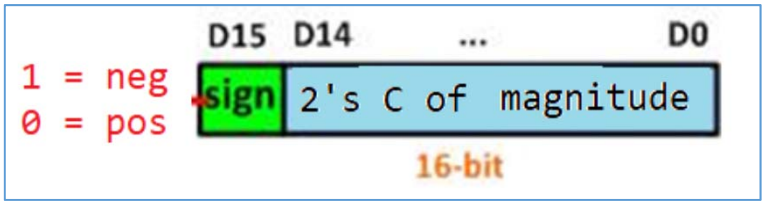
Chapter 5: Signed Integer Number Arithmetic

- Signed integer instructions and operations
- Signed Number
  - MSB (Most Significant Bit) is set aside for sign
    - 0 for pos(+) 1 for neg(-)
  - Rest of the bits: magnitude
    - Sign-2's Complement format
      - +1: 0000 0000 0000 0001
      - -1: 1111 1111 1111 1111
      - Simplified ALU circuitry
      - This is the **standard** format



2’s Complement Format

- Sign-2’s Complement format

- Practice (for 32-bit data size)
  - Find the representation of -5
  - Find the representation of -0x1000
  - Find the representation of   -0xF001

Signed Number Instructions

Overflow Problem in signed number operations

Example 1)8-bit size case:

(+96) + (+70) = (+ 166)

or 0x60 + 0x46 = 0xA6

Signed Number Instructions

Overflow Problem in signed number operations

Example 2) 8-bit size case:

(-128) + (-2) = (-130)

or 0x80 + 0xFE = 0x7E

Signed Number Instructions

Overflow Problem in signed number operations

Overflow flag (V) setting:

V = 1 if there is Carry to MSB but no Carry (out of MSB)

V = 1 if there is Carry (out of MSB) but no Carry to MSB

Or

V = (Carry to MSB)EOR(Carry) // Excluive-OR

If V = 1, the result is incorrect

EX1) For 8-bit data size, if A=+7 and B=+18, check the flags from the result of ADD A, B

C = ____ , N = ____ , V = ____ .

Ans.

Signed Number Instructions

EX2) For 16-bit data size, if A=0x6E2F and B=0x13D4, check the flags from the result of ADD A, B

C = ____ , N =___ , V = ____ .

Ans.

Signed Number Instructions

CLASS ACTIVITY

Q1) For 32-bit data size, if A=0x6E2F356F and B=0x13D49530, check the result of ADD A, B if it is correct or incorrect.

C = _____ , N =____ , V = ____ .

Q2) For 32-bit data size, if A=0x542F356F and B=0x12E09530, check the result of ADD A, B if it is correct or incorrect.

C = ____, N =____ , V = ____ .

Sign Extension

Overflow in signed number operation is caused by the limited data size of the result.

We can extend the operands to a larger data size.

"**Sign extension**" – copy the MSB of the operand to every bit of the upper bits

EX) 8-bit operand: 0x7F → extend to 16-bit : 0x007F

EX) 8-bit operand: 0x8F → extend to 16-bit : 0xFF8F

EX) 16-bit operand: 0xABCD → extend to 32-bit: 0xFFFFABCD

ARM Instruction

LDRSB (load register signed byte): 8-bit → 32-bit

LDRSH (load register signed half-word): 16-bit → 32-bit

Signed Number Multiplication

SMULL (signed multiply long)

- Similar to the unsigned multiplications: MUL & UMULL
- Multiplication of two 32-bit signed numbers and the product is in a 64-bit signed number
- Format: SMULL RdLo, RdHi, Rm, Rs

  //RdHi:RdLo ← Rm * Rs
- **Example: Write code which calculates -3500 * -100 and stores the product to r3|r2. Check the result if it is correct or incorrect.**