 0 and F)

| | |
|---|---|
| 05 + ?E cf=0 of=1 | POSSIBLE (e.g., ?=7) |
| 14 + ?E cf=1 of=1 | NOT POSSIBLE |
| F2 + 0? cf=1 of=0 | POSSIBLE (e.g., ?=F) |
| 80 + ?1 cf=1 of=0 | NOT POSSIBLE |
| 80 + ?1 cf=0 of=1 | NOT POSSIBLE |
| A0 + ?4 cf=1 of=1 | POSSIBLE (e.g., ?=A) |

# (q5) High-Level Code

- Is this way to detect unsigned overflow in some high-level language correct?

```
unsigned int a, b;
…
if (a - b < 0){
  throw OverflowException;
}
unsigned int result = a - b
```

# (q5) Solution

- Is this way to detect unsigned overflow in some high-level language correct?

```
unsigned int a, b;
…
if (a - b < 0){
  throw OverflowException;
}
unsigned int result = a - b
```

NO. Should be “a < b”

# (q6) High-Level Code

- How should one detect overflow for the following code in some high-level language?

```
unsigned int a, b;
…
if ( ??? ){
   throw OverflowException;
}
unsigned int result = a + b
```

# (q6) Solution

- How should one detect overflow for the following code in some high-level language?

```
unsigned int a, b;
…
if ( a  > UINT_MAX - b ){
   throw OverflowException;
}
unsigned int result = a + b
```